МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
федеральное государственное бюджетное образовательное учреждение высшего образования
КРАСНОЯРСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.П.АСТАФЬЕВА
(КГПУ им. В.П.Астафьева)

Институт/факультет — Математики, физики и информатики
(полное наименование института/факультета/филиала)

Выпускающая кафедра — Базовая кафедра Информатики и информационных технологий в образовании
(полное наименование кафедры)

Нигматулина Эльмира Альфредовна

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема **Методика телесно-полиязыкового обучения будущих учителей информатики программированию**

Направление подготовки — 44.04.01 Педагогическое образование
(код и наименование направления)

Направленность (профиль) образовательной программы — Информатика в образовании
(наименование программы)

ДОПУСКАЮ К ЗАЩИТЕ
Заведующий кафедрой
д.п.н., профессор Пак Н.И.
(ученая степень, ученое звание, фамилия, инициалы)
(дата, подпись)

Руководитель магистерской программы
д.п.н., профессор Пак Н.И.
(ученая степень, ученое звание, фамилия, инициалы)
(дата, подпись)

Научный руководитель
К.п.н., доцент Степанова Т.А.
(ученая степень, ученое звание, фамилия, инициалы)
(дата, подпись)

Обучающийся Нигматулина Э.А.
(фамилия, инициалы)
(дата, подпись)

Красноярск 2017

Реферат

Магистерская диссертация состоит из 134 страниц, 20 рисунков, 3 таблиц, 117 источников литературы и 4 приложений.

Краткая характеристика работы

Объект исследования: процесс обучения студентов педагогического вуза курсу «Языки и методы программирования».

Предмет исследования: методика формирования программистского стиля мышления будущих учителей информатики в процессе обучения курсу «Языки и методы программирования» на основе телесно-полиязыкового подхода.

Цель исследования: теоретическое обоснование и разработка методики обучения студентов курсу «Языки и методы программирования» на основе телесно-полиязыкового подхода.

Задачи исследования:

1. Проанализировать состояние разработанности проблемы исследования в психолого-педагогической, научно-методической науке и практике.

2. Уточнить содержание понятия «программистский стиль мышления», построить структурную модель программистского стиля мышления и определить этапы его формирования, выделить уровни его сформированности и критерии их измерения.

3. Выявить возможности использования телесного подхода и разработать концепцию полиязыкового подхода к обучению студентов программированию.

4. Разработать структурно-логическую модель обучения студентов курсу «Языки и методы программирования» с применением средств (кинестетические тренажеры) и методов (параллельное изучение языков программирования) телесного и полиязыкового подходов.

5. Разработать комплекс натурных, кинестетических и полиязыковых средств обучения.

В рамках магистерской диссертации были использованы следующие **методы исследования:** теоретические (изучение и анализ педагогической, психологической, методической и предметной литературы по теме исследования, анализ теоретических и эмпирических данных, изучение и обобщение педагогического опыта, сравнительный анализ, классификация); эмпирические (наблюдение, анкетирование, беседа, тестирование, педагогический эксперимент).

Научная новизна исследования заключается в том, что:

1. Определена категорийно-понятийная компонента «программистский стиль мышления» в структуре алгоритмического мышления, позволяющая оценивать его уровень сформированности в процессе обучения курсу «Языки и методы программирования».

2. Обоснована необходимость и возможность использования телесного и полиязыкового подходов к обучению студентов в процессе предметной подготовки, обеспечивающие формирование их программистского стиля мышления.

3. Разработана методика телесно-полиязыкового обучения студентов программированию, способствующая формированию у них высокого уровня программистского стиля мышления и обеспечивающая их методическую готовность осуществлять подготовку школьников в области программирования за счет натурных, кинестетических и полиязыковых средств и методов обучения программированию.

Теоретическая значимость исследования заключается:

1) в уточнении понятия «программистский стиль мышления», определении этапов, уровней его сформированности и разработки критериев их оценки;

2) в построении структурно-логической модели обучения студентов курсу «Языки и методы программирования» на основе телесного и полиязыкового подходов.

Практическая значимость исследования заключается в том, что:

- разработан комплекс алгоритмических натурных, кинестетических и полиязыковых средств обучения студентов программированию;

- опубликован в соавторстве с группой авторов (Нигматулина Э.А., Пак Н.И., Сокольская М.А., Степанова Т.А.) учебник «Программирование» в 2-х томах для бакалавров педагогического образования профиль «информатика» (М.: Академия, гриф УМО по направлению «Педагогическое образования»), отражающий концепцию полиязыкового обучения программированию.

- разработанная методика телесно-полиязыкового обучения студентов программированию может быть использована в учебном процессе в педагогических вузах, на курсах повышения квалификации.

Результаты магистерского исследования используются в учебном процессе:

1) бюджетного общеобразовательного учреждения «Кириковская средняя школа». В частности: в образовательный процесс при изучении отдельных тем по программированию внедрены натурные и кинестетические тренажеры, частично использована методика полиязыкового обучения с использованием системы практико-ориентированных заданий.

2) Лесосибирского педагогического института – филиала СФУ. В частности: частично апробирован полиязыковой подход в обучении дисциплине «Программирование» по направлению подготовки 44.03.01 «Педагогическое образование» профиль 44.03.01.33 «Информатика».

Оглавление

Введение

Современное информационное, наукоёмкое общество нуждается в людях с высоким уровнем развития абстрактного мышления вообще и алгоритмического мышления в частности, поскольку осуществление абсолютно любой деятельности – не только профессиональной, где это является необходимым условием в настоящее время практически во всех профессиях, но и бытовой, повседневной, предполагает предварительное моделирование и планирование этой деятельности, накопление и анализ информации, необходимой для её осуществления, т.е. составление алгоритма действий.

Способность человека к составлению таких алгоритмов составляет основу алгоритмического способа мышления.

Алгоритмический способ мышления позволяет принимать оптимальные решения в любой сфере человеческой деятельности и сам по себе никак не связан с программированием и вычислительной техникой. В таком неявном виде он существовал всегда, то есть изначально присущ человеческому мышлению. Появление вычислительной техники и профессии программиста привело лишь к тому, что необходимость алгоритмического способа мышления стала явной, по крайней мере, для определенного круга специалистов. Для решения программистских задач алгоритмический способ является единственно возможным. В программистской практике общий алгоритмический подход углубляется и детализируется: структура предметной области становится формализованной информационной структурой, в ней вычленяются количественные взаимосвязи, образующие математическую модель предметной области, алгоритм превращается в компьютерную программу [17; 25; 29]. Т.е. у людей, профессионально занимающихся программированием, алгоритмическое мышление должно быть развито уже не на житейском, повседневном уровне, а на более

высоком, профессиональном. А.П. Ершов в своей статье «О человеческом и эстетическом факторах программирования» писал: «Программист обязан обладать способностью первоклассного математика к абстракции и логическому мышлению в сочетании с эдисоновским талантом сооружать все, что угодно из нуля и единицы, он должен соединять в себе аккуратность бухгалтера с проницательностью разведчика, фантазию автора детективных романов с трезвой практичностью бизнесмена, а кроме того, иметь вкус к коллективному труду, быть лояльным к организатору работ и так далее.

Программист – солдат второй промышленной революции и как таковой должен обладать революционным мышлением и мужеством».

Образ мышления этих специалистов, который стал актуальным именно в процессе становления информационного общества, напервых порах был назван программистским. Термин «программистский стиль мышления» (а этот стиль эмпирически наблюдался психологами, которые исследовали поведение людей, связанных с вычислительными машинами) отражает значительную роль программистов в формулировке и решении важнейшей социальной задачи – формировании нового поколения людей, способных активно жить в условиях нового информационного общества.

Существует явное заблуждение, связанное с мыслью о том, что высокие математическая подготовка и логическое мышление человека являются необходимыми и достаточными условиями для легкого приобретения им компетенций программиста. Математические способы решения задач в виде математических алгоритмов не соответствуют подобным алгоритмам для решения задач на компьютере.

В этой связи возникает проблема обучения студентов с разными уровнями математического мышления. Особую актуальность приобретает эта проблема для подготовки будущих учителей информатики, поскольку им предстоит владеть методами формирования программистского стиля мышления у школьников.

В ФГОС ВО по направлению подготовки «Педагогическое образование» говорится, что бакалавр должен владеть современными средствами обработки информации, ориентироваться в программном обеспечении, уметь использовать современные технологии в профессиональной деятельности. Программирование является существенной составляющей предметной подготовки бакалавров «Педагогического образования». В своей профессиональной деятельности бакалавр как будущий учитель информатики должен уметь использовать современные технологии программирования для разработки приложений образовательного назначения и внедрять их в практику работы учебного заведения для обеспечения качества учебного процесса. А это значит, что у бакалавра, будущего учителя информатики должен быть сформирован программистский стиль мышления, как один из аспектов алгоритмического мышления.

Проблемам обучения курсу программирования школьников и студентов посвящено много исследований. Например, Баженовой И.В., Бобковой В.В., Газейкиной А.И., Ершовым А.П., Жужжаловым В.Е., Звенигородским Г.А., Калитиной В.В., Кузнецовым А.А., Могилевым А.В., Паком Н.И., Хеннером Е.К. и другие.

Однако в них не уделено должного внимания особенностям программистского стиля мышления. При этом отсутствуют подходы, определяющие методические аспекты подготовки будущего учителя информатики, учитывающие особенности перехода от житейского, повседневного алгоритмического мышления имеющегося у каждого человека к абстракциям, которые в основном нацелены на учащихся с визуальным, аудиальным и дигитальным восприятием, поскольку обращаются именно к этим каналам восприятия. При этом органы чувств, связанные с кинестетическими каналами восприятия, остаются незадействованы. Кинестетику, чтобы понять окончательно, необходима деятельность, поделать что-нибудь самому, руками, ощутить на ощупь.

Анализ литературных источников, опытов работ педагогов, исследователей по проблемам обучения программирования показал, что в качестве средств и приемов, позволяющих повысить эффективность обучения программированию, облегчить восприятие достаточно абстрактной учебной информации по этому предмету преподаватели используют практико-ориентированные задачи, постановка которых была бы понятна учащимся, написание программного кода сопровождается подробными комментариями, а также используются компьютерные презентации, обеспечивающие различную степень визуализации учебного материала. Существуют обучающие видеоролики, иллюстрирующие выполнения различных алгоритмов, алгоритмов сортировки, например. Все более широко начинают использоваться ментальные карты и схемы [6; 30].

Но все эти средства обучения нацелены на учащихся с визуальным и аудиальным восприятием, поскольку обращаются именно к этим каналам восприятия. Поэтому необходимы принципиально новые средства обучения программированию, которые были бы нацелены на кинестетические каналы восприятия и активизацию моторной области памяти. И это будет актуально не только для кинестетиков, поскольку активизация всех каналов восприятия приизложении учебного материала позволит существенно повысить его уровень понимания.

Согласно положениям, такого относительно нового течения в психологии как телесный подход, эти наши ощущения играют немаловажную роль в формировании мышления вообще, и, если мы говорим о возможности «подержать в руках», «осязать» процесс алгоритмической деятельности, программистского мышления в частности.

В современных стандартах предъявляются повышенные требования к уровню предметной подготовки учителей, поэтому в курс «Языки и методы программирования» включено изучение всех современных парадигм и технологий программирования – императивной, декларативных (логическое

и функциональное программирование), объектно-ориентированное, параллельное, сценарное программирование.

Изучение языка программирования относящейся к другой парадигме, вызывает определенный ряд сложностей. Так как при переходе к программированию методами, которые относятся к другой парадигме, необходимо изменить не только подход к решению поставленной задачи, но и перестроить мыслительную деятельность относительно новой парадигмы. Каждая парадигма программирования предполагает формирование определенного стиля мышления.

Остается открытым вопрос о методических условиях формирования необходимых стилей мышления. Практически во всех исследованиях [11; 25] предлагается изучать различные парадигмы последовательно, и исследователи спорят лишь об оптимальной последовательности их изучения. На наш взгляд, при проектировании методической системы обучения курсу «Языки и методы программирования» целесообразно предусмотреть параллельное изучение парадигм программирования. Подавляющее большинство исследователей осознают, что сформированное на достаточном уровне алгоритмическое мышление создает сложности при переходе на логическое программирование, объектно-ориентированное и функциональное и наоборот. При изучении парадигм программирования параллельно это проблема значительно ослабляется.

Таким образом, возникают **противоречия**:

- между *необходимостью* выделить в качестве отдельной компоненты в структуре алгоритмического мышления программистский стиль мышления студента и *отсутствием* адекватных моделей, позволяющих критериально оценивать уровень сформированности этого стиля мышления;

- между *возможностями* натурных и телесных алгоритмических средств обучения и *не разработанностью* методической базы их

целесообразного и эффективного использования в процессе обучения студентов программированию с учетом их когнитивных особенностей;

- между *необходимостью* будущему учителю информатики в ограниченное время освоить множество парадигм программирования на основе своих личностных предпочтений и *отсутствием* подходящих методик полиязыкового обучения студентов на занятиях по программированию.

Противоречия определяют **проблему исследования:** На какой методологической платформе и какими методами и средствами обучения на занятиях по программированию можно обеспечить повышение уровня программистского стиля мышления студентов с учетом их когнитивных особенностей

Объект исследования: процесс обучения студентов педагогического вуза курсу «Языки и методы программирования».

Предмет исследования: методика формирования программистского стиля мышления будущих учителей информатики в процессе обучения курсу «Языки и методы программирования» на основе телесно-полиязыкового подхода.

Цель исследования: теоретическое обоснование и разработка методики обучения студентов курсу «Языки и методы программирования» на основе телесно-полиязыкового подхода.

Гипотеза исследования представляет собой предположение о том, что повышение программистского стиля мышления студента в процессе обучения курсу «Языки и методы программирования» может быть обеспечено, если:

1) для оценки уровня программистского стиля мышления студента использовать структурную модель алгоритмического стиля мышления;

2) для формирования у студентов с низким уровнем алгоритмического мышления представлений о базовых алгоритмических конструкциях применять средства и методы телесного подхода;

3) для переориентации студентов с высоким уровнем алгоритмического мышления на машинно-зависимый программистский стиль мышления использовать натурные средства алгоритмических конструкций;

4) для повышения эффективности формирования программистского стиля мышления студентов использовать полиязыковой подход к обучению различным парадигмам программирования.

Цель, предмет и гипотеза исследования определили **задачи исследования:**

1. Проанализировать состояние разработанности проблемы исследования в психолого-педагогической, научно-методической науке и практике.

2. Уточнить содержание понятия «программистский стиль мышления», построить структурную модель программистского стиля мышления и определить этапы его формирования, выделить уровни его сформированности и критерии их измерения.

3. Выявить возможности использования телесного подхода и разработать концепцию полиязыкового подхода к обучению студентов программированию.

4. Разработать структурно-логическую модель обучения студентов курсу «Языки и методы программирования» с применением средств (кинестетические тренажеры) и методов (параллельное изучение языков программирования) телесного и полиязыкового подходов.

5. Разработать комплекс натурных, кинестетических и полиязыковых средств обучения.

Для решения поставленных задач использовались следующие **методы исследования:**

теоретические (изучение и анализ педагогической, психологической, методической и предметной литературы по теме исследования, анализ теоретических и эмпирических данных, изучение и обобщение педагогического опыта, сравнительный анализ, классификация);

эмпирические (наблюдение, анкетирование, беседа, тестирование, педагогический эксперимент).

Научная новизна исследования заключается в том, что:

1. Определена категорийно-понятийная компонента «программистский стиль мышления» в структуре алгоритмического мышления, позволяющая оценивать его уровень сформированности в процессе обучения курсу «Языки и методы программирования».

2. Обоснована необходимость и возможность использования телесного и полиязыкового подходов к обучению студентов в процессе предметной подготовки, обеспечивающие формирование их программистского стиля мышления.

3. Разработана методика телесно-полиязыкового обучения студентов программированию, способствующая формированию у них высокого уровня программистского стиля мышления и обеспечивающая их методическую готовность осуществлять подготовку школьников в области программирования за счет натурных, кинестетических и полиязыковых средств и методов обучения программированию.

Теоретическая значимость исследования заключается:

1) в уточнении понятия «программистский стиль мышления», определении этапов, уровней его сформированности и разработки критериев их оценки;

2) в построении структурно-логической модели обучения студентов курсу «Языки и методы программирования» на основе телесного и полиязыкового подходов.

Практическая значимость исследования заключается в том, что:

- разработан комплекс алгоритмических натурных, кинестетических и полиязыковых средств обучения студентов программированию;

- опубликован в соавторстве с группой авторов (Нигматулина Э.А., Пак Н.И., Сокольская М.А., Степанова Т.А.) учебник «Программирование» в 2-х томах для бакалавров педагогического образования профиль «информатика» (М.: Академия, гриф УМО по направлению «Педагогическое образования»), отражающий концепцию полиязыкового обучения программированию.

- разработанная методика телесно-полиязыкового обучения студентов программированию может быть использована в учебном процессе в педагогических вузах, на курсах повышения квалификации.

Экспериментальная база и этапы исследования. Опытно-экспериментальная работа по теме исследования осуществлялась на базе отделения информатики Института математики, физики, информатики Красноярского государственного педагогического университета им. В.П. Астафьева, на базе физико-математического факультета Лесосибирского педагогического института – филиала СФУ. В педагогическом эксперименте в разное время в общей сложности принимали участие 300 студентов третьего курса, изучавших дисциплину «Языки и методы программирования».

Исследование проводилось с 2009 по настоящее время и состоит из трех этапов.

Первый этап (2009 – 2011 гг.) – изучение предметной области исследования, анализ проблематики исследования, уточнение его методологического аппарата, выделение целей, содержания, методов и средств обучения программированию, теоретическое построение методики, констатирующий эксперимент.

Второй этап (2012 – 2015 гг.) – уточнение и корректировка содержания дисциплины «Языки и методы программирования», построение

модели методической системы, построение информационно-деятельностной модели программистского стиля мышления и выделение этапов его формирования, внедрение методики телесно-полиязыкового обучения программированию в учебный процесс, проведение формирующего эксперимента.

Третий этап (с 2016 г.) – завершение формирующего эксперимента, количественный и качественный анализ его результатов, систематизация и обобщение итогов исследования, окончательная доработка методических пособий для заявленного курса.

Достоверность и обоснованность полученных результатов исследования обеспечиваются научной обоснованностью исходных теоретических положений, соответствием применяемых в исследовании методов цели и задачам исследования.

На защиту выносятся следующие **положения:**

1. Структурная модель алгоритмического мышления студента с уточнением программистского стиля мышления позволяет оценивать уровень его сформированности.

2. Использование натурных и кинестетических полиязыковых средств и методов обучения студентов способствуют формированию их программистского стиля мышления в процессе предметной подготовки с учетом их когнитивных особенностей.

3. Методика телесно-полиязыкового обучения студентов программированию обеспечивает у будущих учителей информатики их методическую готовность осуществлять подготовку школьников в области программирования.

Апробация и внедрение результатов. Материалы исследования обсуждались на заседаниях кафедры информатики и вычислительной техники КГПУ им. В.П. Астафьева, на научно-исследовательских семинарах-вебинарах «Информационные технологии и открытое образование», были

представлены на V Всероссийской научно-практической конференции с международным участием «Открытое образование: опыт, проблемы, перспективы». (Красноярск, 2009 г); на семинаре «Информатика образования» в рамках Ершовской конференции по информатике (PSI'11, Новосибирск, 2011 г.), на VIII Международной научно-практической конференции «Педагогический профессионализм в образовании» (Новосибирск, 2012 г.), на Всероссийской научно-методической конференции «Интегрированная система профессионального образования: проблемы и пути развития» (Красноярск, 2012 г.), на I Всероссийской научно-практической конференции «Всероссийский форум педагогического мастерства» (Москва, 2013 г.), на IV Всероссийской научно-практической конференции с международным участием «Перспективы и вызовы информационного общества» (Красноярск, 2015 г.), на XII International Scientific And Practical Conference «MODERN SCIENTIFIC POTENTIAL – 2016» (г. Шеффилд, 2016 г.), на VIII Международной научно-практической конференции «Актуальные направления фундаментальных и прикладных исследований» (North Charleston, USA, 2016 г.), на I Международной научной конференции в рамках IV Международного научно-образовательного форума «Человек, семья и общество: история и перспективы развития» «Информатизация образования и методика электронного обучения» (Красноярск, 2016 г.), на XII Международна научна практична конференция «Бъдещите изследвания - 2016» (София, 2016 г.), на X Международной студенческой научно-практической конференции «Студенческая наука XXI века» (Чебоксары, 2016), на IX Международной научно-практической конференции «Инновационные технологии в науке и образовании» (Чебоксары, 2017 г.), на XII Международной студенческой научно-практической конференции «Студенческая наука XXI века» (Чебоксары, 2017 г.)

По теме исследования опубликовано 25 работ, в том числе 3 статьи в изданиях, рекомендованных ВАК, учебник по программированию в двух томах (2013 г.), 1 учебное пособие по основам программирования на языке Турбо Паскаль (2006 г.), 1 учебное пособие по программированию на Паскале (2015 г.).

Результаты исследования используются при организации обучения студентов отделения информатики Института математики, физики, информатики Красноярского государственного педагогического университета им. В.П. Астафьева курсам «Профильное исследование», «Языки и методы программирования», на базе физико-математического факультета Лесосибирского педагогического института – филиала СФУ и частично – в бюджетном общеобразовательном учреждении «Кириковская средняя школа».

Глава 1. Теоретические основы телесно-полиязыкового подхода к обучению программированию будущих учителей информатики

1.1 Уточнение содержания понятия «программистский стиль мышления»

Современное информационное, наукоёмкое общество нуждается в людях с высоким уровнем развития абстрактного мышления вообще и алгоритмического мышления в частности, поскольку осуществление абсолютно любой деятельности – не только профессиональной, где это является необходимым условием в настоящее время практически во всех профессиях, но и бытовой, повседневной, предполагает предварительное моделирование и планирование этой деятельности, накопление и анализ информации, необходимой для её осуществления, т.е. составление алгоритма действий.

Способность человека к составлению таких алгоритмов составляет основу алгоритмического способа мышления.

Алгоритмический способ мышления позволяет принимать оптимальные решения в любой сфере человеческой деятельности, и сам по себе никак не связан с программированием и вычислительной техникой. В таком неявном виде он существовал всегда, то есть изначально присущ человеческому мышлению. Появление вычислительной техники и профессии программиста привело лишь к тому, что необходимость алгоритмического способа мышления стала явной по крайней мере для определённого круга специалистов. Для решения программистских задач алгоритмический способ является единственно возможным. В практике общий алгоритмический подход углубляется и детализируется: структура предметной области становится формализованной информационной структурой, в ней вычленяются количественные взаимосвязи, образующие математическую модель предметной области, алгоритм превращается в компьютерную

программу [17; 25; 29]. Т.е. у людей, профессионально занимающихся программированием, алгоритмическое мышление должно быть развито уже не на житейском, повседневном уровне, а на более высоком, профессиональном.

В современной методической литературе, посвященной обучению информатики, сложно найти материалы, в которых не упоминается об алгоритмическом стиле мышления. Но еще сложнее найти публикации, в которых определяется это понятие. В лучшем случая этот термин объясняется на эмпирическом уровне.

В известной статье Дональда Кнута «Алгоритмическое мышление и математическое мышления» [37] проводится сравнительный анализ этих стилей мышления.

Для получения ответа на вопрос, что такое алгоритмическое мышление и чем отличается оно от математического, Д. Кнут поставил интересный эксперимент. Он по разным критериям оценивал девять случайно выбранных математических примеров. Вот его предварительные выводы:

- не существует «математического мышления» как отдельного, изолированного понятия: математики используют множество различных способов мышления;

- программисты и математики очень похожи в том смысле, что у тех и других выделяются несколько общих типов мышления, хотя есть типы, характерные только для математиков или программистов;

- математик от программиста отличается тем, что у математика почти нет понятия «*сложности»* или «*экономичности»* действия или не хватает понятий, связанных с «*операцией присваивания»*, которые меняют количественные данные [37].

В работах М.В.Беляева понятие алгоритмического мышления вводится на основе дихотомии, разделения, способов мышления ассоциативное – алгоритмическое [91].

Вот такое определение дается Копаевым А.В. [40]: «Алгоритмический стиль мышления – это система мыслительных способов действий, приемов, методов и соответствующих им мыслительных стратегий, которые направлены на решение как теоретических так и практических задач, и результатом которых являются алгоритмы как специфические продукты человеческой деятельности».

Недостаток существующих определений в том, что они не сводят определения мышления к основным, неделимым понятиям, определяя понятие «мышление» через понятие «мыслительные процессы», которое в свою очередь требует определения.

Главной стержневой проблемой психологии и дидактики, связанной с познанием и обучением, является отсутствие разумной модели мышления, многообразие интерпретаций сущности алгоритмического мышления [90].

В психологии исследованию мышления человека посвящено множество исследований и теорий. Кратко охарактеризуем некоторые из них.

Бихевиоризм. Приверженцы этого учения считают, что познание является исключительно набором отношений «стимул – реакция». Мышление, при таком подходе, состоит их набора конкретных действий, зависящих от окружающей среды и получаемых из неё стимулов, но осуществляется там, где внешний наблюдатель не может их видеть. Исследователи в этом случае ограничиваются анализом ассоциаций в отношениях «стимул – реакция».

Гештальтпсихология. Исследователи этой области считают, что любое переживание целостно, связно и не может быть понято при помощи простого разделения на составные части. Эта теория также утверждает, что прошлый опыт не влияет на наши текущие переживания. Ещё один постулат этой теории гласит, что человеческая психика, как и любая динамическая система, стремится к максимальному в наличных условиях состоянию стабильности. Из-за ограниченного исследования процессов мышления

гештальтпсихология почти неприменима для построения психологических основ обучения.

Когнитивная психология. Основана на трактовке процесса мышления человека с позиции преобразования информации в вычислительном устройстве. Так были выделены многочисленные структурные составляющие (блоки) познавательных и исполнительных процессов, прежде всего памяти. [15].

Для теоретического обоснования понятия «программистский стиль мышления» будем использовать пространственно-временную модель памяти [77] и психологическую теорию деятельности [10; 13; 49; 50; 96].

В пространственно-временной модели память человека в каждый момент времени может быть разделена на 4 области (рис.1):

- *чувственную область*, в которой фиксируются в виде чувственных образов отражения объектов внешнего мира, воспринятые при помощи сенсорной системы;
- *модельную область*, в которой хранятся образы модельного отражения окружающего мира;
- *понятийную область*, в которой хранятся образы понятий в аудиально-знаковой форме, связанные со свойствами воспринятых объектов и событий [77];
- *абстрактную область*, в которой хранятся образы обобщенных понятий (абстракции).

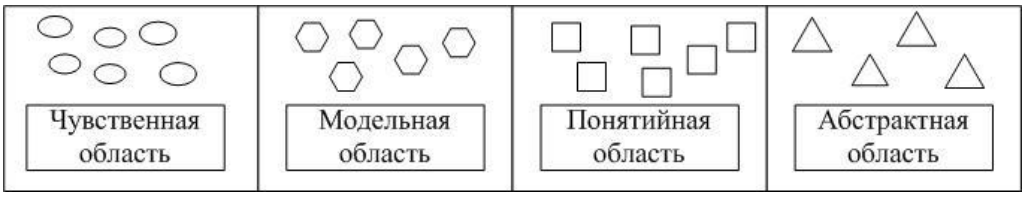


Рис. 1. Пространственно-временная модель структуры памяти

Извлечение информации из памяти происходит путём активации соответствующих образов в рассмотренных областях. В этой связи автор модели предлагает принять следующие определения базовых понятий:

мысль – это активированный образ в памяти;

*мыслительный процесс*или *мышление*– это способ активации последовательной цепочки образов (мыслей).

Цепочки активных образов могут выстраиваться различными способами. Мыслительный процесс может быть чисто чувственным (активная цепочка из образов чувственной области памяти), модельным или абстрактным, либо смешанным – активируется последовательность образов из разных областей памяти (рис. 2).

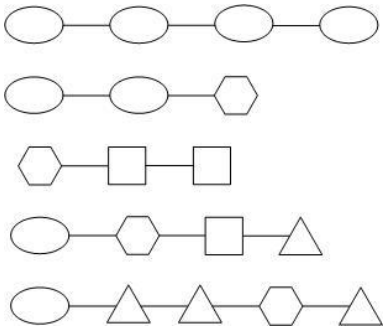


Рис. 2. Последовательный способ мышления

Объекты реального мира, воспринимаемые одновременно, образуют в соответствующей области памяти образы, которые также формируются или активизируются одновременно. При этом если объекты реального мира вовлечены в параллельно протекающие несвязанные процессы, то цепочки, порождаемые такими объектами, также возникают параллельно и могут иметь разные конфигурации (рис. 3). С переходом от чувственной и модельной области к понятийной и абстрактной, процесс формирования сложных, ветвящихся цепочек, по сути, не изменяется.

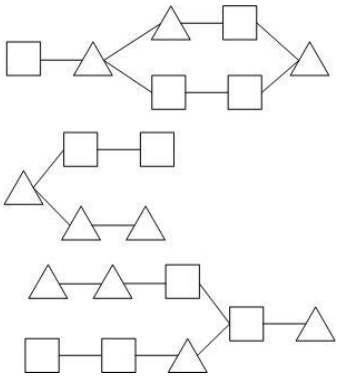


Рис. 3. Параллельный способ мышления

Согласно положениям психологической теории деятельности А.Н. Леонтьева, С.Л. Рубинштейна формирование рассмотренных на рис. 2 цепочек наиболее эффективно осуществляется в процессе соответствующей деятельности. Рассмотрим положения этой теории.

Специальные психические процессы, в том числе и мышление, не могут рассматриваться вне системы отношений деятельность-сознание, их общих системных качеств, поскольку «существуют только в описанных отношениях системы, на тех или иных ее уровнях» [49].

Связи субъекта с материальным миром осуществляются через деятельность, который получает отражение в сознание именно благодаря соответствующей деятельности, поскольку «развитие деятельности необходимо приводит к возникновению психического отражения реальности…» [49].

«Всякий мыслительный процесс является по своему внутреннему строению действием или актом деятельности, направленным на разрешение определенной задачи» [97].

Очень важную роль для мыслительного процесса представляют «специфические «навыки» мышления», возникающие в процессе целенаправленного решения определённых задач [97].

Также Рубинштейн писал о том, что осознание действия в ходе деятельности происходит тогда, когда достигаемый с помощью действия частичный результат является прямой целью действия. Когда цель переносится дальше и прежнее действие превращается лишь в «способ осуществления другого действия», прежнее перестаёт осознаваться. Действия, вбираемые в более обширные общие задачи, «выключаются из сознания, переходят в подсознательное» [97].

В этой связи П.Я. Гальперин замечает, что «отдельная мысль – как явление психологическое – представляет собой не что иное, как предметное

действие, перенесённое во внутренний, умственный план, а затем ушедшее во внутреннюю речь. *Здесь оно уже недоступно самонаблюдению*…» [13].

Если применить рассмотренные положения к процессу обучения, то для формирования определённых «навыков», способов мышления необходимо ставить обучающихся в условия, предполагающие постепенное формирование действий, необходимых для решения комплекса изучаемых теоретических или практических задач, и дальнейшего постепенного перевода этих действий на подсознательный план через постановку более общих целей. Мысленное действие, перешедшее в подсознание, также выполняется, но уже в неосознанной, свёрнутой, сокращённой форме. Путём самонаблюдения свёрнутое действие крайне сложно снова перенести на внешний план. Именно этот процесс проходит преподаватель, выстраивая ход решения учебных задач.

Поскольку, согласно модели памяти, мышление – это способ активации цепочек образов, то при переходе действия на подсознательный план некоторые звенья цепочек «сворачиваются» и «выпадают» из процесса решения поставленной перед субъектом задачи. Для формирования определённого стиля мышления весь процесс выстраивания образных цепочек необходимо тщательно восстанавливать.

Таким образом, для того, чтобы студенты могли выстраивать мысленные алгоритмические цепочки различного характера (рис 3), их нужно вовлекать в целенаправленную деятельность, связанную с выполнением соответствующих алгоритмических упражнений.

Формирование алгоритмического мышления начинается в возрасте 5-6 лет и продолжается формироваться в школе, при изучении школьного курса математики, информатики и других естественно-научных и точных дисциплин, но в значительной степени формирование и развитие алгоритмического мышления происходит все-таки на уроках информатики, и в связи с этим на плечи школьного учителя информатики ложится большая

ответственность. Для того чтобы сформировать алгоритмическое мышление у своих учеников на повседневном уровне, у самого учителя оно должно быть развито на более высоком, профессиональном уровне [70]. Кроме того, современные подходы к совершенствованию системы среднего образования предъявляют высокие требования к уровню предметной подготовки учителя. Несомненно, его научный кругозор должен значительно превышать рамки школьной программы. Следовательно, несмотря на то, что в школьном курсе программирования изучаются только алгоритмические языки, относящиеся к императивной парадигме, учитель информатики должен владеть всеми современными парадигмами и технологиями программирования.

По этой причине курсы программирования, преподаваемые в педагогических университетах, предполагают изучение языков программирования, относящихся к различным парадигмам программирования, сложившимся в современной науке – не только императивной, но и декларативных – объектно-ориентированной, функциональной, логической. Современные стандарты и современный уровень развития суперкомпьютерных технологий, основанных на параллельных вычислениях, повсеместное использование многоядерных процессоров предполагает также изучение параллельного программирования.

Изучение языка программирования относящегося к другой парадигме, вызывает определённый ряд сложностей. Так как при переходе к программированию методами, которые относятся к другой парадигме, необходимо изменить не только подход к решению поставленной задачи, но и перестроить мыслительную деятельность относительно новой парадигмы. Каждая парадигма программирования предполагает формирование определённого стиля мышления – императивного, объектного, функционального, логического, параллельного [11;12; 25; 26].

Некоторые психологи относят к разряду проблем влияние механических мыслительных процессов компьютера на способы мышления людей. Тем не менее, при осуществлении учебно-познавательной деятельности обучаемым часто предлагаются алгоритмы решения задач определенных классов или алгоритм (последовательность операций, или шагов) выполнения некоторого задания. Предполагается, что обучаемый должен уметь его исполнить. Трудности, возникающие при этом, большей частью связаны с неверной интерпретацией исходных данных и отсутствием умения формального исполнения алгоритма. Формируемый при этом стиль мышления ряд исследователей называют *императивным*. Таким образом, императивный стиль мышления предполагает умение обучаемого действовать по заданному алгоритму, умение исполнить его.

Достаточно широко в научной и методической литературе, посвященной проблемам обучения информатике, используется понятие *алгоритмический стиль мышления*. Алгоритмический стиль мышления представляет собой специфический стиль мышления, предполагающий умение создать алгоритм, для чего необходимо наличие мыслительных схем, которые способствуют видению проблемы в целом, ее решению крупными блоками с последующей детализацией и осознанным закреплением процесса получения конечного результата в языковых формах.

Алгоритмическое мышление составляет важную часть интеллектуальной деятельности человека с применением современных информационных технологий. Следует отметить, что понятие «алгоритмический стиль мышления» сложилось в тот период времени, когда преобладала парадигма структурного программирования. Оно базируется на применении алгоритмической декомпозиции при решении задач.

Поскольку в школьном курсе информатики изучаются языки программирования, относящиеся к императивной парадигме, можно сказать, что у студентов в начале обучения сформировано алгоритмическое

мышление в узком смысле, в том понимании, которое сложилось в тот период времени, когда преобладала парадигма императивного, структурного программирования.

Произошедший переход к объектно-ориентированной парадигме создания и использования средств информационных технологий не отрицает необходимости формирования алгоритмического стиля мышления, но расширяет это понятие.

На современном этапе развития информатики для успешного взаимодействия с компьютером необходим стиль мышления, который можно назвать *объектным*. Он предполагает умение разделить сложную систему на объекты и выстроить их иерархию, т.е. произвести объектную декомпозицию системы, а затем описать поведение этих объектов. Основной операцией при таком стиле является объектно-ориентированная декомпозиция, разложение объектов. Всевозможные классификации по различным логическим основаниям и логические методы формирования понятий составляют значительную часть методов, используемых при таком стиле мышления. При описании событий используется алгоритмическая декомпозиция системы и необходим алгоритмический стиль мышления.

Выбор способов обработки информации определяется спецификой предметной области решаемой задачи. Использование различных подходов к программированию существенно влияет на эффективность процесса создания компьютерных программ. Так, например, процесс разработки высокоинтеллектуальной экспертной программной системы существенно упрощается при использовании логической парадигмы. В связи с этим выделяется *логический стиль мышления*. В основе логических языков программирования лежат формализованная логика, т.е. основной принцип состоит в том, что нужно подробно, на логически точном языке, описать условие задачи. Решение её получается в результате определённого процесса поиска решения в соответствии с правилами, который исполняется

компьютером. В этом заключается разница между логическими языками программирования и традиционными, которые требуют описание процедуры решения задачи.

Изучение логической парадигмы обработки информации даёт новое понимание при изучении программирования, способствуя тем самым развитию у студентов иного стиля мышления, предполагающего отказ от императивных стереотипов.

Идея функционального программирования опирается на интуитивное понятие о функциях как о достаточно общем механизме представления и анализа решений сложных задач. Механизм функций основательно изучен математиками, и это позволяет программистам наследовать выверенные построения, обладающие предельно высокой моделирующей силой.

Функциональный стиль объединяет разные подходы к определению процессов вычисления на основе достаточно строгих абстрактных понятий и методов символьной обработки данных. Связь функционального программирования с математикой позволяет в тексте программы наследовать доказательность построения результата, если она достигнута, причём с использованием разных методов абстрагирования решаемой задачи.

Сложность решения задач с помощью функциональных определений преодолевается чисто алгебраически: нацеленностью на формализацию основного множества объектов и определения полной семантической системы операций над ними. Это позволяет представлять классы задач и их решений строгими формулами, для наглядности упрощаемыми введением дополнительных функциональных символов. При необходимости такие символы вносятся в определение алгебраической системы, что приводит к ее расширению. Вводятся новые функции, подобные леммам и другим вспомогательным построениям в математике. Активно используется рекурсия и символьные обозначения как данных, так и действий и любых формул, удобных при определении функций [11; 12; 63; 70].

Технология параллельного программирования не выделяется в отдельную парадигму, т.к. она может быть реализована в каждой из перечисленных ранее. В отличие от программирования последовательных вычислений, концептуальную основу которого составляет понятие алгоритма, реализуемого по шагам строго последовательно во времени, в параллельном программировании программа порождает совокупность параллельно протекающих процессов обработки информации, полностью независимых или связанных между собой статическими или динамическими пространственно-временными, причинно-следственными или информационными отношениями.

Вычислительный параллелизм выступает в разных конкретных формах в зависимости от этапа программирования, от сложности параллельных фрагментов и характера связей между ними.

Параллельное программирование представляется достаточно сложным для изучения, но его легче понять, если считать не «трудным», а просто «немного иным». Оно включает в себя все черты более традиционного, последовательного программирования, но в параллельном программировании имеются три дополнительных, чётко определённых этапа.

В процессе изучения параллельной технологии программирования формируется *параллельный стиль мышления*, предполагающий способность к предварительному умозрительному «распараллеливанию» поставленной задачи – ее анализу с целью выделения подзадач, которые могут выполняться параллельно, «распараллеливанию» потоков данных – выделению потоков данных, которыми будут обмениваться выполняемые параллельно подзадачи. Также такой стиль предполагает возможность удерживания в памяти действий всех подзадач в определённый промежуток времени для правильного управления их совместной работой. При достаточно высоком уровне развития параллельного стиля мышления программист способен предвидеть те проблемы, возникающие при работе

параллельных алгоритмов, которые не встречаются в работе последовательных, проявляют себя не каждый раз и существенно затрудняют отладку программ. Лишь достижение определённого уровня развитияпараллельного стиля мышления позволит программисту эффективно реализовывать параллельные алгоритмы [63].

Таким образом, вузовский курс программирования требует значительного расширения трактовки алгоритмического стиля мышления. Чтобы не путать с его традиционным пониманием, назовем его «*программистским стилем мышлением*».

Появление, распространение и совершенствование вычислительных машин – основных инструментов информационного общества – непосредственно отразилось на образе деятельности и мышлении тех людей, которые в силу своей профессии первыми осознали революционизирующую роль новых информационных технологий. Речь идет о программистах.

А.П. Ершов в своей статье «О человеческом и эстетическом факторах программирования» писал: «Программист обязан обладать способностью первоклассного математика к абстракции и логическому мышлению в сочетании с эдисоновским талантом сооружать все, что угодно из нуля и единицы, он должен соединять в себе аккуратность бухгалтера с проницательностью разведчика, фантазию автора детективных романов с трезвой практичностью бизнесмена, а кроме того, иметь вкус к коллективному труду, быть лояльным к организатору работ и так далее... Программист – солдат второй промышленной революции и как таковой должен обладать революционным мышлением и мужеством».

Образ мышления этих специалистов, который стал актуальным именно в процессе становления информационного общества был назван программистским. Термин «программистский стиль мышления» (а этот стиль эмпирически наблюдался психологами, которые исследовали поведение людей, связанных с вычислительными машинами) отражает значительную

роль программистов в формулировке и решении важнейшей социальной задачи – формировании нового поколения людей, способных активно жить в условиях нового информационного общества.

Существует явное заблуждение, связанное с мыслью о том, что высокие математическая подготовка и логическое мышление человека являются необходимыми и достаточными условиями для легкого приобретения им компетенций программиста. Математические способы решения задач в виде математических алгоритмов не соответствуют подобным алгоритмам для решения задач на компьютере.

Вот некоторые из умений и навыков, образующих программистский стиль мышления.

1) Умение планировать структуру действий, необходимых для достижения цели при помощи фиксированного набора средств.

Когда компьютерный пользователь описывает алгоритм решаемой задачи, он, представляя себе цель решения задачи конечный результат, конструирует программу (в широком смысле этого слова), то есть, план действий, являющийся последовательностью отдельных более или менее стандартных операций. Организуя структуру действий, пользователь должен спланировать не только действия как таковые, но и используемые в этих действиях информационно-технические ресурсы.

2) Умение строить информационные модели для описания объектов и систем.

Хотя современные информационные и программные системы предоставляют пользователям значительные удобства для описания данных, всегда очень важно представлять, к каким классам объектов относятся описываемые величины, каковы их взаимосвязи в решаемой задаче. Имея такие представления, пользователь сможет найти наиболее эффективные реализации решений. В построениях моделей важен навык формализованного описания объектов и связей.

Значение этого навыка постоянно растет в прикладных информационных системах (базах данных, электронных таблицах, редакторах), основу которых составляют информационные модели. В информационной модели отражаются все существенные для решения поставленной задачи свойства объектов в их взаимодействии.

3) Умение организовать поиск информации, нужной для решения поставленной задачи.

Решение задачи становится эффективным только тогда, когда правильно определен объем информации, необходимой для ее решения и правильно организован ее поиск. Навыки использования многообразных поисковых механизмов выходят за рамки собственно программирования. Огромные информационные фонды, уже сегодня доступные по глобальным информационным сетям, делают исключительно важным умение правильно определить, какие именно сведения необходимы и по каким признакам можно организовать их поиск.

4) Дисциплина и структурированность языковых средств коммуникации.

Эти важные качества человеческого мышления (и поведения) означают умение правильно, четко и однозначно сформулировать мысль в понятной собеседнику форме и правильно понять информационное сообщение.

Отсутствие такой дисциплины в общении людей часто компенсируется способностью человека к сопереживанию. Это позволяет понять недосказанную или нечетко выраженную мысль. Компьютер, вообще говоря, не обладает такой способностью, и любая неточность в формулировке задания влечет искажение смысла и ошибку.

Программист должен работать с компьютером, учитывая уровень его обученности: с системами низшего уровня (обладающими незначительным программным обеспечением) приходится общаться на уровне описаний «микродействий» машинных операций; с более развитыми системами

общение оказывается возможным с помощью укрупненных операций (функций и процедур); в системах, богато оснащенных программным обеспечением, программист может конструировать свою программу из модулей – готовых программ. Впрочем, и пользователю, не являющемуся профессионалом в программировании, для эффективного общения с компьютером важно уметь «запроцедурить» часто используемые конструкции, чтобы далее пользоваться ими как элементарными командами.

5) Навык своевременного обращения к компьютеру при решении задач из разных предметных областей.

Если этот навык не выработан (не доведен до уровня привычки), то даже человек, осознающий актуальность отмеченных выше умений и навыков, может не догадаться обратиться к компьютеру, если такая задача прямо не сформулирована. Часто можно наблюдать, как пользователь, сидящий у экрана современного компьютера, тянется к карандашу и листочку бумаги (в лучшем случае – к калькулятору), чтобы сделать тривиальные вычисления или промежуточные записи.

6) Технические навыки взаимодействия с компьютером, в частности, умение работать клавиатурой и мышью.

Формирование перечисленных навыков у всех тех, кто соприкасается с вычислительной техникой, то есть, практически у большинства людей Земли, представляется необходимым для обеспечения эффективного использования ресурсов нового информационного общества.

Роль перечисленных выше умений и навыков оказывается намного значительнее «технологических» знаний, которые позволяют поднять производительность компьютерной техники и эффективность ее использования (при всей экономической важности задачи). В философском, социальном и педагогическом плане каждый из них имеет самостоятельное (и очень важное) значение в системе навыков умственных действий, необходимых каждому современному образованному человеку.

Так, *умение планировать структуру целенаправленных действий* необходимо в любом научном исследовании, в любом производстве, в армии, общественной жизни коллектива, в быту. Особенно важно умение планировать свою деятельность для педагога: план представляет собой определяющий документ в деятельности школьного учителя.

Умение строить информационные модели это лишь частный случай умения правильно строить модели вообще. Это умение необходимо в любом научном исследовании, в любой конструкторской или технологической разработке, когда созданию нового объекта (быть может, очень дорогого или опасного) должен предшествовать этап моделирования.

У*мение организовать поиск информации* необходимо в любой научной, творческой, технической работе, независимо от того, где и как хранится информация: в архиве, в библиотеке, в патентной картотеке, в Интернете или (в частном случае) в памяти компьютера.

Дисциплина общения людей ничуть не менее важна, чем межмашинные или человеко-машинные коммуникации. Отсутствие такого качества существенно затрудняет диалог людей.

Умение инструментировать свою деятельность, то есть, находить в каждой ситуации адекватные средства для решения поставленной задачи, важно вне зависимости от того, какие инструменты находятся в распоряжении человека - счеты, калькулятор, записная книжка. На примере компьютеров необходимость такого качества становится еще более наглядной [86].

На рисунке 4 представлена структурная модель программистского стиля мышления, в состав которой входят императивное мышление, объектно-ориентированное мышление, логическое мышление, функциональное мышление и параллельное мышление.

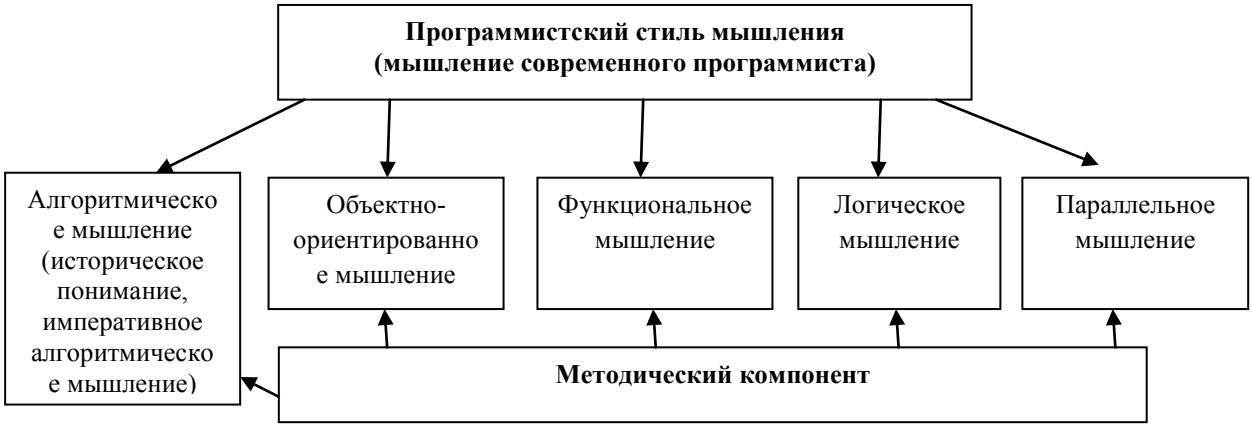


Рис. 4. Структурная модель программистского стиля мышления

Говоря о программистском стиле мышления будущих учителей информатики, это понятие должно быть, расширено методическим компонентом, поскольку у педагогов не только у самих должно быть сформировано программистское мышление на самом высоком уровне, но они еще должны быть способными формировать и развивать программистское мышление своих учеников.

Обучение программированию в вузе должно быть направлено на формирование у студентов перечисленных выше стилей мышления, без чего это обучение не будет являться эффективным.

Согласно предложенной структурной модели программистского стиля мышления можно выделить следующие этапы его формирования:

1) Пропедевтический. Этап может быть реализован в процессе изучения материалов дисциплин, предшествующих изучению курса «Языки и методы программирования», а именно при изучении вводного курса информатики.

2) Основной. Этап формирования программистского стиля мышления, реализуемый в процессе обучения студентов непосредственно в курсе «Языки и методы программирования».

3) Закрепляющий. Этап реализуется через применение знаний, полученных в результате изучения курса «Языки и методы программирования» в процессе обучения на старших курсах, в дисциплинах,

изучаемых в дальнейшем, а именно на дисциплинах «Численные методы», «Компьютерное моделирование», «Исследование операций» и др.

Построенная структурная модель программистского мышления (рис.4) позволяет разработать систему диагностики уровня сформированности программистского стиля мышления.

Оценку уровня сформированности программистского стиля мышления студента следует проводить в рамках интегральной порядковой шкалы по совокупности указанных в модели компонентов.

Для создания системы диагностики уровня сформированности программистского стиля мышления необходимо:

1) Разработать критерии, по которым можно будет проводить оценку уровня сформированности программистского стиля мышления. Выделить уровни сформированности программистского стиля мышления;

2) Разработать разноуровневую систему практико-ориентированных заданий.

3) Предложить механизм расчетаколичественных оценок в баллах за выполнение этих заданий.

4) Предложить методические рекомендации по проведению диагностики.

Для решения первой задачи необходимо:

1) Разработать уровни и критерии сформированности программистского стиля мышленияобучаемых;

2) Дать характеристику уровням и критериям;

3) Разработать в соответствии с этим методику диагностики уровня сформированности программистского стиля мышления.

Решение второй задачи требует:

1) Определить структуру деятельности учащихся на занятиях;

2) Разработать систему практико-ориентированных заданий и кинестетических тренажеров для корректировки отдельных умений, в

соответствии с выделенной структурой и уровнями сформированности программистского стиля мышления;

Для решения третей задачи необходимо:

1) На основе выделенной структуры деятельности учащихся при формировании у них программистского стиля мышления разработать методику, позволяющую преподавателю создать на занятиях условия для осуществления это деятельности;

2) Предоставить преподавателю необходимую методическую базу для диагностики уровня сформированности программистского стиля мышления учащихся и коррекции у них отдельных компонентовпрограммистского стиля мышления.

В качестве критериев, по которым можно проводить оценку уровня сформированности программистского стиля мышления нами выбраны следующие:

Операционный критерий – показывает насколько обучающийся способен прочувствовать особенности используемой парадигмы, осознать ее принципы, представлять, что является составляющими программы, написанной на языке, относящейся к данной парадигме, понимать работы основных структурных единиц, владеет некоторыми разрозненными операциями, но не может сочетать их, не владеет структурой их вложенности.

Системный критерий – позволяющий оценить понимание работы более сложных структурных единиц**,** знание некоторых способов сочетания операций конструкций, создания этих сочетаний, умения решать стандартные учебные задачи.

Методологический критерий – характеризующий способность комбинировать простые и сложные структурные единицы программы, составлять сложные взаимовложенные комбинации в зависимости от целей, решаемой практико-ориентированной задачи, умение использовать уже

имеющиеся мыслительные схемы решения основных алгоритмических проблем, навыки их преобразования в изменяющихся условиях или в условиях неполных данных.

Выделенные критерии и их состав оказывают непосредственное влияние как на выбор методов обучения, так и на подбор учебных задач.

В соответствии сформулированными критериями мы выделяем три уровня сформированности программистского стиля мышления: низкий; средний; высокий. Для формирования программистского стиля мышления важны все три критерия, т.е. их совокупность, позволяет определить уровень его сформированности согласно таблице 1.

Таблица 1. Уровни сформированности программистского стиля мышления

Критерий / Уровень	Понятийный	Аналитический	Операционный
Низкий	Не выполнен	Не выполнен	Не выполнен
	Выполнен	Не выполнен	Не выполнен
	Не выполнен	Выполнен	Не выполнен
	Не выполнен	Не выполнен	Выполнен
Средний	Выполнен	Выполнен	Не выполнен
	Не выполнен	Выполнен	Выполнен
	Выполнен	Не выполнен	Выполнен
Высокий	Выполнен	Выполнен	Выполнен

Приведем характеристику каждого из этих уровней.

I уровень – низкий. Условия всех трех критериев удовлетворяются на уровне общих представлений.

II уровень – средний. сформировано умение применять знания по всем трем критериям длярешение задач.

III уровень – высокий. Обладает навыком использовать знания и умения по всем трем критериям в процессе решения практико-ориентированных задач.

Нами разработаны оценочно-диагностическая карта по оценке и диагностике уровня сформированности каждой из компонент программистского стиля мышления.

Оценка общего уровня сформированности программистского стиля мышления осуществляется на основании оценочно-диагностической карты и складывается из пяти составляющих:

1) Уровня сформированности императивного мышления;
2) Уровня сформированности объектно-ориентированного мышления;
3) Уровня сформированности логического мышления;
4) Уровня сформированности функционального мышления;
5) Уровня сформированности параллельного мышления.

Таблица 2. Оценочно-диагностическая карта по оценке и диагностике уровня сформированности каждой из компонент программистского стиля мышления

	Критерии		
Компоненты	**Операционный**	**Системный**	**Методологический**
Императивное мышление	***Низкий уровень*** (задания на знания – вопросы-тесты)		
	- имеется представление об алгоритме и свойствах алгоритмов; - знает основные операции, приемы и методы алгоритмизации; -умение производить разрозненные операции - знать синтаксис основных конструкций языка, их применение при реализации разработанных алгоритмов	умение записывать алгоритм -владение способами составления простых алгоритмов	-способность представлять решение задачи через базовые алгоритмические конструкции -- знать назначение и состав базовых библиотек.
	Средний уровень (задания на умения – учебные задачи)		
	- владение способами составления простых алгоритмов -имеет представление об исполнителе, системе его команд - уметь разрабатывать линейные, разветвляющиеся алгоритмы решения учебно-тренировочных задач;	- умение решать стандартные задачи по образцу; - умение выполнять алгоритм, в том числе пошагово - уметь разрабатывать циклические алгоритмы решения учебно-тренировочных задач;	- умение понять и усвоить алгоритм для решения типовых задач -
	Высокий уровень (задания на навыки – практико-ориентированные задачи)		
	- владение способами сочетания операций алгоритмических конструкций	- владение способами составления сложных алгоритмов - способность	- владение способами составления сложных алгоритмов, в том числе с использованием

	Критерии		
Компоненты	**Операционный**	**Системный**	**Методологический**
	-способность анализировать множество входных и выходных данных - знать простые и сложные типы данных и их применение при написании программ; - уметь самостоятельно программировать основные компоненты базовых алгоритмов	анализировать и учитывать различные сценарии выполнения алгоритма	подалгоритмов - умение проводить анализ задачи, детализацию алгоритма для ее решения - владеть навыками программирования сложных алгоритмов с использованием динамических структур;
Объектно-ориентированное мышление	***Низкий уровень*** (задания на знания – вопросы-тесты)		
	-анализ предметной области задачи и выделение объектов (реальных и абстрактных), построение их иерархии. Выделение основных событий. - знание простых и сложных типов данных и их применение при написании программ; - знание основных понятий объектно-ориентированного программирования: объект, класс, метод и поля класса	-реализация процессов обработки событий. - умение описать свойства и методы, присущие данному объекту. -описывать классы, создавать объекты	-анализ поведения системы и коррекция объектной модели и алгоритмов обработки событий в случае несовпадения полученного результата с предполагаемым - опытом работы в объектно-ориентированных средах программирования - владеет опытом работыь в среде объектно-ориентированного программирования
	Средний уровень (задания на умения – учебные задачки)		
	- Знание основ технологии объектно-ориентированной декомпозиции программных систем, отношений между классами и основ UML (диаграммы классов и последовательностей) - знание принципов объектно-ориентированного программирования	- умение увидеть объекта в иерархии, определить, какие свойства и методы будут наследоваться им от родителя, а какие будут присущи только ему. - умение реализовывать алгоритм решения задачи на языке программирования	- владеет навыками отладки, тестирования и оптимизации программ в объектно-ориентированной среде программирования
	Высокий уровень (задания на навыки – практико-ориентированные задачи)		
	- знание что такое класс и объект, основные принципы объектно-ориентированного программирования, принципы построения классов, критерии проверки правильности построения классов, основные тенденции в области развития технологий объектно-ориентированного программирования - знание теоретических основ объектно-ориентированного анализа, проектирования и программирования - знание абстракций основных структур данных,	- умение использовать современные методы объектно-ориентированного программирования при кодировании программных систем разного уровня сложности - умение реализовывать алгоритм решения задачи на основе объектно-ориентированной парадигме программирования	- владение методами и технологиями программирования в объектно-ориентированных программных средах

	Критерии		
Компоненты	**Операционный**	**Системный**	**Методологический**
	методов их обработки и способов реализации в объектно-ориентированных программных средах		
Логическое мышление	***Низкий уровень*** (задания на знания – вопросы-тесты)		
	-знание основных принципов и способов логического программирования - знать простые и сложные типы данных и их применение при написании программ	- умение выбирать структуры данных, необходимых для решения задачи проектирования	- владение способами составления «вложенных» правил
	Средний уровень (задания на умения – учебные задачки)		
	- умение правильно совершать такие мыслительные операции, как: классификация, конкретизация, обобщение, сравнение, аналогия и другие - владеть представлением о логическом программировании и альтернативных методах программирования (машины на не архитектуре фон-Неймана и порождаемые этим последствия для языка)	- умение работать в среде различных инструментальных средств по проектированию	- Навык быстро и эффективно решать сложные логические задачи (как учебные, так и прикладные)
	Высокий уровень (задания на навыки – практико-ориентированные задачи)		
	- знание инструментальных систем, поддерживающих логическое программирование - знание реализации многомодульных приложений в среде языка Пролог	- умение анализировать результаты реализации приложений, разработку интерфейса приложения	-владение основными элементами и технологиями программной инженерии на этапе проектирования программных комплексов
Функциональное мышление	***Низкий уровень*** (задания на знания – вопросы-тесты)		
	- знание базовых понятий по основам функционального программирования	- умение пользоваться языком логического программирования	- способность реализовывать различные комбинации
	Средний уровень (задания на умения – учебные задачки)		
	- знание математических основ, основных концепций и приемов функционального программирования	- умение свободно пользоваться языком логического программирования, выбирать средства реализации конкретной программы	- владение средствами функционального программирования
	Высокий уровень (задания на навыки – практико-ориентированные задачи)		
	-знание основных способов создания интеллектуальных приложений на базе использования функционального	-способен создавать приложения с использованием средств функционального программирования	- применение знаний о принципах и технологиях функционального программирования в создании программного

	Критерии		
Компоненты	**Операционный**	**Системный**	**Методологический**
	программирования		обеспечения
Параллельное мышление	***Низкий уровень*** (задания на знания – вопросы-тесты)		
	- знание основных понятий технологии параллельного и программирования: - знание основ архитектуры параллельных вычислительных комплексов	-умение оценивать асимптотическую сложность используемых алгоритмов и выбирать оптимальные алгоритмы для современных программ	- владение приемами распараллеливания алгоритмов и программ
	Средний уровень (задания на умения – учебные задачки)		
	- знание основных технологических этапов разработки параллельных программ; - знание способов эквивалентных и неэквивалентных преобразований последовательных программ, позволяющих использовать их на параллельных вычислительных комплексах	- умение анализировать последовательные программы для выявления возможностей их распараллеливания	- владение средствами и технологиями разработки приложений, обеспечивающих проведение параллельного вычислительного эксперимента
	Высокий уровень (задания на навыки – практико-ориентированные задачи)		
	- Выделение относительных независимых подзадач, определение инф.зависимости между подзадачами, программирование подзадач, организация простого взаимодействия	-сложное заимодействие (множества, обмен между подзадачами), устранение простых ошибок, синхронизация, deadlock	- владение навыкамми отладки и запуска параллельных приложений для проведения вычислительного эксперимента Ярусно-параллельная форма и анализ алгоритма, гибридные программы

В общем виде уровень сформированности программистского стиля мышления представляется в трехмерном пространстве, в виде параллелепипеда, состоящий из 45 частей (рис.5). Каждая часть параллелепипеда описывает различные уровни программистского стиля мышления. Например, сегмент (1,1,1) – это уровень полного непонимания. Последний уровень (3,3,5) описывает ситуацию, когда происходит полное понимание. В процессе обучения нас интересуют те уровни, которые обеспечивают достаточный уровень сформированности программистского стиля мышления.

Рис.5 Уровни и критерии программистского стиля мышления

Выполнимость критерия определяется с помощью соответствующих тестовых заданий, заданий в контрольных работах, результатов наблюдений, опросов и анкетирования.

Диагностировать уровень сформированности того или иного компонента программистского стиля мышления у обучающихся предлагается с помощью заданий на каждом уровне: на низком уровне – заданий на знания в виде вопросов-тестов, на среднем уровне – задания на умения – учебные задачки, на высоком уровне – задания на навыки – практико-ориентированные задачи. Количество заданий для каждого компонента программистского стиля мышления предлагается разное.

Для того чтобы определить уровень сформированности того или иного компонента, во всех методиках подсчитывается количество правильно решенных задач по отношению к общему количеству задач. Это отношение, умноженное на 100, называется коэффициентом успешности (КУ). Затем высчитывается среднеарифметическое по всем трем критериям оценивания, и определяется уровень сформированности того или иного компонента программистского стиля мышления. Так низкому уровню сформированности компонента программистского стиля мышления соответствует средний КУ

менее 50% по всем критериям. Средний уровень составляет КУ в пределах от 51% до 75% по всем критериям. Высоким уровнем сформированности компонента программистского стиля мышления обладают обучающиеся, средний КУ которых более 75% по всем критериям.

Подсчет общего балла, характеризующего уровень сформированности программистского стиля мышления у учащихся определяется из суммы оценки среднего арифметического каждой из компонент, входящих в состав данного стиля мышления:

$ПМ_{ср}=А_{ср.ар.}+В_{ср.ар}+С_{ср.ар.}+Д_{ср.ар.}+Е_{ср.ар.}$,

где $ПМ_{ср}$ – средний уровень сформированности ПМ, $А_{ср.ар}$ – среднеарифметическое уровня сформированности императивного мышления, $В_{ср.ар}$ – среднеарифметическое уровня сформированности объектно-ориентированного мышления, $С_{ср.ар}$ – среднеарифметическое уровня сформированности логического мышления, $Д_{ср.ар.}$ – среднеарифметическое уровня сформированности функционального мышления, $Е_{ср.ар.}$ – среднеарифметическое уровня сформированности параллельного мышления,

1.2. Телесный подход к обучению программированию

В настоящее время существует ряд российских и западных исследований, посвященных разработке методических подходов к обучению программированию, среди которых можно выделить: дифференцированный подход, системный подход, деятельностный подход, когнитивный подход, проблемный подход, семиотический подход [20].

Дифференцированный подход к обучению предполагает организацию учебного процесса, при которой учитываются индивидуально-типологические особенности личности (способности общие и специальные, уровень развития, интересы, психофизиологические свойства нервной системы и т.д.), характеризуется созданием групп учащихся, в которых

содержание образования, методы обучения, организационные формы различаются [73].

Самыми распространенными формами дифференциации являются а) выполнение учащимися заданий разного уровня сложности и различной направленности на развитие ученика; б) дозирование помощи учителя ученикам; в) групповая работа учащихся по модели полного усвоения.

Системный подход к обучению предполагает рассмотрение объектов как систем. Он ориентирует исследование на раскрытие целостности объекта, на выявление многообразных типов связей в нем и сведение их в единую теоретическую картину [89].

Основная идея системного подхода в контексте обучения программированию состоит в рассмотрении учебных задач в тесной взаимосвязи со средствами, методами программирования и в целом с технологическим процессом. Изучение каждого из этих направлений должно происходить не изолировано, а в тесных связях и зависимостях между ними.

Деятельностный подход основан на «принципиальном положении о том, что психика человека неразрывно связана с его деятельностью и деятельностью обусловлена» [20; 33].

Под деятельностным подходом понимают такой способ организации учебно-познавательной деятельности обучаемых, при котором они являются не пассивными «приёмниками» информации, а сами активно участвуют в учебном процессе. Суть деятельностного подхода в обучении состоит в направлении «всех педагогических мер на организацию интенсивной, постоянно усложняющейся деятельности» [27].

В обучении информатике и программированию концепция деятельностного подхода предполагает:

а) формирование готовности ученика на активное, адекватное и эффективное применение информационных и телекоммуникационных технологий в процессе обучения информатике,

b) выявление и формирование творческой индивидуальности ученика,

c) развитие его будущих профессиональных взглядов.

Когнитивный подход предполагает овладение учащимися операциями программирования, начиная с элементарных, постепенно продвигаясь к сложным.

Основная цель когнитивного подхода – развитие когнитивных качеств обучающихся: быстроты и экономичности мышления, способности к решению нестандартных проблем, готовности воспринимать и анализировать противоречивую информацию [20].

Реализация когнитивного подхода возможна с помощью метода демонстрационных примеров, метода ключевых задач, метода раскрутки задач и др.

Проблемный подход предполагает, что весь курс программирования строится на основе системы специально разработанных «проблемных» задач или ситуаций, содержание которых интересно для учащихся и, по возможности, связано с их будущей профессиональной деятельностью [20].

Семиотический подход к обучению программированию заключается в целенаправленном развитии у обучающихся знаково-символических действий (замещения, кодирования, схематизации, моделирования). При этом учитывается, что обучающийся, уже знаком с некоторыми знаковыми системами (русским языком, дорожными знаками, позиционными системами счисления) и имеющиеся у него знания, опыт деятельности можно эффективно использовать при изучении особенностей языков программирования как знаковых систем [32].

Целесообразность применения семиотического подхода к обучению программированию может быть обоснована тем, что язык программирования изначально имеет знаковую природу. Как всякий искусственный язык он имеет «повторный перевод», другими словами, знания о мире формализуются, «переводятся» на обычный язык, а затем находят свое

отражение в алгоритмических конструкциях, встроенных функциях и др., что влечет за собой постепенное развитие языка программирования [20].

Ментальный подход рассматривает реальность и человека с позиции общекультурной и индивидуальной смысловой матрицы (ментальности), возникшей в результате формирования и образования личности и дальнейшей интерпретации индивидуального и коллективного жизненного опыта. Каждый человек является носителем ментальности – программы выживания и организации жизни («программы жизни»), которая предопределяет мысли и возможный набор поступков человека в любой возможной жизненной ситуации еще до возникновения реальной ситуации. Ментальность выражена в системе ценностей человека, его убеждениях и жизненных принципах, закрепленных в организации логики мышления, неосознанных решениях и моделях поведения, реализуется в чувствах и ощущениях человека, в его восприятии и интерпретации реальности [113].

Одним из принципов семиотического подхода является принцип учета ведущего канала восприятия. Согласно этому принципу, должны учитываться особенности учащихся с разными ведущими каналами восприятия (визуалами, аудиалами, дигиталами и кинестетиками). В этом отношении новым и мало изученным является телесный подход, в котором учитывается взаимосвязь телесных, кинестетических ощущений, восприятия и развития мышления.

Суть телесного подхода заключается в том, чтобы воздействовать преимущественно на кинестетические каналы восприятия, моторную область памяти, при том, что все остальные подходы к обучению воздействуют на дигитальные, аудиальные и/или визуальные каналы восприятия. Между тем, согласно телесному подходу, наше мышление «отелеснено», неразрывно связано с телесными ощущениями, и, следовательно, этот подход является единственно приемлемым для обучающихся с ведущим кинестетическим

каналом восприятия и существенно более эффективным для всех остальных, поскольку задействует все каналы восприятия.

Телесный подход начал интенсивно развиваться в западной когнитивной науке примерно с начала 1990-х годов. Английское словосочетание «embodied cognition approach» точнее было бы переводить как подход с точки зрения «отелесненности» процесса познания, телесной облеченности всякого познающего существа. Такое уточнение имеется в виду, когда говорят неуклюже, но кратко: «телесный подход». Среди создателей нового подхода такие ученые, как Р. Бир, Р. Брукс, Т. Ванн Гелдер, Э. Кларк, Ж. Лакофф, П. Маес, Э. Прем, Э. Телен, Ф. Варела и ряд других [4]. Другими словами, телесный подход фокусирует свое внимание на телесной организации познающего существа, так как то, что познается и как познается, зависит от телесности живого существа и его конкретных функциональных особенностей.

Возникновение и быстрое развитие телесного подхода было подстегнуто глубокой неудовлетворенностью ряда ученых доминировавшим с 60-х гг. вычислительным подходом (computation alapproach) к объяснению познавательных способностей человека и животных. Основной сферой приложения усилий представителей вычислительного подхода была проблема искусственного интеллекта. Идеалом виделась возможность построения системы, полностью имитирующей человеческий интеллект. В качестве модели для имитации брался компьютер. Предполагалось, что и мозг работает по принципам компьютера. Наглядным образцом такого рода устройства стали программы для игры в шахматы, основанные на просчитывании всех возможных ходов максимально далеко вперед.

Создателей этих программ радовало и обнадеживало то, что возможности компьютера в чем-то даже превосходят возможности человеческого интеллекта. Казалось, стоит еще и еще повысить тактовую частоту процессора, добавить гигабайтов памяти, объединить множество

компьютеров в единую сеть, запустив по возможности параллельно, что уподобило бы их нейронной сети мозга, одновременно выполняющей множество задач – и цель достигнута.

Сторонники телесного подхода выдвинули следующее возражение резервы на этом пути, конечно, есть, но сам путь обходит стороной главное.

Их возражения в адрес вычислительного подхода в обобщенной форме можно сформулировать следующим образом:

1. Вычислительный подход сводит функции познания к функциям чистого, абстрактного интеллекта. Интеллект в этом случае существует как бы вне тела, вне физического организма, взятого в его естественном функционировании и движении и в окружении других материальных тел. Фактор телесной облеченности субъекта восприятия и мышления в вычислительной модели не только излишен, но и вреден, поскольку только затемняет и искажает деятельность чистого интеллекта. Тем самым эта модель лишается связи с реальностью, а потому объяснительной и эвристической силы.

2. Вычислительный подход рассматривает когнитивные функции, причем сведенные лишь к интеллектуальным функциям, в их наличной данности, в полностью развитом виде, игнорируя и общее эволюционное происхождение этих функций (процесс филогенеза), и постепенность их формирования в процессе индивидуального развития особи (процесс онтогенеза).

3. Мыслительные операции человека, в представлении вычислительного подхода, строятся по принципу символического представления (репрезентации – поэтому вычислительный подход нередко называют также репрезентатистским), который лежит в основе работы компьютера, где входные данные переводятся на особый символический язык, посредством которого обрабатываются. Это означает, что если процесс «вне» головы, вызвавший когнитивный акт, можно объяснить как

физический динамический процесс, то процесс «в» голове следует объяснять по законам семантики, т.е. смыслового отношения одной системы символов с другой системой символов. Тем самым процесс познания, а с ним и мир в целом, оказывается удвоен, разорван, по меньшей мере, на две несводимые реальности – физическую и семантическую.

Черты самоорганизации и самодвижения в компьютерах весьма ограничены. Моделируя когнитивные функции по образцу функций компьютера, мы лишаем этих черт и естественное функционирование мозга, а потому нуждаемся во введении в модель мозга внешней движущей силы или программы, посредством которой осуществляются самотестирование системы и ее самоконтроль.

Ведь, во-первых, если подходить к пониманию деятельности мозга по семантической схеме, вряд ли можно утверждать, что семантические системы, в отличие от физических динамических систем, способны к спонтанному движению, в том числе «переведению» себя в другую систему символов. Во-вторых, с технической точки зрения, компьютерные операции осуществляются в тактовом режиме, а не непрерывно: с каждым новым тактом, отмеряемым процессором, изменяется распределение электромагнитных состояний, что в более широком временном масштабе выглядит как непрерывная работа компьютера. Но такого рода движение есть не подлинное динамическое движение, а последовательность сменяющих друг друга статических состояний. Тем более не есть оно самодвижение, поскольку осуществляется только путем «подталкивания» со стороны процессора, и одно статическое состояние никоим образом не способно самопроизвольно перетекать в другое состояние, разве что как сбой системы.

В противовес вычислительному подходу была выдвинута теоретическая концепция, базирующаяся на следующих утверждениях.

1. Познание телесно, или «отелесненно». То, что познается и как познается, зависит от строения тела и его конкретных функциональных

особенностей, способностей восприятия и движения в пространстве. Устроены тела по-разному – потому и познают тела по-разному. Если раньше гносеологи говорили, что познание теоретически нагружено (т.е. то, что мы видим, во многом определяется имеющимися у нас теоретическими представлениями), то теперь можно сказать к тому же, что познание телесно нагружено.

Нельзя понять работу человеческого ума, если брать ум абстрагированнным от организма, телесности, определенным образом обусловленного восприятия посредством конкретных органов чувств.

2. Познание ситуационно. Познающее тело погружено в более широкое – внешнее природное и, в случае человека, социокультурное окружение, оказывающее свои влияния.

3. Познание инактивировано (enacted cognition). Познание осуществляется в действии и через действия. Через действия формируются когнитивные способности, как видовые, так и индивидуальные. Когнитивная активность в мире создает и саму окружающую по отношению к познающему существу среду – в смысле отбора, вырезания им из мира именно и только того, что соответствует его телесным потребностям, когнитивным способностям и установкам.

4. Познавательные системы есть динамические и самоорганизующиеся системы. В этом функционирование познавательных систем принципиально сходно, единосущно функционированию познаваемых природных систем, т.е. объектов окружающего мира. Потому и мир остается един, а не разорван на динамическую «внеголовную» и семантическую «внутриголовную» половины.

С точки зрения сторонников вычислительного подхода, мир поделен на две несводимые реальности: символическую, представленную внутри головы, и физическую – вовне. Поэтому для них встает проблема: как соединить обе эти реальности или объяснить одну через другую.

Телесный подход говорит о единстве реальности и принадлежности к ней как агента со свойственными ему когнитивными процессами, так и внешней ему среды. Эта единая реальность имеет физический характер, а процессы в ней являются динамическими процессами самоорганизации.

В телесном подходе присутствуют два противоположных мотива в трактовке роли тела в процессе познания. С одной стороны, только через тело и его движение формируется и осуществляется познание. Но, с другой стороны, тело привносит такие искажающие влияния, от которых приходится избавляться. Идея чистого интеллекта присутствует и как предмет критики, с точки зрения ее нежизненности и научной несостоятельности, и как идеал. Сам по себе недостижимый, именно он задает направленность вектора объективации. Такое противоречие можно снять в уме с помощью гегелевской диалектической триады: до помещения в тело чистый интеллект есть выхолощенное представление, но после помещения в тело и с учетом такой помещенности – на новом витке спирали, он становится важным притягивающим аттрактором и задающим критерием.

В телесном подходе получила развитие тема категоризации человеком мира исходя из физического существования его тела. Сетка этих категорий как бы накладывается на мир, и только согласно ее клеточкам мир и воспринимается. Категории эти не случайны и не произвольны, как полагалось ранее. Они происходят из телесного опыта, а он имеет повторяющийся и устойчивый характер [111].

М.А. Степанов отмечает, что телесный подход открывает новую перспективу на культурный мир и человека в нём, где главное отличие проходит через смену отношения к месту и времени: истолкование традиционно происходит после события, телесное познание требует готовности к будущему и решительности в спонтанном акте его реализации. Или, иначе говоря, компетентности и способности к действию [105].

Современная школа ориентирована преимущественно на логическую и аудиальную стороны мышления. По мнению профессора психологии Гарвардского лингвистическим университета Говарда Гарднера [24], учебные заведения созданы для детей с логико-математическими лингвистическим типом мышления. Однако в современных психологических исследованиях кроме них выделяют телесно-кинестетический, визуально-пространственный, внутриличностый типы мышления.

Согласно полученным психологами данным, лингвистический и логический (математический) типы интеллекта вовсе не преобладают среди молодежи, тем более среди подростков или младших школьников. Получается, что большинство детей разного возраста остается за бортом качественного образования.

Практикующие психологи дают свои рекомендации по особенностям воспитания и обучения детей с разными типами интеллекта [24]. Так, у частности, если у ребенка преобладает телесно-кинестетический интеллект, то обработка информации у него происходит на уровне тактильных сигналов. У такого ученика умелые руки и отличное чувство баланса. Он лучше всего чувствует себя в случае, когда надо много времени посвящать физической деятельности, ремеслу, он легко управляет предметами. Для максимально эффективного обучения ему необходимо прикасаться, двигаться и обрабатывать поступившую информацию посредством тактильных ощущений.

Если ребенок кинестетик, то он предпочитает учиться, выполняя задание. Самым эффективным способом обучения для таких детей, как правило, является экспериментирование, а также разработка способов решения поставленной задачи. Ребенок-кинестетик лучше всего запоминает и усваивает информацию посредством прикладного опыта, в движении, в ролевых играх, в игре и групповых мастерских, в импровизации, в работе с материалами.

Представляется, что если строить систему обучения, опираясь на эти советы, использовать средства обучения, нацеленные на активизацию кроме визуальной, аудиальной и абстрактной еще и моторной зоны памяти, процесс обучения программированию будет проходить более эффективно.

1.3. Полиязыковой подход обучения программированию

Совершенствование системы среднего образования предъявляет высокие требования к уровню предметной подготовки учителя. Несомненно, его научный кругозор должен значительно превышать рамки школьной программы. Следовательно, несмотря на то, что в школьном курсе программирования изучаются только алгоритмические языки, относящиеся к императивной парадигме, учитель информатики должен владеть всеми современными парадигмами и технологиями программирования.

Курс «Языки и методы программирования» является одним из базовых в системе предметной подготовки бакалавров педагогического образования по профилю «информатика», поскольку в основном именно в этом курсе реализуется подготовка будущих учителей информатики в области алгоритмизации и программирования. Актуальность и важность этой подготовки обуславливается, во-первых, тем фактом, что цель обновленного, согласно ФГОС второго поколения, школьного курса информатики не изучение различных частных аспектов прикладной, технологической направленности, а формирование представления об информатике как фундаментальной естественнонаучной дисциплине. Линия «Алгоритмизация и программирование» как раз является одной из основных фундаментальных составляющих школьного курса информатики.

Во-вторых, от уровня подготовки в области алгоритмизации и программирования напрямую зависит успешность освоения большинства

дисциплин предметной подготовки: «Численные методы», «Компьютерное моделирование», «Исследование операций» и т.п.

По этой причине курс «Языки и методы программирования», являющийся одним из основных курсов в предметной подготовке будущих учителей информатики, предполагает изучение языков программирования, относящихся к различным парадигмам программирования, сложившимся в современной науке – не только императивной, но и декларативных – объектно-ориентированной, функциональной, логической. Современный уровень развития суперкомпьютерных технологий, основанных на параллельных вычислениях, повсеместное использование многоядерных процессоров предполагает также изучение параллельного программирования.

Несмотря на многочисленные исследования, проблема готовности будущего учителя информатики к решению практических задач, связанных с алгоритмизацией и программированием продолжает оставаться актуальной.

Изучение языка программирования относящегося к другой парадигме, вызывает определенный ряд сложностей. Так как при переходе к программированию методами, которые относятся к другой парадигме, необходимо изменить не только подход к решению поставленной задачи, но и перестроить мыслительную деятельность относительно новой парадигмы. Каждая парадигма программирования предполагает формирование определенного стиля мышления – объектного, функционального, логического, параллельного [70; 63].

Практически во всех исследованиях [11; 25] предлагается изучать различные парадигмы последовательно, и исследователи спорят лишь об оптимальной последовательности их изучения. Мы предлагаем, при проектировании методической системы обучения курсу «Языки и методы программирования» предусмотреть параллельное изучение парадигм программирования.

Применение параллельного способа обучения при изучении дисциплин предметной подготовки рассмотрено в работах Н.И.Пака и Т.А.Степановой [81; 82]. Параллельному способу изучения языков программирования, относящихся к одной парадигме, посвящены исследования В.Е.Жужжалова [25; 26].

Мы в исследовании предлагаем методику параллельного изучения языков, относящиеся к различным парадигмам программирования.

Подавляющее большинство исследователей осознают, что сформированное на достаточном уровне алгоритмическое мышление создает сложности при переходе на логическое программирование, объектно-ориентированное – на функциональное и наоборот. При изучении парадигм программирования параллельно это проблема значительно ослабляется.

Параллельный способ обучения в исследовании мы будем называть полиязыковым обучением, которое, по сути, является интегрированным подходом к обучению языкам программирования.

Реализация полиязыкового подхода в курсе «Языки и методы программирования» осуществляется следующим образом.

Проводится предварительный анализ содержания учебного материала, с целью выявления тех тем, которые

- целесообразно изучать параллельно,
- необходимо изучить предварительно, для успешной организации параллельного процесса,
- будут изучаться после (либо будут вкраплены в процесс параллельного изучения), это те темы, которые являются специфичными именно для определенной парадигмы, темы, которые невозможно запараллелить.

Результатом такого анализа будет являться принципиально новый учебно-тематический план курса «Языки и методы программирования», существенно отличающийся от учебно-тематического плана этого же курса,

разработанный на основе последовательного изучения парадигм программирования, хотя бы в том плане, что, допустим, если мы предполагаем последовательно изучать, скажем Паскаль, Си, Пролог и Лисп, то многие темы будут дублироваться: структура программы, типы данных, реализация основных алгоритмических конструкций и т.п., потребуется как минимум четыре теоретических и четыре практических занятия на изучение каждой темы, в каждом из изучаемых языков отдельно. При полиязыковом обучении этих языков на изучение каждой темы отводится одно теоретическое занятие, на котором рассматривается, как эта тема реализована одновременно на всех четырех языках и необходимое количество практических занятий, но в любом случае, оно не будет превышать четырех.

Для проведения лабораторных занятий разрабатывается система задач, которые могут быть выполнены посредством языков программирования, относящихся к различным парадигмам, для каждой задачи определяются парадигмы, в которых будет предложено реализовать поставленную задачу.

Возникают опасения, что при полиязыковом обучении синтаксис языков программирования в голове студентов «свалится в кучу», особенности различных парадигм смешаются. Однако, при умелых методических приемах, напротив, полиязыковой (параллельный) подход к обучению имеет преимущество над последовательным, использовавшимся до сих пор. Педагогический опыт показывает, что при последовательном изучении, например, Паскаля и Си, студенты мучительно вспоминают, что относится к одному языку, что к другому, где какие операторные скобки ставятся, как описываются и вызываются подпрограмм и т.п. Изучая же языки параллельно, преподаватель сразу разводит особенности синтаксиса разных языков, используя при этом методы визуализации (допустим, схему, на которой представлена реализация одной и той же алгоритмический структуры в различных языках). И тогда восприятие студентов заостряется

именно на отличиях синтаксиса языков, при написании программного кода в голове встает эта схема – в Паскале так, в Си – вот так.

Кроме того, полиязыковому параллельному изучению парадигм программирования обязательно предшествует первая, вводная лекция, на которой констатируется тот факт, что в современном программировании сложились данные парадигмы, излагается история возникновения и развития парадигм, кратко характеризуются их особенности. Завершает процесс параллельного изучения парадигм лекция, посвященная сравнительному анализу изученных парадигм, их отличий, преимуществ и недостатков, сфер применения. Здесь также эффективно применить визуальные методы представления учебной информации, например, оформить результаты проведенного анализа в виде таблицы.

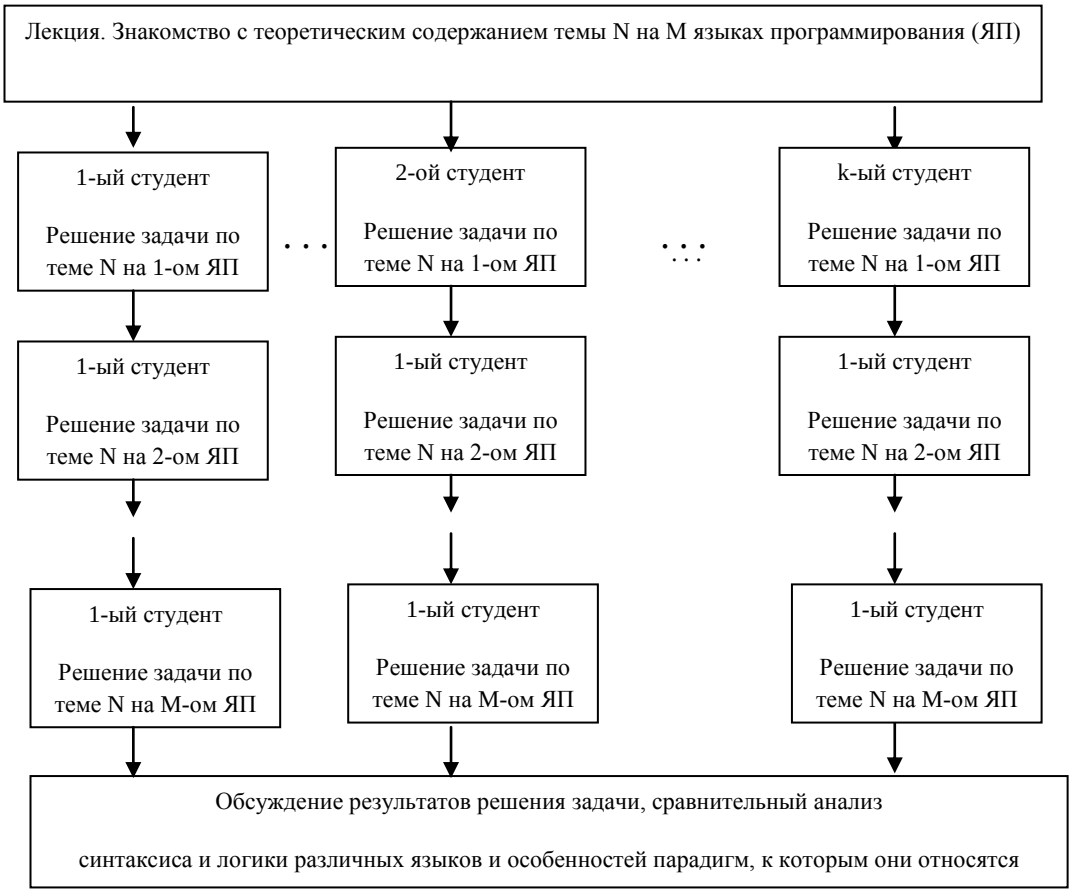
Согласно методике полиязыкового подхода к изучению парадигм программирования теоретический материал на лекциях дается параллельно. Например, излагается каким образом некоторая конкретная цель, скажем, организация повторений, достигается в императивной парадигме (циклы), а каким – в логической парадигме (рекурсия). В рамках одной парадигмы синтаксис языков также возможно давать на лекции параллельно – вот такие виды циклов существуют в языке Паскаль, вот их синтаксис, вот такие типы циклов существуют в Си, вот их синтаксис.

Проведение лабораторных занятий по курсу основывается на следующем принципе: на каждом лабораторном занятии студентам предлагается решить одну и ту же конкретную задачу средствами языков программирования, относящимся к различным парадигмам, причем преподаватель не акцентирует внимание, что, например, алгоритмическая парадигма предназначена для решения вычислительных задач, а логическая – интеллектуальных. К таким выводам студенты должны придти сами, провести в конце лабораторной работы обсуждение полученных результатов, сравнить эффективность решения той или иной задачи при помощи

различных парадигм. Используя этот методический прием, мы добьемся осознанного понимания особенностей каждой парадигмы, будущий специалист сможет с легкостью переходит с одной парадигмы на другую, определить, применение какой парадигмы будет более оптимальным для решения поставленной задачи.

Организовать параллельный способ обучения на лабораторных занятиях можно с помощью различных подходов (рис.6). Первый подход предполагает, что каждый студент получает свою задачу и выполняет ее при помощи нескольких парадигм, на одном занятии получая навыки работы в каждой из них. Программные коды он пишет, конечно, последовательно, но в голове все время присутствует мысль, что ему предстоит реализовать этот же алгоритм и на других языках, и непроизвольно возникают рассуждения, что на Паскале этот цикл будет выглядеть так, а в Си вот так, а на Прологе вместо этого цикла придется организовать рекурсию. Т.е., несмотря на то, что программные коды пишутся последовательно, рассуждения протекают параллельно, теоретические сведения по синтаксису и логике языков программирования, относящихся к различным парадигмам, закрепляются на таком лабораторном занятии все равно параллельно. Завершает такое занятие обсуждение результатов решения задачи, сравнительный анализ синтаксиса и логики различных языков и особенностей парадигм, к которым они относятся. Его может провести преподаватель, подводя итоги выполнения задач, акцентируя эти моменты в своем заключительном слове, а можно поставить такую цель перед каждым студентом. Тогда, чтобы сдать выполненную лабораторную работу нужно будет не только продемонстрировать преподавателю работающие программы, выполненные на различных языках программирования, но и провести подобный анализ.

Первый подход



Второй подход

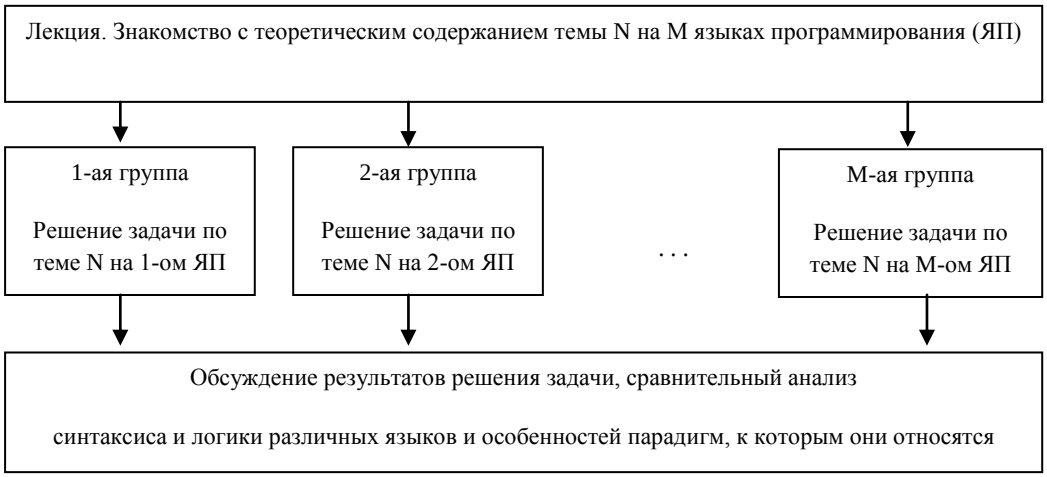


Рис. 6. Схемы организации параллельного способа обучения

Этот подход можно использовать для задач, решение которых требует написания сравнительно небольшого программного кода, например, вычисление факториала. Для более сложных и объемных задач, например, умножение матрицы на вектор, лучше использовать второй подход, который

предусматривает формирование нескольких рабочих групп студентов (по числу языков программирования, в которых преподаватель планирует реализовать решение поставленной задачи).

Каждая рабочая группа реализует решение одной и той же задачи при помощи различных языков программирования, в том числе и относящихся к различным парадигмам. Выполнение задания в этом случае рассматривается как учебный мини-проект. В конце лабораторного занятия (или в начале следующего, если у рабочих групп возникнет необходимость самостоятельной доработки проектов) организуется защита проектов, на которой каждая рабочая группа представляет и комментирует свой способ решения задачи, определяемый особенностями выбранной парадигмой, свой программный код, отражающий особенности синтаксиса языка программирования. Подводя итог защите проектов, преподаватель совместно со студентами проводит анализ эффективности выполнения поставленной задачи в различных парадигмах. В результате все студенты получат навыки реализации поставленной задачи во всех парадигмах. Те, кто выполнял эту задачу – непосредственно, а те, кто разбирал решение задачи во время защиты проектов – опосредованно. Непосредственное закрепление практических навыков можно в этом случае вынести на самостоятельную работу.

Как видно из приведенного описания, параллельный способ обучения опирается на методики проблемного обучения: метод проектов, предложенный Дж. Дьюи [71], коллективный способ обучения, разработанный В.К. Дьяченко [22], технологию конференции однородных групп, рассмотренную в работах А.К. Колеченко [39].

На наш взгляд, использование параллельного способа изучения различных парадигм программирования в курсе «Языки и методы программирования» способствует качественному усвоению учебного материала этого курса, профессиональной готовности будущих учителей

информатики к многовариантному решению различного типа практических задач и обеспечит дальнейшее развитие мировоззрения студентов, расширение их научного кругозора.

Схема организации учебного процесса на основе полиязыкового подхода представлена на рисунке 7.

Рис. 7. Структурная схема полиязыкового подхода к решению задачи

Безусловно, полиязыковый поход требует больших временных затрат и умственных усилий студента, но он по сути реализует студент-центрированный принцип обучения. Разные стили мышления студентов позволяют им индивидуально уловить смысл одного из способов представления алгоритма задачи, а затем с легкостью освоить другие парадигмы программирования.

Выводы по первой главе

В первой главе рассмотрены теоретические основы телесно-полиязыкового обучения программированию. На основе пространственно-временной модели памяти и положений психологической теории деятельности построена структурная модель программистского стиля мышления, уточнено содержания понятия «программистский стиль мышления». Выделены критерии, которые в совокупности могут охарактеризовать уровень сформированности программистского стиля мышления. А также выявлены возможности использования телесного и полиязыкового подходов к обучению студентов программированию.

Глава 2. Методика обучения программированию будущих учителей информатики на основе телесно-полиязыкового подхода

2.1. Структурно-логическая модель обучения студентов курсу «Языки и методы программирования»

Методика обучения программированию будущих учителей информатики на основе телесно-полиязыкового подхода представлена структурно-логической моделью, в которой присутствуют следующие компоненты: целевой, содержательный, процессуальный, результативный (рис.8). Модель отражает взаимосвязь всех компонент методики обучения и их состав, выбор подходов к обучению, влияющих на содержание компонент.

Целевой компонент является системообразующим, он определяет принципы, содержание, педагогические условия, методы, формы и средства обучения, способы контроля. Целевой компонент представляет собой совокупность целей. В основу выстраивания системы целей изучения курса «Языки и методы программирования» в педагогическом вузе положены следующие принципы:

1. Соответствие целей обучения программированию современному состоянию науки.
2. Соответствие целей обучения программированию федерального государственному образовательному стандарту.

Целью предлагаемой модели является формирование компонентов программистского стиля мышления учащихся на материале курса «Языки и методы программирования».

Помимо цели процесс формирования программистского стиля мышления учащихся определяется действием как общих принципов педагогического процесса, так и целым рядом дополнительных принципов [41].

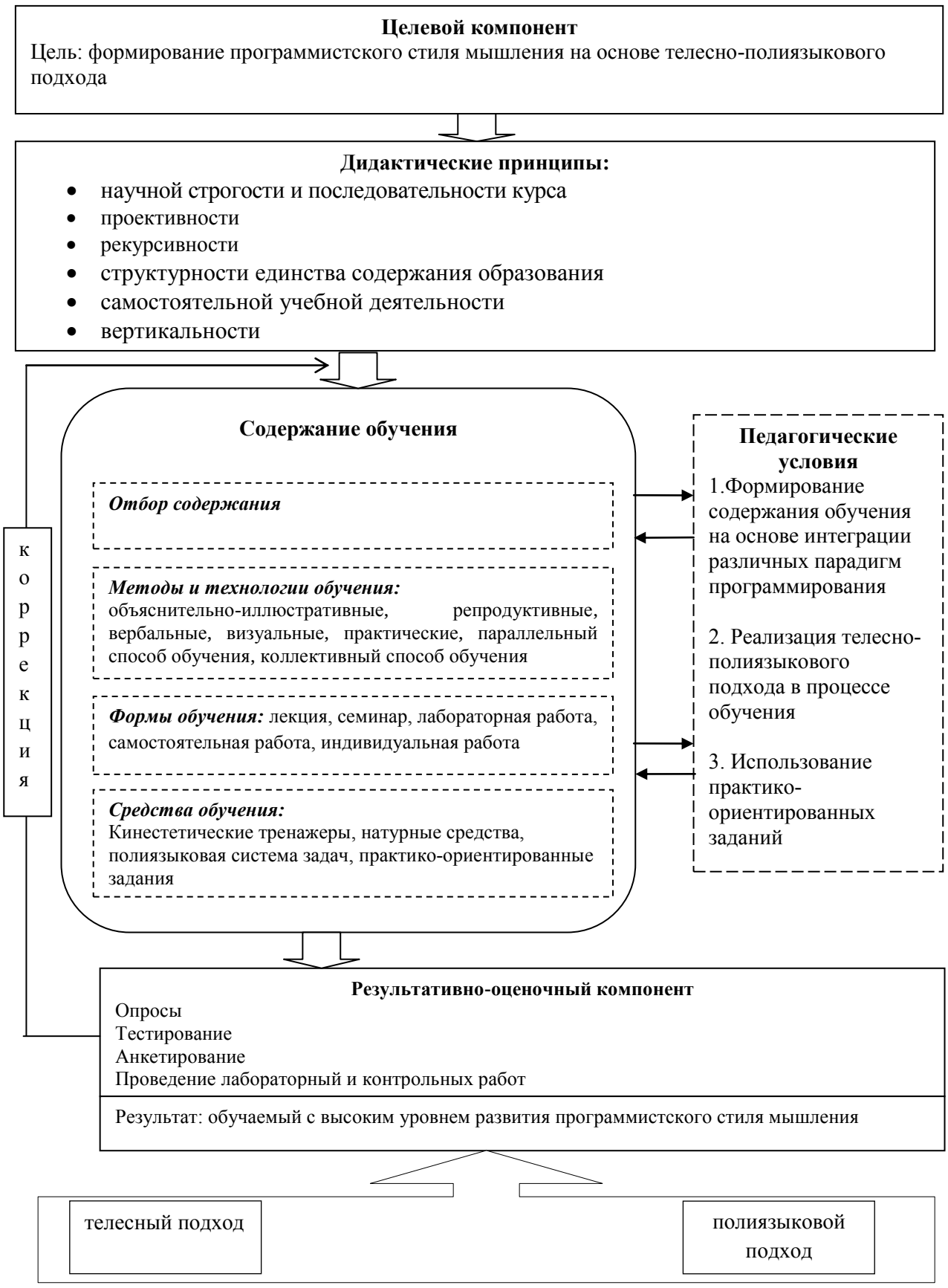


Рис. 8. Структурно-логическая модель методики обучения будущих учителей информатики программированию

Общедидактические принципы, определяющие содержание, организационные формы и методы учебного процесса, были изложены в работах М.А. Данилова, В.А. Дистервега, Б.П. Есипова, И.Я. Лернера, М.Н. Скаткина [51; 52; 100] – это принципы воспитывающего и развивающего обучения, связи обучения с жизнью, научности и посильной трудности, систематичности и последовательности, сознательности и творческой активности, наглядности в обучении и другие.

В плане построения системы формирования программистского стиля мышления на основе телесно-полиязыкового подхода мы полагаем необходимым опираться дополнительно на следующие принципы:

- *принцип научной строгости и последовательности курса* (предполагает непротиворечивость и логическую последовательность изложения материала. Для практической реализации данного критерия отбора содержания определены критерии научной строгости и последовательности учебного материала: каждая тема должна быть изложена логически непротиворечиво, реализация каждой темы должна отвечать оценке научного уровня и характеристикам логической строгости);

- *принцип проективность* [5; 79] (проекция будущей профессиональной деятельности на педагогический процесс в виде реализации личного образовательного проекта);

- *принцип рекурсивности* [5; 79](создание и использование образовательных ресурсов самими обучающимися в учебном процессе);

-*принцип структурности единства содержания образования* на разных уровнях его формирования с учетом личностного развития и становления обучающегося, предполагающий взаимную уравновешенность, пропорциональность и гармоничность компонентов образования;

- *принцип самостоятельной учебной деятельности* [23; 87] (направленный на создание условий по включению учащихся в различные виды самостоятельной деятельности);

- *принцип параллельности* [26; 81; 82]

- *принцип вертикальности* [83; 84; 79] (предполагает выстроить непрерывной процесс формирования программистского стиля мышления).

Ключевая цель курса – формирование системы понятий, знаний, умений и навыков в области структурного, объектно-ориентированного, функционального и логического программирования на базе языков Паскаль, Дельфи (C, C++), Lisp и Prolog. Так же, параллельно, изучаются методы проектирования, анализа и создания программного обеспечения.

Выбор цели обусловлен следующими положениями: во-первых, программирование является существенной частью предметной подготовки педагогов-бакалавров профилей «Информатика», «Математика и информатика», «Физика и информатика». Во-вторых, программирование является предметом, позволяющим не только развивать способы алгоритмической мыслительной деятельности, но и реализовывать возможности межпредметных связей.

В данном учебном курсе рассмотрение начинается с основ программирования на языке Паскаль в среде Delphi 7 (C в среде VisualStudio), в которые включены линейные, разветвляющиеся, циклические конструкции, а также их сочетания. Также предполагается изучение простых и составных типов данных, основных принципов работы с оперативной памятью, динамических структур данных. В дальнейшем эта база используется для перехода к вопросам проектирования программного обеспечения (ПО) как с помощью новых техник программирования, так и с помощью вынесения поддерживающих их инструментов в отдельные языки, созданные для поддержки нового образа программирования и устранения человеческого фактора при их использовании.

При этом выделяются следующие компетенции:

- владение культурой мышления;

- способность использования математического аппарата, методологии программирования и современных компьютерных технологий для решения практических задач получения, хранения, обработки и передачи информации;
- владение современными формализованными математическими, информационно-логическими и логико-семантическими моделями и методами представления, сбора и обработки информации;
- способность реализации аналитических и технологических решений в области программного обеспечения и компьютерной обработки информации.

Следующим компонентом предлагаемой модели является *содержательный* компонент, который отражает содержание курса «Языки и методы программирования».

Необходимо выполнить отбор содержания так, чтобы была достигнута поставленная цель, соблюдены принципы и вместе с тем, студенты получили основу для дальнейшей самостоятельной работы.

«Наиболее сложной педагогической задачей является построение содержания образования по конкретной области знаний, а также выбор форм оптимального представления его на разных уровнях образования. Кроме того, установлено, что качество формирования содержания образования зависит от дидактических целей, решаемых педагогических задач, уровня компетентности педагога, специфики области знаний, потенциальных возможностей и индивидуально-типологических особенностей студентов, а также материально-технической оснащенности образовательного учреждения» [101].

По мнению Б.С. Гершунского содержание образования должно быть: ориентированным на профессиональную модель специалиста, рассчитано на ближайшую перспективу [16].

В методической системе формирования программистского стиля мышления будущих учителей информатики принцип вертикальности предполагает выстраивание предметных информационных дисциплин бакалавров и магистров педагогического образования профилей «Информатика», «Математика и информатика», «Физики и информатика», магистерской программы «Информатика в образовании» в непрерывную, последовательную вертикаль, целенаправленно формирующую программистский стиль мышления.

Первой дисциплиной в этой вертикали будет являться вводный курс «Информатика», изучаемый бакалаврами педагогического образования профилей «Информатика», «Математика и информатика», «Физики и информатика» на 1 курсе. В рамках этого курса осуществляется актуализация школьного курса информатики, обзорно даются основные направления развития современной информатики, которые при дальнейшем обучении будут подробно изучаться в соответствующих курсах. С точки зрения формирования программистского стиля мышления этот курс имеет целью выравнивание уровня сформированности императивной составляющей программистского стиля мышления, подготовку к освоению курса «Языки и методы программирования».

Основной курс, при изучении которого формируется программистский стиль мышления – это курс «Языки и методы программирования», изучаемый бакалаврами на 2 и 3 курсах. Данный курс предусматривает изучение всех современных парадигм программирования, основных языков, реализующих эти парадигмы.

Особенностью предлагаемой системы формирования программистского стиля мышления является то, что в ней различные парадигмы программирования изучаются, используя полиязыковой подход к обучению.

Мы не привносим в курс нового содержания, а выделяем, переструктурируем и адаптируем тот материал, в рамках которого можно наиболее эффективным способом достичь поставленной цели. Проводится предварительный анализ содержания учебного материала, с целью выявления тех тем, которые целесообразно изучать параллельно, необходимо изучить предварительно, для успешной организации параллельного процесса, будут изучаться после (либо будут вкраплены в процесс параллельного изучения), это те темы, которые являются специфичными именно для определенной парадигмы, темы, которые невозможно запараллелить.

Возражением оппонентов может быть то, что при параллельном изучении в голове студентов может возникнуть «каша», различные парадигмы и языки невообразимо перемешаются. Опыт показывает, что, напротив, при условии тщательного содержательного и методического планирования параллельное изучение различных парадигм позволяет проводить их сравнительный анализ, эффективность и оптимальность применения их для тех или иных классов задач, а, значит, процесс обучения происходит не формально, механически, а аналитически, вдумчиво, что обеспечивает совершенно новый уровень понимания материала. Кроме того, параллельное изучение парадигм подготавливает студентов к формированию в последующем, при изучении курса «Параллельное программирование», параллельного стиля мышления. Действительно, синтаксис языков, относящихся к различным парадигмам, приходится изучать и удерживать в голове параллельно! Методике преподавания курса «Параллельное программирование» посвящено исследование М.А. Сокольской [104].

Понятно, что если полученные знания не применяются, то они быстро забываются, если только что сформированные стили мышления не будут использоваться, то они не получат дальнейшего развития.

Поэтому предлагается в дальнейшем, при проведении лабораторных практикумов по дисциплинам предметного цикла, предполагающим

дальнейшее использование навыков программирования, таким как «Численные методы», «Компьютерное моделирование», «Исследование операций», «Основы искусственного интеллекта» и пр., изучаемых на старших курса бакалавриата и в магистратуре, использовать все существующие парадигмы (таблица 3).

Таблица 3. Вертикальный формат системы формирования программистского стиля мышления

Период обучения	Учебная дисциплина	Этапы формирования программистского стиля мышления
Бакалавриат, 1 курс	Информатика ↓	Выравнивание уровня сформированности импе-ративного стиля мышления
Бакалавриат, 2-3 курс	Языки и методы программирования ↓	Расширение программистского стиля мышления. Параллельное формирование объектного, логи-ческого, функционального стилей мышления
Бакалавриат, 3 курс	Параллельное программирование ↓	Формирование параллельного стиля мыш-ления.
Бакалавриат, 3-4 курсы	Численные методы, Компьютерное моделирование ↓	Развитие программистского стиля мышления (проведение лабораторных практикумов по конкретным дисциплинам с исполь-зованием всех парадигм программирования)
Магистратура	Разработка интеллектуальных обучающих систем (Основы искусственного интеллекта)	

На занятиях присутствуют студенты с разным уровнем подготовки по программированию и разным уровнем ввосприятии и усвоении учебного материала. Некоторыестуденты отстают из-за пробелов в знаниях, или из-за

неумения организовать своюучебную деятельность. Играют роль и индивидуальные психические особенности личности.

Как мы уже говорили в параграфе 1.2, в зависимости от особенностей восприятия и обработки информации людей разделяют на три категории: визуалы, аудиалы, кинестетики. Условно выделяют еще одну категорию– дигиталы, которые оперируют словами и числами. Из них наиболее распространенныйтип – это кинестетики (40 %), визуалы – около 30 % , дигиталы – 20 % , аудиалы – 10 %.

Современная система образования ориентирована преимущественно на аудиальную, визуальную и логическую стороны мышления, то есть подразумевает передачу информации через аудиальные (лекция, устное объяснение), визуальные (презентации, наглядные пособия, фильмы), дигитальные (схемы, символы, цифры) каналы. При этом органы чувств, связанные с кинестетическими каналами восприятия (осязание, обоняние, ассоциации, выполняемые действия) практически не используются.

Поэтому для каждого студента следует строить свою траекторию обучения в зависимости от особенностей восприятия и обработки информации. Такая организация обучения возможна с привлечением всего арсенала средств обучения: устного объяснения преподавателя, компьютерного сопровождения лекционных занятий, наглядных средств (моделей, макетов, плакатов), использование кинестетических тренажеров, полиязыковых средств самостоятельной деятельности при выполнении практико-ориентированных заданий, что дает возможность активизации всех каналов восприятия информации: аудиального, визуального, дигитального и кинестетического.

Перед тем как построить студентам свою траекторию обучения на первом занятии проводиться входное тестирования для определения особенностей восприятия и обработки информации, т.е. определяем кто из студентов является: визуалом, аудиалом, кинестетиком с помощью

диагностики доминирующей перцептивной модальности С. Ефремцева на тип восприятия: аудиальный, визуальный, кинестетический (Приложение 1) [21].

Второе тестирование для определения уровня сформированности программистского стиля мышления студентов.

После проведение диагностики при обучении программированию можно условно разбить учащихся на три группы.

Процессуальный компонент представлен в соответствии с целью, принципами, содержанием обучения. Так как программистский стиль мышления формируется в учебной деятельности, то необходима такая организация этой деятельности, при которой формирование программистского стиля мышления будет происходить наиболее эффективно. Эффективность, в свою очередь, будет обеспечиваться выбором наиболее оптимальных методов, форм и средств обучения, форм контроля.

Процесс формирования программистского стиля мышления студентов подразумевает различные формы деятельности. А.С. Шаров выделяет трибазовых формы деятельности в процессе изучения любой учебной дисциплины: знаковую, моделирующую и проективную [114]. Под знаковой формой деятельности автор понимает усвоение студентами содержания знаний как системы понятий. Активность студентов при такой форме деятельности проявляется, в основном, в процессах восприятия и запоминания. Чтобы студент умел применять полученные знания на практике, он должен осуществлять какую-либо деятельность. В качестве такой деятельности выступает деятельность моделирования и проективная деятельность [114]. Таким образом, основываясь на выделенных в исследовании А.С. Шарова формах деятельности, в нашей работе мы предлагаем следующие этапы формирования компонентов программистского стиля мышления учащихся: пропедевтический, основной и закрепляющий.

На каждом этапе задаются учебные цели, определяются методы, формы и средства обучения, а также формы контроля.

Таким образом, в соответствии с целью, дидактическими принципами и на основе содержательного и процессуального компонентов модели были сформулированы следующие *педагогические условия*, выполнение которых обеспечивает эффективное формирование компонентовпрограммистского стиля мышления учащихсяна материале курса «Языки и методы программирования»:

1.Формирование содержания обучения на основе интеграции различных парадигм программирования

2. Использование специально разработанного системы практико-ориентированных заданий

3. Применение в учебном процессе натурных средств, кинестетических тренажеров, полиязыковых средств, способствующих формированию компонентов программистского мышления учащихся.

Результативный компонент модели подразумевает итоговый контроль и оценку результатов – уровня сформированности компонентов программистского стиля мышления учащихся. Контроль подразумевает обратную связь и, в случае необходимости, коррекцию методов, форм, средств обучения, форм контроля в учебном процессе.

2.2. Комплекс натурных, кинестетических и полиязыковых средств обучения как средства реализации телесно-полиязыкового подхода

Для того чтобы повысить уровень сформированности программистского стиля мышления обучающихся в процессе обучения программированию согласно телесно-полиязыкового подхода, были разработаны комплекс натурных, кинестетических и полиязыковых средств обучения по курсу «Языки и методы программирования».

Следует отметить, комплекс натурных, кинестетических и полиязыковых средств обучения ориентированы на преодоление сложностей, возникающих у обучающихся при изучении сложных тем курса.

Сегодня существует множество средств обучения информатике, традиционно многие учителя информатики используют при объяснении компьютерные презентации, обеспечивающие различную степень визуализации учебного материала.

Существуют обучающие видеоролики, иллюстрирующие выполнения различных алгоритмов, алгоритмов сортировки, например. Все более широко начинают использоваться ментальные карты и схемы [6; 30].

Эффективным средствами изучения алгоритмизации являются программа «Комплект Учебных МИРов», в котором используются встроенные исполнители – «Кузнечик», «Черепашка», «Водолей», «Робот» и другие. Все эти средства облегчают понимание, они собственно являются теми недостающими звеньям перехода от житейского, повседневного алгоритмического мышления, имеющегося у каждого человека к абстракциям блок-схем и программных кодов, но они в основном нацелены на учащихся с визуальным, аудиальным и дигитальным восприятием, поскольку обращаются именно к этим каналам восприятия. При этом органы чувств, связанные с кинестетическими каналами восприятия остаются незадействованы. Между тем, согласно известной статистике, примерно у 40% людей ведущими являются именно кинестетические каналы восприятия. Эти люди чувствуют окружающий мир и воспринимают большую часть информации чувственно, через осязание, обоняние, ассоциации, выполняемые действия. Кинестетику, чтобы понять окончательно, необходима деятельность, поделать что-нибудь самому, руками, ощутить на ощупь [108]. Получая информацию, воспринимая окружающую действительность, кинестетик стремится всё перевести на язык телесных ощущений, вкуса, осязания и обоняния, стараясь дотронуться до

собеседника, даже если не смотрит на него, используя как основной инструмент для обработки информации интуицию, телесные, чувственные ассоциации, память о выполнении некоторых действий, активизирующих моторную область памяти, а не модельную как визуалы, или понятийную, как аудиалы или абстрактную, как дигиталы [9].

Использование в учебном процессе по школьному курсу информатики исполнителей тип Кумир, Черепашка, конечно же, активизирует кинестетические каналы восприятия, поскольку учащийся осуществляет сам непосредственно алгоритмическую деятельность, управляя исполнителем, и видит ее результаты, но лишь частично, поскольку эти результаты визуализированы, это компьютерные, виртуальные исполнители, тактильные, моторные ощущения они задействуют в невысокой степени.

Натурные средства обучения позволяют раскрыть изучаемые понятия или алгоритмы на тактильном уровне, погрузить учащегося в процесс прохождения алгоритма или рассмотреть изучаемый объект с разных позиций. Иначе говоря, используя данные средства обучения, учащийся может сначала «вручную» разобрать смысл изучаемого процесса или понятия, а затем на основе выявленных действий построить алгоритм, создать информационную модель или выполнить другое абстрактное действие.

Натурные средства в рамках телесного подхода в обучении должны реализовать основные принципы дидактики: принцип наглядности, доступности, принцип сознательности и активности, т.к. обучающийся сам будет руководить процессом изучения понятия или метода дисциплины, манипулируя объектами в реальном времени и пространстве.

При этом с точки зрения подготовки будущего учителя информатики интерес вызывает самостоятельная разработка данных натурных средств самими обучающимися. Это позволит усилить когнитивную активность обучающихся, исключить изучение материала на уровне «бездумного

заучивания», погрузить их в суть изучаемого материала, т.к. для создания конкретного средства им необходимо разобраться с изучаемыми методами и понятиями на уровне полного и глубокого уровня понимания. Таким образом, обучение будет осуществляться в действии и через действия. Обучающиеся сами создают натурные средства обучения и используют их в своей подготовке и обучении школьников курсу информатики.

Натурные средства обучения будут незаменимы при изучении многих разделов предметной области информатики (где это возможно), например, «Кодирование и измерение информации», «Устройство компьютера», «Алгоритмизация и программирование», «Моделирование» и другие разделы.

В рамках нашего исследования такими средствами могут быть:

- Работы алгоритмов с простыми типами данных;
- Работы алгоритмов с массивами и строками;
- Работы с динамическими структурами данных;
- Работа с классами и признаками объектов.

Данные средства могут позволить задействовать всю чувственную (сенсорную) систему человека в процессе обучения, активируя понимание не на уровне воображения, а на тактильном уровне, позволяя оперировать реальными объектами в реальном пространстве. Таким образом, процесс обучения будет строиться от простого к сложному, от единичных операций к множественным, от реальных объектов к абстрактным.

Рассмотрим тему «Циклы». Данная тема является одной из основных тем в изучении программирования. При обучении этой теме, преподавателю приходится сталкиваться с такими проблемами, как:

- неверная трактовка блок-схем, которые учитель предоставляет в качестве визуального примера.

- непонимание учениками механизма организации повторений;

- неправильный подсчет количества повторений;

- затруднения при определении границ и правила изменения счетчика цикла;

- зачастую ученики не могут корректно определить условие окончания цикла, либо не понимают, что в теле цикла значение переменной, присутствующей в этом условии, обязательно должно меняться, вследствие чего получают «зацикливание» программы;

- допускают ошибки при формировании заголовка либо вообще забывают написать его;

- забывают о том, что если в теле цикла более одного оператора, нужно использовать операторные скобки и т.п.

Для решения этих проблем разработаны анимированные презентации, наглядно демонстрирующие работу циклов. Основная идея – поэтапное выполнение алгоритма – наглядно демонстрируется в ролике и может служить основой для написания алгоритма (рис.9).

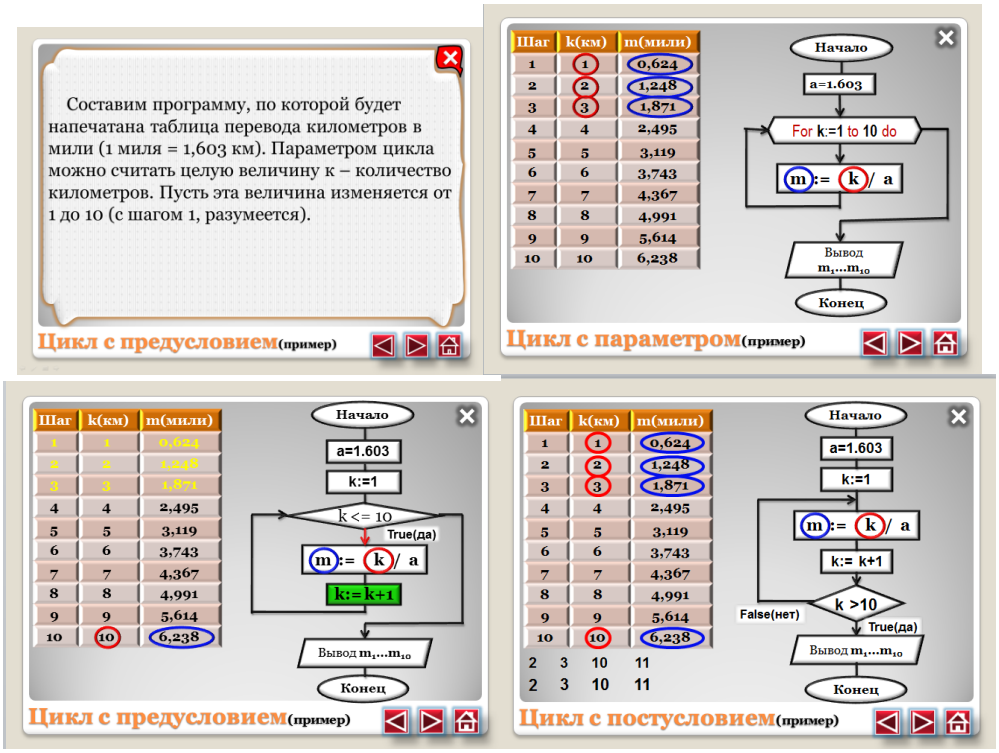


Рис.9. Анимированные презентации выполнения алгоритмов – цикла с параметром, цикла с предусловием, цикла с постусловием

Аналогичные анимированные презентации разработаны по темам «Массивы», «Динамические переменные».

Рассмотрим натурное средство при изучении темы «Классы и признаки объектов». Студентами был предложен комплект, состоящий из объектов с разными признаками (например, кубики и шары желтого и зеленого цвета), а также форма для разделения объектов по классам (рис.10).

На первом этапе предлагается учащимся просто разделить объекты на два класса: класс желтых и класс круглых. Для упрощения задачи лучше, чтобы все кубики были желтого цвета, тогда все объекты можно разделить на два класса. В каждом классе необходимо посчитать количество объектов и ответить на вопрос, можно ли по количеству объектов в каждом классе посчитать общее количество объектов? С помощью рассуждений можно перейти к объяснению формулы включений и исключений.

Учащиеся также раскладывают объекты в форме, разделяя их на объекты с разными признаками и объекты с наличием признаков двух классов (пересечение классов) [7].

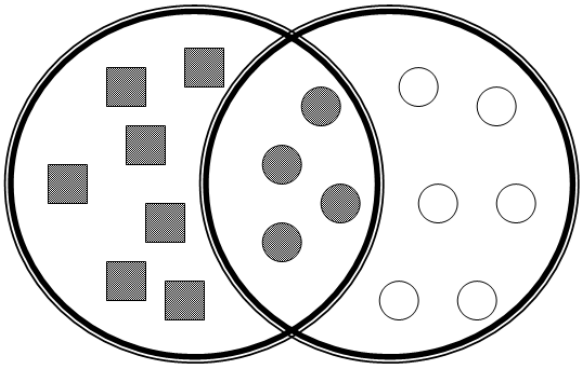


Рис. 10. Классы и признаки объектов

Кроме этого, студентами был предложен ряд натурных средств – кинестетических тренажеров, позволяющих наглядно разобрать работу конкретного алгоритма, например, сортировку массива, поиск максимального или минимального элемента и т.п.

Создание новых средств формирования программистского стиля мышления, которые нацелены на кинестетические каналы восприятия и активизацию моторной области памяти являются актуальными не только для учащихся с преобладающим кинестетическим типом восприятия, поскольку, как известно деление по типам восприятия достаточно условно редко встречаются чисто выраженные типы, чаще смешанные. Поэтому активизация всех каналов восприятия при изложении учебного материала позволит существенно повысить его уровень понимания.

А согласно положениям нового течения в психологии как телесный подход, эти наши ощущения играют немаловажную роль в формировании мышления вообще, и, если мы говорим о возможности «подержать в руках», «осязать» процесс алгоритмической деятельности, алгоритмического мышления в частности.

Эти основные положения теории телесного подхода, дают теоретическое обоснование необходимости создания средств развития алгоритмического мышления, средств обучения алгоритмизации и программированию, учитывающих взаимосвязь телесных, кинестетических ощущений и восприятия и развития мышления.

Мы дали этим средствам рабочее название – кинестетические тренажеры.

Кинестетический тренажер представляет собой некое приспособление, изготовленное преподавателем или самими студентами, позволяющее реализовать алгоритм решения задачи.

Мы попытались спроектировать такие тренажеры для изучения составных типов данных на примере массива, алгоритмов сортировки и поиска массивов на примере метода пузырька, а также для изучения динамических типов данных на примере стека и очереди.

Если изучение простых типов данных дается относительно легко, то изучение составных типов данных – массивов, записей, множеств, уже

вызывает определенные затруднения. Часто возникают такие проблемы как: обучающиеся путают значение индекса элемента массива со значением самого элемента массива; или например, не понимают, что все элементы массива должны быть одного типа. Что массив должен обрабатываться поэлементно с помощью цикла, как обратиться к нужному элементу массива, не допускать выход за пределы массива. И так далее. Для решения всех этих проблем представляется эффективным применять в процессе обучения кинестетические тренажеры, которые позволят «осязать» массив, подержать его в руках, выполнить самим то, что в дальнейшем следует предписать в виде операторов языка программирования машине.

Например, для моделирования понятия переменной можно использовать специальные ящики. Имя переменной – это буква, записанная на ящике, а присваиваемое ей значение – это величина (число), помещаемое в ящик. Манипуляции с подобными ящиками при словесных записях в виде алгоритмических инструкций формируют ментальные схемы на чувственном уровне. Составление программы на языке программирования по данному алгоритму интерпретируется как перевод с языка обычного на язык программирования. При этом ящики для переменных заменяются на ячейки памяти ЭВМ.

Большие «телесные» алгоритмические возможности имеет конструктор LEGO (рис.11), в котором фигуры и конструкции собираются из различных частей-кубиков. Инструкция по сборке конкретной конструкции легко понимается как алгоритм на интуитивном уровне.

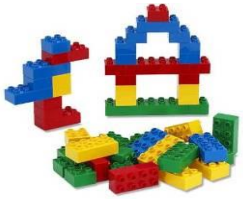


Рис.11. Конструктор LEGO

Рассмотрим проект кинестетического тренажера для изучения статического одномерного массива [2].

Выглядит этот тренажер как коробка с крышкой, имитирующая область памяти, в которой будет располагаться массив (рис.12).

Рис. 12. Кинестетический тренажер для изучения составных типов данных: статический одномерный массив

На крышке коробки (на «экране») следует написать инструкцию по работе с тренажером, внутри будет поле размещения элементов массива (ячейки памяти). В первом пункте инструкции попросить обучающегося определиться с размерностью его массива, задать количество элементов массива N. Затем предложить взять ровно столько маленьких коробочек (которые заранее заготовлены), какова размерность его массива, при этом коробочки (ячейки) заранее пронумерованы и ученик берет коробочки с цифрами от 1 до N и раскладывает их в порядке возрастания в поле для ячеек (рис.13).

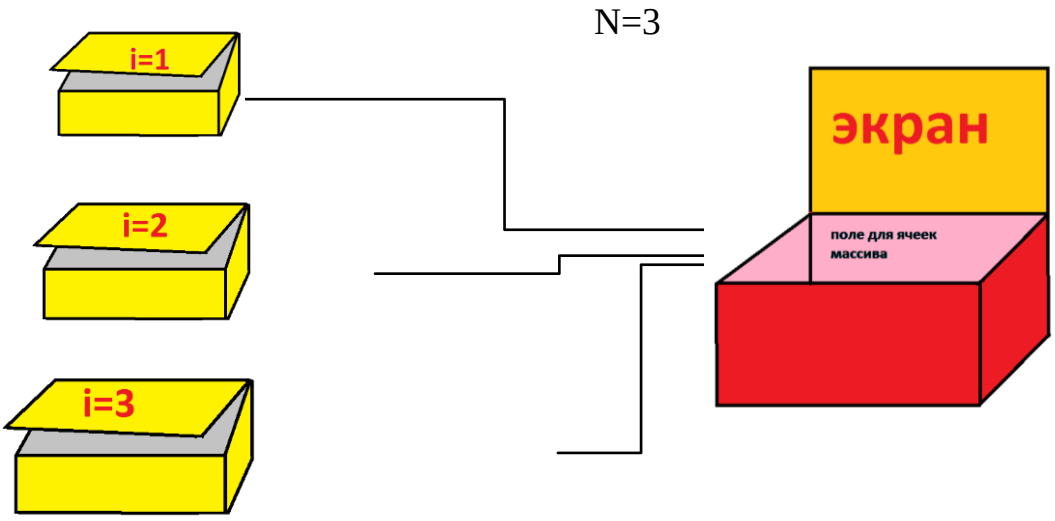


Рис.13. Размерность массива

Далее определиться с типом данных, ведь массив это набор однотипных компонентов. Данные заготовлены заранее – цифры на круглых фишках, символы на квадратных и т.д., данные одного типа нанесены на фишки одной формы, чтобы подчеркнуть их однотипность.

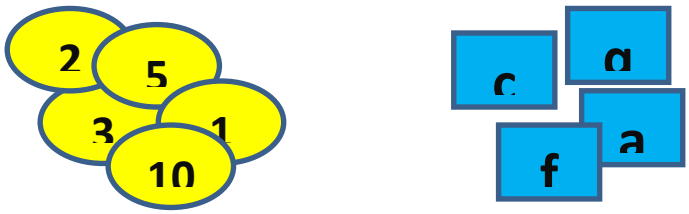


Рис.14. Тип данных массива

Созданный массив нужно заполнить элементами, заполнение осуществляется с помощью цикла, поэтому предлагаем по порядку с 1 до N открывать ячейку и класть в нее данные. Здесь важно сделать акцент на том, каким образом, будет заполняться массив: с клавиатуры либо с помощью генератора случайных чисел. Если с клавиатуры, то обучающийся берет нужное число, если с помощью генератора, то он берет число наугад. При этом параллельно демонстрируется работа генератора случайных чисел. Действия будут повторяться и здесь усвоится смысл и необходимость применения цикла для заполнения массива.

При проектировании данного тренажера существует еще множество нюансов – позволить быть открытой только одной коробочке, имитирующей элемент массива, т.е. она должна закрываться автоматически, позволить доставать из нее содержимое только пинцетом – для иллюстрации того, что доступ к элементам массива осуществляется поэлементно, что мы не можем одновременно достать и держать в руках два или более элемента массива сразу, и т.д. и т.п.

Определенные сложности возникают при изучении алгоритмов сортировки массива, даже такой простейший алгоритм, имеющий явную натурную ассоциацию как метод пузырька, не всеми обучающимися воспринимается после его словестного описания, на понятийном уровне. При

его объяснении преподаватели информатики зачастую прибегают к его наглядной реализации, например, выстраивая учеников по росту и т.п. Вообще, в своей практике преподаватели, сталкиваясь с непониманием учебного материала, часто прибегают к использованию кинестетических тренажеров – ящики комода как элементы массива, тубус для теннисных шариков как модель динамических структур данных – стека, в данном случае и т.д. То есть они интуитивно понимают необходимость задействования всех каналов восприятия и апеллирования к образной и моторной зонам памяти.

Например, кинестетический тренажер для изучения алгоритмов сортировки может выглядеть как набор бильярдных шаров, уложенных на свою подставку. (Рис.15)

Рис. 15. Кинестетический тренажер для изучения алгоритмов сортировки массива

На каждом шаре, имитирующем элемент массива, подписан номер – индекс элемента массива, причем этот номер можно стереть и написать заново, поскольку в ходе выполнения алгоритма нам придется менять элементы местами. Шары разного веса. Количество шаров N, для определенности положим N=10. Цель – упорядочить шары в порядке убывания веса. Для усиления наглядных ассоциаций, можно окрасить шары массива таким образом, что когда массив будет отсортирован, цвета сложатся в определенной последовательности – радуга или постепенное усиление тона. Имитация выполнения сортировки массива методом пузырька заключается в следующем: поскольку в соответствии с алгоритмом мы можем сравнивать только два соседние элемента массива, ученику предлагается взять в руки два рядом лежащих шара, определить какой из них

тяжелее, и если необходимо, поменять их местами, переписав на них индексы. Потом проделать эту процедуру со следующей парой и т.д. до последней пары. Нам придется выполнить 9=N-1 взвешиваний. Тут наглядно демонстрируется необходимость использования цикла, поскольку повторяется выполнение одних и тех же действий, и определяется количество повторений N-1. После этого самый легкий шар-«пузырек» станет последним – «всплывет на поверхность», но остальные шары останутся неупорядоченными. Для того чтобы, продолжить процедуру, необходимо повторить процесс с самого начала, с первой пары шаров, но выполнять сравнение веса каждой пары нам придется уже не 9, а 8= N-2 раз, поскольку последний элемент находится на своем месте – он самый легкий, и нет необходимости взвешивать и сравнивать последнюю пару. После этого второй по легкости шар займет свое место - станет предпоследним – «всплыл» следующий «пузырек» и т.д. Наглядно убедившись, что процесс сравнивания пар шаров придется начинать с начала столько раз, сколько элементов в массиве, т.е. N, ученик осознанно понимает необходимость использования конструкции из двух вложенных циклов для реализации алгоритма сортировки. Причем, поскольку в каждом следующем случае приходится сравнивать все меньшее и меньшее количество пар, ему становится понятно, почему индекс вложенного цикла (допустим j)меняется не до N, а до N-i, где i - индекс внешнего цикла, который меняется от 1 до N – столько раз, сколько приходится начинать сначала.

Практика использования этой модели показывает, что после выполнения алгоритма вручную, текст программы, реализующей этот алгоритм, становится понятен абсолютно всем обучающимся.

Подобный тренажер можно использовать и для наглядной иллюстрации более сложных алгоритмов сортировки – сортировки Хоара, битовой или разрядной сортировки, которые зачастую остаются непонятыми после первого, словестного объяснения, даже сильным ученикам.

Еще одна задача, которая вызывает трудности при составлении алгоритмов, это задача поиска. Задача поиска заключается в отыскании в последовательности элемента (или нескольких элементов) с заданными свойствами его значения. Например, найти элемент с наибольшим (наименьшим) значением или найти элемент с заданным значением.

Рассмотрим кинестетические тренажеры для более детального анализа алгоритмов поиска на конкретных задачах.

1. Найти минимальное значение элементов последовательности (все элементы разные).

2. Найти номер минимального элемента последовательности (все элементы разные)

3. Найти максимальный элемент и его номер в последовательности с совпадающими элементами.

Рассмотрим подробнее первую задачу. На рис. 16 задача 1 рассматривается как задача определения размера самого маленького яблока из лежащих в ящике, разделенном на ячейки. Сделаем из мягкой проволоки рамку размером в любое произвольное яблоко. Тем самым мы задались исходным размером; назовем рамку эталоном. Берем следующее яблоко и пытаемся пронести его через рамку. Если оно не проходит (больше эталона), то оно нас не интересует. Если же окажется, что очередное яблоко меньше эталона, то это уже претендент на наименьшее. Изменим эталон до размера этого яблока, подкрутив проволоку, и продолжаем сравнение. Когда мы, таким образом, пропустим все яблоки через рамку, ее размер и будет размером наименьшего.

Аналогично можно найти наибольшее яблоко, но только тогда рамку нужно каждый раз увеличивать до размеров большего яблока. Распространим эту идею на поиск минимального элемента в последовательности. Поиск будем проводить путём сравнения всех элементов с эталоном-переменной,

которой заранее присвоим значение какого-то элемента (о том, какой элемент лучше выбрать для этой цели, скажем позже).

Пусть эталонная переменная определена. Сравним с нею первый элемент. Если окажется, что он меньше эталона, то изменим эталон (присвоим ему значение первого элемента), затем перейдем к сравнению его со вторым элементом; в противном случае сразу перейдем к сравнению эталона со вторым элементом. Подобные сравнения проведем для всех элементов последовательности. По завершении просмотра эталонная переменная будет иметь значение, равное минимальному значению элементов последовательности.

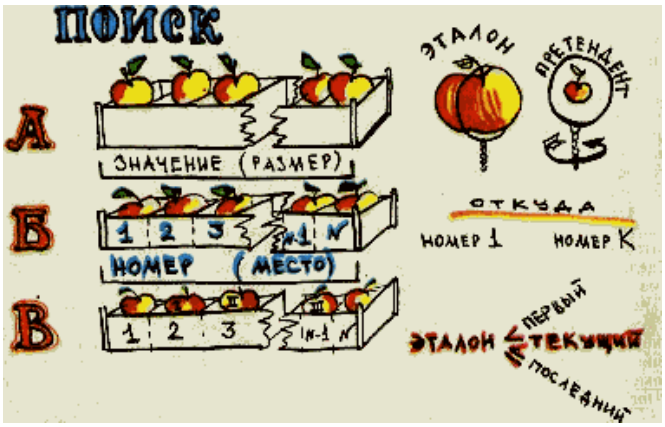


Рис.16. Кинестетические тренажеры для более детального анализа алгоритмов поиска

Также один из сложных для понимания моментов – принципы работы с динамическими типами данных. Отчасти эти сложности обусловлены тем, что при работе со статическими типами данных – допустим, с массивами, у обучающихся возникают определенные стереотипы. Например, что по индексу можно обратиться напрямую к любому элементу, чего нельзя сделать при работе с такими динамическими типами данных как стеки и очереди. Кинестетический тренажер для изучения этих типов данных может выглядеть как тубус, в котором хранятся мячи для игры в теннис (Рис. 17).

Рис. 17. Кинестетический тренажер для изучения динамического типа данных стек.

Стек – это однонаправленный список, в котором все включения и удаления выполняются с конца списка. Он работает по принципу LIFO (last-in, first-out; «последним пришел – первым вышел»). Его моделью может быть такой тубус для мячей, который открывается только с одного конца. Хорошо, если мячи будут разного цвета, и ученики будут наглядно видеть, что если мы первым опустили в тубус красный мячик, а затем желтый, зеленый и др., то для того, чтобы достать красный (и почитать, что на нем написано, например), нужно выкатить все шары, которые были помещены в стек позже.

Очередь – это однонаправленный список, в котором удаление элементов происходит из начала, а включение новых элементов – в конце. В этом случае работает принцип FIFO (first -in, first-out; «первым пришел – первым вышел»). Его моделью может быть такой тубус для мячей, который открывается сдвух сторон, но с каждой стороны установлены анти клапаны, работающие в только в одном направлении – позволяющие только выкатывать мячи из тубуса в начале и только закатывать мячи в тубус в конце.

Еще один пример тренажера для изучения динамических структур данных. В курсе программирования в вузе данная тема является одной из самых трудных для студентов. Проблемы возникают при понимании того, как организован этот тип данных, как выделяется память для его хранения, что такое указатель и как с ним работать и др. [113].

Чтобы показать принцип работы тренажера, мы использовали динамическую структуру данных стек.

Для имитации динамического выделения памяти мы решили использовать кармашки, нашитые на ткань или другое более плотное полотно. Это необходимо, чтобы информация, помещенная в кармашек, становилась невидима для глаз ученика. Все кармашки пронумерованы различными адресами (рис. 18).

Указатели мы представили в виде карточек. Их размер выбран так, чтобы они были видны, т.е. часть карточки выглядывает из кармашка. Нам необходимо два указателя: head (определяет вершину стека) и tmp (вспомогательный указатель).

Различные области памяти узла стека мы представили в виде карточек разного цвета. Данные карточки должны полностью помещаться в кармашки.

К тренажеру необходимо приложить инструкцию.

В первом пункте инструкции попросить ученика определиться с типами данных, которые будут располагаться в каждом узле стека. За каждым типом данных будет зарезервирован свой цвет карточки. Белые карточки будут обозначать указатели next, которые отвечают за связь узлов стека между собой.

Во втором пункте написать часть кода программы, которая отвечает за формирование стека и предложить повторить данный код при помощи тренажера.

Выделение памяти для указателя tmp и заполнение узла стека данными показано на рис.16:

a) stek *head = NULL – указатель head объединяется с карточкой null;

b) stek *tmp – в работу вступает указатель tmp;

c) tmp = newstek – указатель tmp помещается в произвольный кармашек (1).

d) Соответствующие карточки заполняются информацией выбранного типа данных и помещаются в кармашек (2).

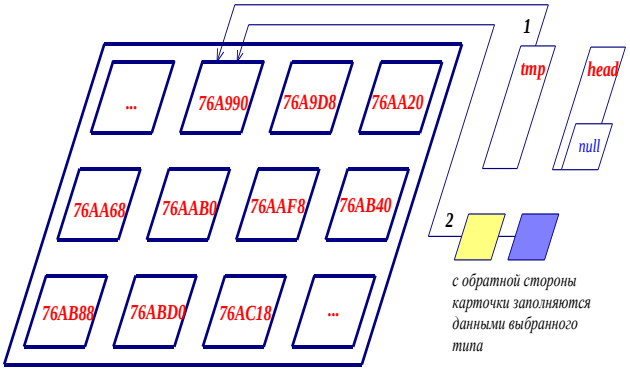


Рис. 18. Выделение памяти для указателя tmp и заполнение узла стека данными

В третьем пункте можно попросить просмотреть содержимое стека и найти его определенный элемент при помощи кода программы. Затем в четвертом пункте необходимо будет удалить стек.

Данный тренажер можно использовать и для таких динамических структур, как очереди и деки. Для этого необходимо будет просто заготовить дополнительные указатели.

Представляется, что использование в учебном процессе подобных кинестетических тренажеров позволит повысить уровень понимания и усвоения учебного материала по алгоритмизации и программированию, будет способствовать процессу развития программистского стиля мышления.

Помимо выше описанных натурных средств и кинестетических тренажеров, в процессе обучения используется полиязыковой подход к обучению.

Практически во всех исследованиях по методике обучения программированию предлагается изучать различные парадигмы последовательно, и исследователи спорят лишь об оптимальной последовательности их изучения. При проектировании методической системы обучения курсу «Языки и методы программирования» целесообразно предусмотреть *параллельное изучение парадигм*

программирования с помощью системы практико-ориентированных задач. Студентам предлагается решить конкретную задачу средствами языков программирования, относящихся к различным парадигмам.

Подобный *полиязыковый подход* позволяет студенту критически анализировать стили и парадигмы программирования.

Пример вычисления факториала представлен на рисунке 19.

Императивная парадигма (Pascal)	Логическая парадигма (Prolog)	Функциональная парадигма (Lisp)

program XXX;
var F, i: integer;
begin
i:=0;
F:=1;
writeln('Введите n');
readln (n);
while i<n do
begin
Inc(i);
F:=F*i;
end;
writeln(' ', n, '=, F);
end.
	Predicates
	Fact(integer, real)
	clauses
	Fact(0, 1):-!.
	Fact(X, FactX):-Y=X-1,
	Fact(Y, FactY),
	FactX=X Fact Y.
		(defun fakt (n) “Вычисление факториала”
		(cond
		((=n 1) 1)
		(T (*n (fakt (– n 1))))
		))

Рис. 19. Пример вычисления факториала

Пример ввода данных для решения системы линейных уравнений методом прогонки на языке Pascal и Delphi показан на рисунке 20.

Процедурная парадигма (Pascal)	Объектно-ориентированная парадигма (Delphi)

writeln('Vvediteelementiglavnoidiagonali: ');
for i:=1 to 4 do
begin
read(b[i]);
end;
writeln('Vvediteelementi a: ');
for i:=2 to 4 do
begin
read(a[i]);
end;
writeln('Vvediteelementi c: ');
for i:=1 to 3 do
begin
read(c[i]);
end;
writeln('Vvediterezyltati: ');
for i:=1 to 4 do
begin
read(f[i]);
end;
	procedure TForm1.Button1Click(Sender: TObject);
	begin {вводим элементы главной диагонали}
	b[1]:=strtoint(edit1.Text);
	b[2]:=strtoint(edit2.Text);
	b[3]:=strtoint(edit3.Text);
	b[4]:=strtoint(edit4.Text);
	{вводим элементы, стоящие под главной диагональю}
	a[2]:=strtoint(edit5.Text);
	a[3]:=strtoint(edit6.Text);
	a[4]:=strtoint(edit7.Text);
	{вводим элементы, стоящие над главной диагональю}
	c[1]:=strtoint(edit8.Text);
	c[2]:=strtoint(edit9.Text);
	c[3]:=strtoint(edit10.Text);
	{вводимрезультаты}
	f[1]:=strtoint(edit11.Text);
	f[2]:=strtoint(edit12.Text);
	f[3]:=strtoint(edit13.Text);
	f[4]:=strtoint(edit14.Text);

Рис. 20. Пример ввода данных для решения системы линейных уравнений методом прогонки

Каждый из рассмотренных нами языков, принадлежащих к различным парадигмам программирования, обладает своими достоинствами и недостатками. Программная реализация одной и той же задачи на различных языках позволит студентам выявить эти самые преимущества парадигм, либо, наоборот, их недостатки, сравнить на практике эффективность каждой из парадигм. Таким образом, после того, как они проведут сравнительный анализ реализации одно и того же алгоритма на различных языках, у студентов будет формироваться умение выбрать парадигмы программирования, исходя из специфики поставленной задачи. Результаты сравнительного анализа могут быть оформлены в виде таблице как представлено в приложении 3.

2.3. Проверка результативности разработанной методики телесно-полиязыкового обучения студентов в курсе «Языки и методы программирования»

Для проверки эффективности методической системы обучения будущих учителей информатики программирования с 2009 по настоящее время на базе отделения информатики Института математики, физики и информатики Красноярского государственного педагогического университета им. В.П. Астафьева был проведен частичный *педагогический эксперимент*.

Цель педагогического эксперимента заключалась в проверке достоверности выдвинутой гипотезы, а также в оценке влияния разработанной методики обучения программированию будущих учителей информатики на формирование программистского стиля мышления на основе телесно-полиязыкового подхода.

Экспериментальная работа выполнялась в условиях целостного педагогического процесса университета и состояла из трех этапов.

На первом этапе эксперимента – поисковом (2009 – 2011 годы) – были решены следующие задачи:

1. Изучена предметная область исследования, выделены её составляющие и решаемые задачи. Определена проблема исследования и выявлена её актуальность.
2. Изучена педагогическая и методическая литература по проблеме исследования.
3. Изучены требования федерального государственного образовательного стандарта, структура предметной компетенции, определены роль и место изучения программированию в системе предметной подготовки педагогов-бакалавров профилей «Информатика», «Математика и информатика», «Физика и информатика».
4. Выполнено предварительное построение содержания обучения.
5. Выбраны подходы к обучению.

На данном этапе использовались такие методы исследования как анализ, обобщение, систематизация, а также наблюдение, беседа, анкетирование, изучение педагогического опыта.

Для решения первой задачи была изучена доступная литература, определены технические возможности факультета (в настоящее время отделения) информатики, которые необходимы для обеспечения учебного процесса по изучению программирования.

Для решения второй задачи изучались современные подходы к построению методических систем, ведущие подходы к обучению, анализировалась психологическая литература и положения педагогической психологии, касающиеся развития мышления в обучении, формирования мыслительных действий, возможностей воздействия на мыслительные процессы обучающихся. Данный анализ позволил выделить ряд положений и подходов, позволивших конкретизировать возможные изменения в мыслительном процессе под воздействием практики.

Третья задача заключалась в анализе официальных документов, программ обучения бакалавров профилей «Информатика», «Математика и информатика», «Физика и информатика», определения места, способов включения в системе предметной подготовки учителей информатики. В результате анализа были выявлены межпредметные связи элементов программирования с дисциплинами в профессиональной подготовке учителя информатики.

В ходе решения четвертой задачи изучена научно-методическая литература, касающаяся проблематики отбора содержания учебных дисциплин, выбран подход к отбору содержания модуля, выстроена модель содержания согласно ведущему подходу.

В рамках пятой задачи нами был выбран телесно-полиязыковой подход к обучению как системообразующий для формирования «навыков» мыслительной деятельности, разработана основная структура методической системы.

Второй этап – *констатирующий*- педагогического эксперимента проводился в 2012-2015 гг. Во время проведения этого этапа:

– уточнены теоретические положения и подходы к построению методической системы обучения педагогов-бакалавров профиля информатика курсу программирования.

– построена модель методической системы обучения программированию

– окончательно определено содержание курса

– выстроена система методов и приемов обучения

– построена модель программистского стиля мышления и выдвинуты предположения о способах его формирования на основе телесно-полиязыкового подхода, разработана диагностика, определяющая уровень его сформированности.

Методика телесно-полиязыкового обучения введена в программу курса «Информатика» на 1 курсе, в программу «Языки и методы программирования» на 2-3 курсах для бакалавров педагогического образования, профиль «Информатика», «Математика и информатика», «Физика и информатика», также в курс по выбору «Теория алгоритмического мышления» для магистрантов педагогического образования обучающихся по магистерской программе «Информатика в образовании» в ИМФИ КГПУ им. В.П. Астафьева.

Перед началом изучения курса проведено тестирование, определяющее особенности восприятия и обработки информации студентов, и тестирование определяющее уровень сформированности программистского стиля мышления.

Также проведен опрос и анкетирование с целью определения представлений у обучающихся о парадигмах программирования до начала обучения и как они видят роль парадигм программирования и их значимость после изучения.

По окончанию изучения курса «Языки и методы программирования» проводится контроль знаний, при помощи тестов содержащий задания по программированию. С помощью этого контроля выявляется уровень усвоения знаний по предмету. Сопоставляется результаты диагностики уровня сформированности программистского стиля мышления и уровня усвоения предметных знания, с целью влияния уровня сформированности программистского стиля мышления на успешность обучения программирования.

Третий этап – *формирующий* – педагогического эксперимента проводится с 2016 гг. и направлен на коррекцию содержания курса «Языки и методы программирования», корректировку и доработку методической системы обучения, проверку эффективности методической системы обучения

программирования будущих учителей информатики на основе телесно-полиязыкового подхода, сравнение в динамике результатов обучения.

Выводы по второй главе

Во второй главе на основе телесного и полиязыкового подходов к обучению построена структурно-логическая модель методики обучения студентов программированию, включающая систему целей, содержание, систему методов и приемов обучения с применением средств (кинестетические тренажеры) и методов (параллельное изучение языков программирования) телесного и полиязыкового подходов, комплекс натурных, кинестетических и полиязыковых средств алгоритмических конструкций, контрольно-измерительных материалов.

Заключение

В заключение магистерской диссертации приведены основные результаты проведенного исследования.

1. Анализ современного состояния, информационного, наукоёмкого общество и системы высшего профессионального образования показал, что общество нуждается в людях с высоким уровнем развития абстрактного мышления вообще и алгоритмического мышления в частности, поскольку осуществление абсолютно любой деятельности – не только профессиональной, где это является необходимым условием в настоящее время практически во всех профессиях, но и бытовой, повседневной, предполагает предварительное моделирование и планирование этой деятельности, накопление и анализ информации, необходимой для её осуществления, т.е. составление алгоритма действий.

2. На основе пространственно-временной модели памяти и положений психологии построена модель алгоритмического мышления, уточнена и выявлена сущность понятия «программистский стиль мышления».

3. На основании дополнительных этапов и способов мыслительной деятельности выделены критерии, которые в совокупности могут охарактеризовать уровень сформированности программистского стиля мышления.

4. На основе телесного полиязыкового подходов к обучению построена модель методики обучения студентов программированию, включающая систему целей, содержание, систему методов и приемов обучения с применением средств (кинестетические тренажеры) и методов (параллельное изучение языков программирования) телесного и полиязыкового подходов, комплекс натурных, кинестетических и

полиязыковых средств алгоритмических конструкций, контрольно-измерительных материалов.

5. На основе структурно-логической модели разработана методика обучения будущих учителей информатики курсу «Языки иметоды программирования»

Практическая значимость исследования заключается в том, что:

- разработан комплекс алгоритмических натурных, кинестетических иполиязыковых средств обучения студентов программированию;

- опубликован в соавторстве с группой авторов (Нигматулина Э.А., ПакН.И., Сокольская М.А., Степанова Т.А.) учебник «Программирование» в2-х томах для бакалавров педагогического образования профиль «информатика» (М.: Академия, гриф УМО по направлению«Педагогическое образования»), отражающий концепцию полиязыковогоизучения программирования.

- разработанная методика телесно-полиязыкового обучения студентов программированию может быть использована в учебном процессе впедагогических вузах, на курсах ПК.

Таким образом, цель исследования достигнута, поставленные задачи решены.

Результаты магистерского исследования используются в учебном процессе:

1) бюджетного общеобразовательного учреждения «Кириковская средняя школа». В частности: в образовательный процесс при изучении отдельных тем по программированию внедрены натурные и кинестетические тренажеры, частично использована методика полиязыкового обучения с использованием системы практико-ориентированных заданий.

2) Лесосибирского педагогического института – филиала СФУ. В частности: частично апробирован полиязыковой подход в обучении

дисциплине «Программирование» по направлению подготовки 44.03.01 «Педагогическое образование» профиль 44.03.01.33 «Информатика».

Направления дальнейших исследований по теме диссертации видятся в

1. Оформлении разработанных кинестетических тренажеров в виде полностью завершенного комплекта натурных средств обучения, охватывающих все темы курса «Языки и методы программирования», а также школьного курса информатики.
2. Дальнейшей детализации методики полиязыкового подхода к обучению программированию.
3. Совершенствовании системы диагностики уровня сформированности программистского стиля мышления.

Библиографический список

1. АбдуразаковМ.М. Совершенствование содержания подготовки будущего учителя информатики в условиях информатизации образования: Автореферат диссертации на соискание степени доктора педагогических наук URL:http://www.phido.ru/Disser/8242/View.aspx (дата обращения: 18.05.2011).
2. Алексеева Д.В., Степанова Т.А., Хайбрахманова А.Т. Кинестетические тренажеры как средства развития алгоритмического мышления // Материалы XIV Всероссийской (с международным участием) научно-практической конференции студентов, аспирантов и молодых ученых «Молодежь и наука» – Красноярск, 2014.
3. Алферьева Т.И. Формирование алгоритмической культуры при изучении математических дисциплин. ww.dusk12.ru/teoria/alf.doc
4. Алюшин А.Л., Князева Е.Н. Телесный подход в когнитивной науке // Философские науки, 2009. № 2. С. 106-125.
5. Баженова И.В. От проективно-рекурсивной технологии обучения к ментальной дидактике: монография / И.В. Баженова, Н. Бабич, Н.И. Пак. – Красноярск: Сиб. федер. ун-т, 2016. – 160с.
6. Баженова И.В., Степанова Т.А. Использование методики ментальных карт при обучении программированию в высшей школе // Материалы II Всероссийской научно-практической конференции с международным участием «Перспективы и вызовы информационного общества», 2013. С. 173.
7. Бархатова Д.А. Натурные средства обучения в подготовке будущего учителя информатики // Перспективы и вызовы информационного общества: материалы IV Всероссийской научно-практической конференции с международным участием. Красноярск, 12 ноября 2015 г. [Электронное ресурс] / КГПУ им. В.П. Астафьева. – Красноярск, 2015. С. 97-104.

8. Беляев М.В. Алгоритмическое мышление как цель современного образования //Сборник материалов Международная региональная конференция ЮНЕСКО «Экология человека: взаимодействие культуры и образования в современных условиях».ч.1, Новосибирск, 1998. С.

9. Борисёнок И. Аудиалы, визуалы, кинестетики и дигиталы. [Электронный ресурс] URL:http://vseklass.ru/obshhenie/audialy-vizualy-kinestetiki-i-digitaly.html (26.10.2015г).

10. Выготский Л.С. Психология развития как феномен культуры / под редакцией М.Г. Ярошевского.- М.: Издательство «Институт практической психологии», Воронеж: НПО «МОДЭК», 1996.- 512 с.

11. Газейкина А.И. Стили мышления и обучение программированию студентов педагогического вуза. URL: http://ito.edu.ru/2006/moscow(дата обращения: 14.12.2010).

12. Газейкина А.И. Стили мышления и обучение программированию // Информационные технологии в общеобразовательной школе. – 2006. - № 6. – С.12-19.

13. Гальперин П.Я. Психология как объективная наукапод редакцией А.И. Подольского.- М.: Издательство «Институт практической психологии», Воронеж: НПО «МОДЭК», 1998.- 480 с.

14. Гальперин П.Я. Психология мышления и учение о поэтапном формировании умственных действий. - Исследования мышления в советской психологии.- М.: Наука, 1966.- с. 236-277.

15. Гейвин Хелен. Когнитивная психология. – СПб.: Питер, 2003.– 272 с.

16. Гершунский Б. С. Содержание обучения как объект прогностического исследования. Программное обучение.- М., 1980.- 37 с.

17. Глазков В.В., Киселев Г.М. Параллельное обучение алгоритмическим языкам. // Информатика и образование. 2002.- №5, 6.

18. Городняя Л.В. Парадигмы программирования в профессиональной подготовке информатиков // Проблемы специализированного образования. - Новосибирск, 1998.- с. 115-124 .

19. Городняя Л.В. Парадигмы программирования.URL: http://www.intuit.ru (дата обращения: 14.02.2010).

20. Гребнева Д.М. Обучение школьников программированию на основе семиотического подхода: Автореферат дис. ... канд. пед. наук. - Екатеринбург, 2014. - 24 с.

21. Диагностика доминирующей перцептивной модальности (С.Ефремцева) / Фетискин Н.П., Козлов В.В., Мануйлов Г.М. Социально-психологическая диагностика развития личности и малых групп. – М., 2002. С.237-238.

22. Дьяченко В.К. Организационная форма учебного процесса и ее развитие. – М.: Педагогика, 1989.

23. Есипов, Б.П. Самостоятельная работа учащихся на уроке [Текст] / Б.П. Есипов. – М.: Учпедгиз, 1961. – 239 с.

24. Если у ребенка кинестетический тип интеллекта. [Электронный ресурс] URL: http://udoktora.net/esli-u-rebenka-kinesteticheskiy-tip-intellekta-62484/ (30.10.2015г).

25. Жужжалов В.Е. Основы интеграции парадигм программирования в курсе информатики // Российская академия образования. Институт содержания и методов обучения.- М.: Образование и информатика, 2004. – 127 с.

26. Жужжалов В.Е. Интеграционные методы изучения программирования в вузовском курсе информатики // Вестник МГПУ, Серия “Информатика и информатизация образования”, М., 2003, № 1 (1).

27. Загвязинский В.И. Теория обучения: современная интерпретация: учеб. пособие для студентов высш. пед. учеб. заведений / В.И. Загвязинский. – М.: Академия, 2001. – 192 с.

28. Зак А.З. Развитие и диагностика мышления подростков и старшеклассников / А.З. Зак. – М.; Обнинск: ИГ – СОЦИН, 2010. – 350 с.

29. Звенигородский Г.А. Первые уроки программирования. - М.: Наука, 1985.- 208 с.

30. Калитина В.В., Пушкарева Т.П., Степанова Т. А. Алгоритмические ментальные карты как эффективное средство обучения программированию // Сборник статей Международной научно-практической конференции «Фундаментальные и прикладные научные исследования», Москва, 2015. С. 179.

31. Калитина В.В., Пушкарева Т.П., Степанова Т. А. Развитие алгоритмического стиля мышления при обучении программированию в вузе // «Теоретические и практические аспекты психологии и педагогики». Аэтерна, Уфа, 2015. С. 101.

32. Кауфман, В.Ш. Языки программирования: концепции и принципы / В.Ш Кауфман. – М. : ДМК-пресс, 2010. – 464 с.

33. Киргизова Е.В. Методика обучения студентов теоретической информатике на информационно-деятельностной основе. Дис. … канд. пед. наук. – Красноярск, 2010. – 201 с.

34. Киргизова Е.В, Нигматулина Э.А. «Специфика обучения программированию будущих учителей информатики» // Научно-технический журнал «Образовательные технологии», г.Воронеж, ВГПУ, №1, 2008. [с. 7-9](УДК 681.142.1.01).

35. Киргизова Е.В., Фаттахова Э.А. (Нигматулина) «К вопросу об интеграции парадигм программирования к обучению информатике в ВУЗе»//Сибирский электронный образовательный журнал «Современное образование», г. Красноярск, КГПУ.-2006. http://port.kspu.ru/ivt/magazine/1/17.htm (Дата посещения 20.02.2010).

36. Кириллов Андрей Григорьевич. Формирование профессиональных компетенций будущего учителя информатики в процессе

обучения программированию :Дис. ... канд. пед. наук : 13.00.02 Шадринск, 2005 162 с. РГБ ОД, 61:05-13/1990.

37. Кнут Дональд Э. Алгоритмическое мышление и математическое мышление/ Пер. И.В.Лебедева. – М.: Изд. иностр. литературы, 1999. – 110 с.

38. Коджаспирова Г.М, Коджаспиров А.Ю. Словарь по педагогике. – Москва: ИКЦ «МарТ»; Ростов н/Д: Издательский центр «МарТ», 2005. – 448 с.

39. Колеченко А.К. Энциклопедия педагогических технологий. – СПб: Каро, 2000.

40. Копаев А.В. О практическом значении алгоритмического стиля мышления // Информационные технологии в общеобразовательной школе. – 2008. - № 6. – С.6-11.

41. Куписевич, Ч. Основы общей дидактики [Текст] / Ч. Куписевич ; Перевод с польского и предисловие О.В. Долженко. – М.: Высшая школа, 1986. – 367 с.

42. Лапчик М.П. и др. Методика преподавания информатики: Учеб.пособие для студ. пед. вузов /М.П. Лапчик, И.Г. Семакин, Е.К.Хеннер; Под общей ред. М.П. Лапчика. М.: Издательский центр «Академия», 2001. – 624 с.

43. Лапчик М.П. Информатика и информационные технологии в системе общего и педагогического образования. Монография. – Омск: Изд-во ОмГПУ, 1999. – 294 с.

44. Лапчик М.П. Использование общеобразовательных аспектов программирования для ЭВМ в совершенствования среднего математического образования. Дис. … канд. пед. наук. – М., 1974. – 163 с.

45. Лапчик, М.П. Методика преподавания информатики [Текст] : учеб.пособие для студ. пед. вузов / М.П. Лапчик, И.Г. Семакин, Е.К. Хеннер ; под общей ред. М.П. Лапчика. – 4-е изд., стер. – М.: Издательский центр «Академия», 2007. – 624 с.

46. Лапчик М.П. Структура и методическая система подготовки кадров информатизации школы в педагогических вузах. Дис. … докт. пед. наук в форме научного доклада. – М., 1999. – 82 с.

47. Лебедева Т.Н. Формирование алгоритмического мышления школьников в процессе обучения рекурсивным алгоритмам в профильных классах средней общеобразовательной школы: Дис. ... канд. пед. наук. - Екатеринбург, 2005. - 219 с.

48. Леднев В.С. Содержание образования: сущность, структура, перспективы. – М.: Высш.шк., 1991. – 224 с.

49. Леонтьев А.Н. Деятельность. Сознание. Личность. / А.Н. Леонтьев.- 2-е изд., стер.- М.: Смысл; Издательский центр «Академия». 2005.- 352 с.

50. Леонтьев А.Н. Проблемы развития психики. - М.: Издательство Академии педагогических наук РСФСР, 1959 г.

51. Лернер, И.Я. Дидактические основы методов обучения [Текст] / И.Я. Лернер. – М.: Педагогика, 1981. – 186 с.

52. Лернер И.Я. Процесс обучения и его закономерности. – М.: Знание, 1980. – 96 с.

53. Лучко Л.Г. Формирование алгоритмической культуры учащихся в процессе обучения базовому курсу информатики :Дис. …канд. пед. наук. - Омск, 1999 – 152 с.

54. Магомедов Рамазан Магомедович. Формирование системно-логического мышления будущего учителя информатики при изучении объектно-ориентированного программирования :Дис. ... канд. пед. наук : 13.00.02 : Москва, 2002 142 с. РГБ ОД, 61:03-13/572-0

55. Найсер У. Познание и реальность. М.: Прогресс, 1981.

56. Наумов А.А. Алгоритмическая культура в контексте подготовки специалистов в области информатики. // Вестник Московского городского

пед.университета. «Информатика и информатизация образования». №2, 2006. http://mf.mgpu.ru/main/content/vestnik/Vestnik7/Vestnik7.php

57. Нигматулина Э. А. Использование практико-ориентированных «живых» задач при обучении программированию [Текст] / Э. А. Нигматулина // Студенческая наука XXI: материалы XIIIМеждунар. студенч. науч.-практ. конф. (Чебоксары, 25 янв. 2017 г.) / редкол.: О. Н. Широков [и др.]. – Чебоксары: ЦНС «Интерактив плюс», 2017. – № 1 (12). – ISSN 2413-3825.

58. Нигматулина Э. А. Разработка и использование системы практико-ориентированных задач при обучении программированию [Текст] / Э. А. Нигматулина // Инновационные технологии в науке и образовании : материалы IX Междунар. науч.-практ. конф. (Чебоксары, 15 янв. 2017 г.) / редкол.: О. Н. Широков [и др.]. – Чебоксары: ЦНС «Интерактив плюс», 2017. – № 1 (9). – ISSN 2413-3981.

59. Нигматулина Э.А. Интеграция парадигм программирования в вузовском курсе информатики / Открытое образование: опыт.проблемы, перспективы: материалы V Всероссийской научно-практической конференции с международным участием. Ерасноярск, 21-22 мая 2009 года / Красноярский государственный педагогический университет им. В.П., Астафьева. – Красноярск, 2009. – 332 с. [с. 43-46].

60. Нигматулина Э.А. Особенности формирования программистского стиля мышления будущих учителей информатики / Информатизация образования и методика электронного обучения: материалы I Международной научной конференции в рамках IV Международного научно-образовательного форума «Человек, семья и общество: история и перспективы развития» (Красноярск, 27-30 сентября 2016 г.); под общ.ред. М.В. Носова. – Красноярск: Сиб. федер. ун-т, 2016. – 468 с. С.209-213.

61. Нигматулина Э.А. Практико-ориентированные «живые» задачи в школьном курсе информатики / Студенческая наука XXI века: материалы X Международной студенч. науч.-практ. конф. (Чебоксары, 30 июня 2016 г.) /

редкол.: О.Н. Широков [и др.]. – Чебоксары: ЦНС «Интерактив плюс», 2016. - № 3 (10). – 344 с. С.44-47.

62. Нигматулина Э.А., Пак Н.И. Студент-центрированное обучение программированию в педагогическом вузе // Информатика и образование, № 2, 2017. – с.8-14 (Журнал из перечня ВАК).

63. Нигматулина Э.А., Сокольская М.А., Степанова Т.А. Расширение понятия алгоритмического мышления при изучении современных технологий программирования в педагогическом вузе (статья) // Материалы VIII Международной научно-практической конференции «Педагогический профессионализм в образовании». – Новосибирск, 2012. – с.152-158.

64. Нигматулина Э.А., Сокольская М.А., Степанова Т.А., Пак Н.И. Программирование: В 2 т. Т.1: учебник для студ. учреждений высш. проф. образования / Под ред. Н.И.Пака. – М., Издательский центр «Академия», 2013. – 272 с.

65. Нигматулина Э.А., Сокольская М.А., Степанова Т.А., Пак Н.И. Программирование: В 2 т. Т.2: учебник для студ. учреждений высш. проф. образования / Под ред. Н.И.Пака. – М., Издательский центр «Академия», 2013. – 240 с.

66. Нигматулина Э.А., Степанова Т.А. Использование параллельного способа обучения в курсе «Языки и методы программирования» // Материалы Всероссийской научно-методической конференции «Интегрированная система профессионального образования: проблемы и пути развития» (25-26 октября 2012 г., г. Красноярск, СибГАУ), с.262-265.

67. Нигматулина Э.А., Степанова Т.А. Параллельное изучение языков программирования в педагогическом вузе // Всероссийский форум педагогического мастерства: сборник научных трудов по материалам I Всероссийской научно-практической конференции 30 июня 2013 г. В 2 томах Том II. – М: «Образование и Информатика», 2013 г. – с. 151-160.

68. Нигматулина Э.А., Степанова Т.А. Реализация параллельного изучения в педвузе языков программирования различных парадигм // Журнал «Информатика и образование» № 4, 2014. – с.28-30 (Журнал из перечня ВАК).

69. Нигматулина Э.А., Степанова Т.А. Телесный подход к обучению программированию // Перспективы и вызовы информационного общества: материалы IV Всероссийской научно-практической конференции с международным участием. Красноярск, 12 ноября 2015 г. [Электронное ресурс] / КГПУ им. В.П. Астафьева. – Красноярск, 2015. С. 131-136.

70. Нигматулина Э.А., Степанова Т.А. Условия формирования алгоритмической культуры студентов на основе информационного подхода // Вестник Красноярского государственного педагогического университета им. В.П.Астафьева. 2011 (1) / Красноярский государственный педагогический университет им. В.П. Астафьева. – Красноярск, 2011. – 280 с. [с. 82-86] (Журнал из перечня ВАК).

71. Новые педагогические и информационные технологии в системе образования: Учебное пособие для студентов пед. вузов и системы повышения квалиф. пед. кадров. / Под ред. Полат. – М.: Изд. центр «Академия», 1999

72. Орел Е.А. Диагностика особенностей мыслительной деятельности специалистов в области информационных технологий (программистов): Автореферетдис…псих.нак. – Москва – 2007. – 23 с.

73. Осмоловская, И.М. Как организовать дифференцированное обучение в школе / И.М. Осмоловская. – М.: Сентябрь, 2002. – 143с.

74. Основы программирования на языке Турбо Паскаль: учебное пособие / А.М. Гилязутдинова, Е.В. Киргизова, Э.А. Фаттахова (Нигматулитна); Красноярский государственный университет. – Красноярск, 2006. – 175 с.

75. Пак Н. И. Гипермозг как основа становления ментальной

дидактики // Интернет – свободный, безопасный, образовательный: Межрегиональная научно-практическая конференция (18–19 октября 2013 года, г. Омск). Омск: Полиграфический центр КАН, 2013.

76. Пак Н.И. Информационное моделирование: учебное пособие; Краснояр. гос. пед. ун-т им. В.П. Астафьева. – Красноярск, 2010. – 152 с.

77. Пак Н.И. Обучение разума как информационный процесс // Российско-корейская научная конференция: Тезисы конференции. Звенигород, 2011.- 184 с.

78. Пак Н. И. От классно-урочной системы к кластерному образованию: образовательная технологическая платформа «Мега-класс» // Международная научно-практическая конференция «Информатизация образования – 2016», 14–17 июня, г. Сочи. М.: Изд-во СГУ, 2016.

79. Пак Н.И. Проективный подход в обучении как информационный процесс: Монография. – Красноярск:РИО КГПУ, 2008. – 111 с.

80. Пак Н. И. Пространственно-временная информационная модель памяти // Сб. трудов конференции «Фундаментальные науки и образование». Бийск, 2012.

81. Пак Н.И., Степанова Т.А. Параллельный способ обучения курсу «Численные методы» // Педагогическая информатика, 2001 г., № 1.

82. Пак Н.И., Степанова Т.А. Использование параллельных технологий обучения в курсах информатики // Новые информационные технологии в университетском образовании: Тезисы конференции. – Новосибирск: СГУПС, ИДМИ, 2001

83. Пак Н.И., Степанова Т.А., Хегай Л.Б., Яковлева Т.А. Вертикальная модель подготовки учителя информатики в педагогическом вузе // Вестник Красноярского государственного педагогического университета им. В.П. Астафьева. 2009 (1) / Красноярский государственный педагогический университет им. В.П. Астафьева. – Красноярск, 2009.

84. Пак Н.И., Степанова Т.А., Хегай Л.Б., Яковлева Т.А. Вертикальная модель подготовки на основе интеграции школьного и вузовского образования // Открытое образование. 2009 (1). – Красноярск, 2009. –с. 4-9.

85. Пак Н. И. Студент-центрированное обучение в образовательных кластерах //IМеждународная научная конференция «Информатизация образования и методика электронного обучения», 27–30 сентября, г. Красноярск. Красноярск: Изд-во СФУ, 2016.

86. Первин Ю. А. От операционного стиля мышления через педагогические компетенции к универсальным учебным действиям. http://ito.edu.ru/sp/SP/SP-0-2010_11_23.html

87. Пидкасистый, П.И. Самостоятельная познавательная деятельность школьников в обучении [Текст] : Теоретико-экспериментальное исследование / П.И. Пидкасистый. – М.: Педагогика, 1980. – 240 с.

88. Программирование на языке Паскаль: учеб.пособие / Е.В. Киргизова, Э.А. Нигматулина, Т.В. Захарова. – Красноярск: Сибирский федеральный университет, 2015. – 110 с.

89. Прохоров, А.М. Большой энциклопедический словарь / А.М.Прохоров.– СПб. : Норинт, 2004. – 1456 с. - с.12-15

90. Психология XXI века: Учебник для вузов / Под ред. В.Н. Дружинина. – М.: ПЕР СЭ, 2003.

91. Пушкарева Т.П., Степанова Т.А., Калитина В.В., Нигматулина Э.А. Развитие алгоритмического мышления в вузе как основа успешности в профессиональной деятельности / Materials of the XII International scientific and practical conference, «Modern scientific potential – 2016». Volume 9. Pedagogical sciences. Sheffield. Science and education LTD – 88 стр. С. 9-12.

92. Пушкарева Т.П., Степанова Т.А., Калитина В.В., Нигматулина Э.А. Средства развития алгоритмического мышления / Актуальные направления фундаментальных и прикладных исследований: материалы VIII

международной научно-практической конференции. Том 3. 9-10 марта 2016 г. North Charleston, USA. С. 86-88.

93. Пушкарева Т.П., Степанова Т.А., Калитина В.В., Нигматулина Э.А. Кинестетические тренажеры как эффективное средство развития алгоритмического стиля мышления // Современный научный вестник, г. Белгород. Том 5, № 2. 2015. С.76-80.

94. Пушкарева Т.П., Степанова Т.А., Калитина В.В., Нигматулина Э.А. Кинестетические тренажеры как эффективное средство развития алгоритмического стиля мышления / Материали за 12-а международнанаучна практична конференция, «Бъдещитеизследвания», - 2016. Том 5. Педагогически науки. София. «Бял ГРАЛ-БГ» ООД -112 стр. С.69-71.

95. Пушкарева Т.П., Степанова Т.А., Калитина В.В., Нигматулина Э.А. Развитие алгоритмического мышления в вузе как основа успешности в профессиональной деятельности // Современный научный вестник, г. Белгород. Том 5, № 2. 2015. С.81-85.

96. Российская педагогическая энциклопедия. В 2-х томах. Т.1. М. – Научное издательство «Большая российская энциклопедия», 1993.

97. Рубинштейн С.Л. Проблемы общей психологии. 2-е изд., М., 1976.- с. 186.

98. Рукосуева Д.А., Сокольская М.А., Нигматулина Э.А. «Методология информационного подхода к визуализации абстрактных понятий курса «Информатика» // Доклады и тезисы секции «Информатика образования» Ершовской конференции по информатике 2011. Новосибирск, 2011. - стр. 93-98

99. Салангина Н.Я. Реализация линии алгоритмизации в курсе «Языки и методы программирования» физико-математических специальностей педагогических вузов: Дис. ... канд. пед. наук. - Москва, 1999. - 157 с.

100. Скаткин, М.Н. Проблемы современной дидактики [Текст] / М.Н. Скаткин. – 2-е изд. – М.: Педагогика, 1984. – 96 с.

101. Скибицкий Э.Г. Построение содержания образования по учебной дисциплине //Всероссийская научно-методическая конференция «Инновационная интегрированная система профессионального образования: методы и пути решения»: Материалы конференции. – Красноярск: Сибирский государственный аэрокосмический университет, 2011. С. 24-27

102. Сластенин В.А. и др. Педагогика: Учеб.пособие для студ. высш. пед. учеб. заведений / В. А. Сластенин, И. Ф. Исаев, Е. Н. Шиянов; Под ред. В.А. Сластенина. - М.: Издательский центр «Академия», 2002. - 576 с.

103. Слинкина И.Н. Использование компьютерной техники в процессе развития алгоритмического мышления у младших школьников: Автореф. дис.. канд. пед. наук. - Екатеринбург, 2000. - 22с.

104. Сокольская М.А. Методическая система обучения основам параллельного программирования будущих учителей информатики: Дис. ... канд. пед. наук. - Красноярск, 2012. - 157 с.

105. Степанов М.А. Опыт мышления тела: автореф. дис. … канд. философ.наук. СПб, 2011. с.6

106. Степанова Т.А. Методическая система обучения курсу «Численные методы» в условиях информационно-коммуникационной предметной среды: Дис. ... канд. пед. наук. - Красноярск, 2003.

107. Степанова Т. А. Сущность алгоритмического мышления с позиций информационного подхода // Инновации в непрерывном образовании, 2012, №3. С. 95.

108. Степанова Т.А. Теория алгоритмического мышления: учебное пособие для магистрантов, учителей общеобразовательных учреждений, преподавателей вузов; Краснояр. гос. пед. ун-т им. В.П. Астафьева. – Красноярск, 2014. – 72 с.

109. Стоякова К.Л. Использование логической парадигмы программирования для обучения информатике студентов в инженерных вузах: Дис. ... канд. пед. наук. -. Москва, 2008. – 159 с.

110. Теория и методика обучения информатике: учебник / [М.П. Лапчик, И.Г. Семакин, Е.К. Хеннер, М.И. Рагулина и др.]; под ред. М.П. Лапчика. – М.: Издательский центр «Академия», 2008. – 592 с.

111. Тхостов А.Ш. Психология телесности. — М.: Смысл, 2002. – 287 с.

112. Чепикова Е.Ю. Разработка кинестетического тренажера по теме «динамические структуры данных» курса «языки и методы программирования» // Актуальные проблемы информатики и информационных технологий в образовании: материалы Всероссийской научно-практической конференции с международным участием в рамках XVIII международного научно-практического форума студентов, аспирантов и молодых ученых «Молодежь и наука XXI века». Красноярск, 23 мая 2017 г. [Электронный ресурс] / ред. кол.; отв. ред. П.С. Ломаско. – Электрон. дан. / Краснояр. гос. пед. ун-т им. В.П. Астафьева. – Красноярск, 2017.

113. Что такое ментальный подход? [Электронный ресурс]: ментология – управление мышлением. Режим доступа: http://mentally.eu/mentologiya/ (25.02.2017)

114. Шаров, А.С. Рефлексивный подход в обучении информатике [Текст]: Монография / А.С. Шаров, Д.А. Шаров. – Омск: Издательство ОмГПУ, 2007. – 202 с.

115. Шрайнер А.А. Повышение качества математического образования учащихся посредством формирования и развития их алгоритмической культуры :Дис. ... канд. пед. наук. - Новосибирск, 1997. - 191 с.

116. Федеральный государственный образовательный стандарт высшего образования (уровень высшего образования бакалавриат) по

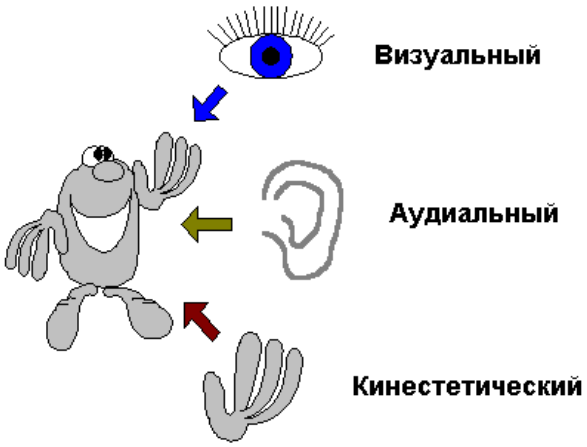
направлению 44.03.01 Педагогическое образование утвержденный от 4 декабря 2015. – 17 с.

117. Федеральный государственный образовательный стандарт высшего образования (уровень высшего образования бакалавриат) по направлению подготовки 44.03.05 Педагогическое образование (с двумя профилями подготовки) утвержденный от 9 февраля 2016. – 17 с.

Приложения

Приложение 1

Тест аудиал, визуал, кинестетик (диагностика доминирующей перцептивной модальности С. Ефремцева. (Методика на ведущий канал восприятия)



Диагностика доминирующей перцептивной модальности С. Ефремцева служит для определения ведущего типа восприятия: аудиального, визуального или кинестетического.

Инструкция к тесту.

Прочитайте предлагаемые утверждения. Поставьте знак «+», если Вы согласны с данным утверждением, и знак «-», если не согласны.

Тестовый материал (вопросы).

№	Вопросы	Ответ «+» или «–»
1.	Люблю наблюдать за облаками и звездами	
2.	Часто напеваю себе потихоньку	
3.	Не признаю моду, которая неудобна	
4.	Люблю ходить в сауну	
5.	В автомашине цвет для меня имеет значение	
6.	Узнаю по шагам, кто вошел в помещение	
7.	Меня развлекает подражание диалектам	
8.	Внешнему виду придаю серьезное значение	
9.	Мне нравится принимать массаж	
10.	Когда есть время, люблю наблюдать за людьми	
11.	Плохо себя чувствую, когда не наслаждаюсь движением	
12.	Видя одежду в витрине, знаю, что мне будет хорошо в ней	

13.	Когда услышу старую мелодию, ко мне возвращается прошлое	
14.	Люблю читать во время еды	
15.	Люблю поговорить по телефону	
16.	У меня есть склонность к полноте	
17.	Предпочитаю слушать рассказ, который кто-то читает, чем читать самому.	
18.	После плохого дня мой организм в напряжении	
19.	Охотно и много фотографирую	
20.	Долго помню, что мне сказали приятели или знакомые	
21.	Легко могу отдать деньги за цветы, потому что они украшают жизнь	
22.	Вечером люблю принять горячую ванну	
23.	Стараюсь записывать свои личные дела	
24.	Часто разговариваю с собой	
25.	После длительной езды на машине долго прихожу в себя	
26.	Тембр голоса многое мне говорит о человеке	
27.	Придаю значение манере одеваться, свойственной другим	
28.	Люблю потягиваться, расправлять конечности, разминаться	
29.	Слишком твердая или слишком мягкая постель для меня мука	
30.	Мне нелегко найти удобную обувь	
31.	Люблю смотреть теле- и видеофильмы	
32.	Даже спустя годы могу узнать лица, которые когда-либо видел	
33.	Люблю ходить под дождем, когда капли стучат по зонтику	
34.	Люблю слушать, когда говорят	
35.	Люблю заниматься подвижным спортом или выполнять какие-либо двигательные упражнения, иногда и потанцевать	
36.	Когда близко тикает будильник, не могу уснуть	
37.	У меня неплохая стереоаппаратура	
38.	Когда слушаю музыку, отбиваю такт ногой	
39.	На отдыхе не люблю осматривать памятники архитектуры	
40.	Не выношу беспорядок	
41.	Не люблю синтетических тканей	
42.	Считаю, что атмосфера в помещении зависит от освещения	
43.	Часто хожу на концерты	
44.	Пожатие руки много говорит мне о данной личности	
45.	Охотно посещаю галереи и выставки	
46.	Серьезная дискуссия – это интересно	
47.	Через прикосновение можно сказать значительно больше, чем словами	
48.	В шуме не могу сосредоточиться	

Ключ к тесту аудиал, визуал, кинестетик.

• **Визуальный канал восприятия**: 1, 5, 8, 10, 12, 14, 19, 21, 23, 27, 31, 32, 39, 40, 42, 45.

• **Аудиальный канал восприятия**: 2, 6, 7, 13, 15, 17, 20, 24, 26, 33, 34, 36, 37, 43, 46, 48.

• **Кинестетический канал восприятия**: 3, 4, 9, 11, 16, 18, 22, 25, 28, 29, 30, 35, 38, 41, 44, 47.

Уровни перцептивной модальности (ведущего типа восприятия):

• 13 и более – высокий;

• 8-12 – средний;

• 7 и менее – низкий.

Интерпретация результатов:

Подсчитайте, количество положительных ответов в каждом разделе ключа. Определите, в каком разделе больше ответов «да» («+»). Это Ваш тип ведущей модальности. Это ваш главный тип восприятия.

Визуал. Часто употребляются слова и фразы, которые связаны со зрением, с образами и воображением. Например: «не видел этого», «это, конечно, проясняет все дело», «заметил прекрасную особенность». Рисунки, образные описания, фотографии значат для данного типа больше, чем слова. Принадлежащие к этому типу люди моментально схватывают то, что можно увидеть: цвета, формы, линии, гармонию и беспорядок.

Кинестетик. Тут чаще в ходу другие слова и определения, например: «не могу этого понять», «атмосфера в квартире невыносимая», «ее слова глубоко меня тронули», «подарок был для меня чем-то похожим на теплый дождь». Чувства и впечатления людей этого типа касаются, главным образом, того, что относится к прикосновению, интуиции, догадке. В разговоре их интересуют внутренние переживания.

Аудиал. «Не понимаю что мне говоришь», «это известие для меня…», «не выношу таких громких мелодий» – вот характерные высказывания для людей этого типа; огромное значение для них имеет все, что акустично: звуки, слова, музыка, шумовые эффекты.

Несмотря на то, что основных каналов восприятия существует три, человек обрабатывает свой жизненный опыт четырьмя способами. Ведь существует еще и дигитальный канал – некий внутренний монолог, связанный со словами и числами.

Дигитал *(*он же ***дискрет)*** – весьма своеобразный и достаточно редко встречающийся типаж, которому свойственно особое восприятие мира. Выражения эмоций, разговоров о чувствах, красочного описаний картин природы и т.п. от дискретов дождаться сложно. Этот тип ориентирован, прежде всего, на логику, смысл и функциональность. В разговоре с дискретом складывается впечатление, что он как будто ничего не чувствует, но много знает, и еще больше – стремится узнать, осмыслить, понять и разложить по полочкам. Но это совсем не так! Люди с дигитальным каналом восприятия как раз невероятно чувствительны и ранимы.

Среди представителей этого типа особенно много шахматистов, программистов, а также всевозможных исследователей и ученых. В их лексиконе часто встречаются выражения: «где тут логика?», «надо проанализировать ситуацию», «итак, методом исключения мы выясняем…». Поскольку дискреты воспринимают мир через логическое осмысление, общаться с ними стоит именно с помощью логических доводов, желательно еще и подкрепленных статистическими данными.

Отличительные признаки	**Визуальный тип**
Способ получения информации	Посредством зрения – благодаря использованию наглядных пособий или непосредственно наблюдая за тем, как выполняются соответствующие действия
Восприятие окружающего мира	Восприимчивы к видимой стороне окружающего мира; испытывают жгучую потребность в том, чтобы мир вокруг них выглядел красиво; легко отвлекаются и впадают в беспокойство при виде беспорядка
На что обращают внимание при общении с людьми	На лицо человека, его одежду и внешность
Речь	Описывают видимые детали обстановки – цвет, форму, размер и внешний облик вещей
Движения глаз	Когда о чем-нибудь размышляют, обычно смотрят в потолок; когда слушают, испытывают потребность смотреть в глаза говорящему и хотят, чтобы те, кто их слушают, также смотрели им в глаза
Память	Хорошо запоминают зримые детали обстановки, а также тексты и учебные пособия, представленные в печатном или графическом виде

Отличительные признаки	Аудиальный тип
Способ получения информации	Посредством слуха – в процессе разговора, чтения вслух, спора или обмена мнениями со своими собеседниками
Восприятие окружающего мира	Испытывают потребность в непрерывной слуховой стимуляции, а когда вокруг тихо, начинают издавать различные звуки – мурлычут себе под нос, свистят или сами с собой разговаривают, но только не тогда, когда они заняты учебой, потому что в эти минуты им необходима тишина; в противном случае им приходится отключаться от раздражающего шума, который исходит от других людей
На что обращают внимание при общении с людьми	На имя и фамилию человека, звук его голоса, манеру его речи и сказанные им слова
Речь	Описывают звуки и голоса, музыку, звуковые эффекты и шумы, которые можно услышать в окружающей их обстановке, а также пересказывают то, что говорят другие люди
Движения глаз	Обычно смотрят то влево, то вправо и лишь изредка и ненадолго заглядывают в глаза говорящему
Память	Хорошо запоминают разговоры, музыку и звуки
Отличительные признаки	**Кинестетический тип**
Способ получения информации	Посредством активных движений скелетных мышц – участвуя в подвижных играх и занятиях, экспериментируя, исследуя окружающий мир, при условии, что тело постоянно находится в движении
Восприятие окружающего мира	Привыкли к тому, что вокруг них кипит деятельность; им необходим простор для движения; их внимание всегда приковано к движущимся объектам; зачастую их отвлекает и раздражает, когда другие люди не могут усидеть на месте, однако им самим необходимо постоянно двигаться
На что обращают внимание при общении с людьми	На то, как другой себя ведет; что он делает и чем занимается
Речь	Широко применяют слова, обозначающие движения и действия; говорят в основном о делах, победах и достижениях; как правило, немногословны и быстро переходят к сути дела; часто используют в разговоре свое тело, жесты, пантомимику
Движения глаз	Им удобнее всего слушать и размышлять, когда их глаза опущены вниз и в сторону; они практически не смотрят в глаза собеседнику, поскольку именно такое положение глаз позволяет им учиться и одновременно действовать; но если поблизости от них происходит суета, их взгляд неизменно направляется в ту сторону
Память	Хорошо запоминают свои и чужие поступки, движения и жесты

Приложение 2

Методические рекомендации по использованию полиязыкового подхода к обучению программированию (на примере императивной и объектно-ориентированной парадигм)

В настоящее время в вузах и в школах в основном преподается один из двух процедурных языков – Паскаль или Бейсик. Покажем на примере возможность объединение двух парадигм программирования – имеративной и объектно-ориентированной парадигмы. В качестве языков программирования императивной парадигмы рассмотрим – Паскаль, а объектно-ориентированной парадигмы – Delphi. Во-первых, основной принцип, заложенный в языке Паскаль, – поддержка структурной методики программирования. Этот же принцип заложен в основе алгоритмического языка. Кроме того, в нем, в отличие от языка Бейсик, можно отразить концепцию типов данных. Во-вторых, язык Delphi основывается на языке Паскаль, что позволяет объединить изучение этих языков в учебном процессе.

На изучение линии алгоритмизации и программирования учителю отведено мало времени. Поэтому изучение языка программирования лучше начать тогда, когда ученики начнут сами составлять алгоритмы, т. е. алгоритмизацию и язык программирования изучать параллельно. При изучении основных алгоритмических конструкций (следование, ветвление, цикл) следует показать не только то, как они реализуются на алгоритмическом языке, но и на Паскале. Понятие процедуры и функции необходимо вводить вместе с понятием вспомогательной подпрограммы на алгоритмическом языке.

Изучение Delphi лучше начать после изучения алгоритмических конструкций. При этом необходимо сделать вводный урок, на котором рассказать ученикам об императивной и объектно-ориентированной парадигмах, дать понятие объекта, свойств и методов объекта. По сути,

обработчик некоторого метода объекта – это своего рода процедура, которая в свою очередь так же может вызвать некоторые другие методы. Необходимо пояснить ученикам, что можно пользоваться не только стандартными процедурами обработки события, но и создавать свои процедуры и функции, которые будут вызываться из обработчиков.

Понятие объекта, его свойств и методов легко объяснить, используя в качестве примеров объекты реального мира. Например, объект «линейка» обладает следующими свойствами: шкала (миллиметры, дюймы и т. д.), тип линейки (обычная, логарифмическая, чертежная и т. д.), размеры и др. Методы этого объекта зависят от его типа. Например, «провести отрезок между двумя точками», «измерить длину отрезка», «поделить отрезок пополам». Классом будет множество всех линеек, а объектом – конкретный экземпляр.

Начать изучение объектно-ориентированной парадигмы следует начать на языке Паскаль. Это можно сделать на том же вводном уроке. В нашей работе мы далее предлагаем свой вариант конспекта вводного занятия.

Тематическое планирование

Мы предлагаем следующий план проведения занятий:

Тема	Цели	Основные понятия	Рекомендации
Алгоритм. Свойства алгоритма. Способы записи алгоритма.	1. Дать понятие алгоритма 2. Познакомить со свойствами алгоритма и способами их записи	Алгоритм, свойства алгоритма, блок-схема	Учитель подводит детей к понятию алгоритма, используя некоторый пример, например, распорядок дня. Свойства дискретности и массовости являются следствием определения понятия алгоритма, поэтому эти свойства можно не выделять.
Исполнители алгоритмов. Компьютер как формальный исполнитель алгоритмов.	1. Дать понятие исполнителя алгоритма и СКИ 2. Дать понятие формального	Исполнитель алгоритма, система команд исполнителя (СКИ),	На уроке следует показать схему функционирования исполнителя алгоритма. Необходимо указать, что свойство точности алгоритма с точки зрения исполнения

Тема	Цели	Основные понятия	Рекомендации
	исполнения алгоритма	формальное исполнение алгоритма	означает, что в алгоритм должны входить команды, входящие в СКИ
Языки программирования и алгоритмические языки. Типы данных, ввод и вывод данных.	1. Дать понятие языков программирования и алгоритмических языков 2. Пояснить назначение языков программирования 3. Познакомить со свойствами величин в алгоритмах	Язык программирования, алгоритмический язык, величина, тип данных	На уроках алгоритмизация и программирование осваиваются параллельно. Ученики должны освоить, что всякая величина занимает определенное место в памяти. Понятие о различных типах данных можно раскрыть, используя понятия, полученные при изучении табличного процессора и баз данных.
Алгоритмическая конструкция следования и ветвления.	1. Дать понятие алгоритмической конструкции 2. Дать понятия конструкций следования и ветвления	Алгоритмическая конструкция, следование, условие, ветвление	Учитель подводит учеников к понятию алгоритмов следования и ветвления, используя некоторые примеры. Тут же разбираются формы представления этих конструкций
Разработка линейного и разветвляющегося алгоритма программирования.	1. Закрепить теоретический материал 2. Выработать навыки создания алгоритмов следования и ветвления		На практике ученики не только разрабатывают алгоритмы решения задач, но и строят блок-схемы этих алгоритмов
Алгоритмическая конструкция повторения. Логический тип данных.	1. Дать понятие конструкции повторения 2. Дать понятие логического типа данных	Цикл, предусловие, постусловие, счетчик, логический тип данных	Необходимо указать, в каких случаях применяется тот или иной вид циклического алгоритма. Полезно познакомить учеников с операциями над величинами логического типа.
Разработка алгоритма с циклом.	1. Закрепить теоретический материал 2. Выработать навыки создания алгоритмов		

Тема	Цели	Основные понятия	Рекомендации
	повторения		
Вспомогательные подпрограммы.	1. Дать понятие вспомогательных алгоритмов, подпрограмм, функций и процедур 2. Пояснить назначение вспомогательных алгоритмов	Вспомогательный алгоритм, подпрограмма, функция, процедура	Необходимо объяснить ученикам, чем вызвана необходимость использовать в алгоритмах вспомогательные алгоритмы. Ученики должны четко понимать различия между функцией и процедурой, локальных и глобальных переменных.
Разработка алгоритма, содержащего подпрограмму.	1. Закрепить теоретический материал 2. Выработать навыки создания алгоритмов, содержащих вспомогательные алгоритмы		Важно, чтобы ученики сами определяли вид подпрограммы (функция или процедура), который удобнее будет использовать в данной задаче.
Массивы.	1. Дать понятие массива 2. Научить обработке массивов	Массив	Необходимо объяснить ученикам, чем вызвана необходимость использовать массивы. Можно вскользь коснуться с понятием множества.
Разработка алгоритма, содержащего обработку массива.	1. Закрепить теоретический материал 2. Выработать навыки обработки массивов		Полезно разобрать на уроке задачу по сортировке массива или поиску элемента в массиве.
Этапы разработки программы.	1. Познакомить с этапами разработки программ 2. Научить правильной разработке программ	Отладка, тестирование	Необходимо указать на сложность отыскания некоторых синтаксических ошибок, а также указать на достоинства и недостатки языков, требующих описания используемых в их программах переменных
Проверочная работа по пройденному материалу.	Провести контроль знаний и умений учащихся		
Введение в	1. Дать понятие	Процедурное и	Все понятия необходимо

Тема	Цели	Основные понятия	Рекомендации
объектно-ориентированное программирование.	процедурного и объектно-ориентированного языков; 2. Указать на преимущества объектно-ориентированных языков; 3. Продемонстрировать объектные возможности языка Паскаль.	объектно-ориентированное программирование, объект, его свойства и методы	сопровождать конкретными примерами. В качестве примера лучше рассмотреть решение некоторой задачи с помощью объектно-ориентированного языка программирования.
Среда Delphi. Основные объекты: форма, кнопка, поле ввода. Разработка программ в среде Delphi (3 часа).	1. Познакомить со средой Delphi 2. Познакомить с основными объектами, их свойствами и методами 3. Закрепить теоретический материал на практике	Язык Delphi, форма, кнопка, поле ввода	Необходимо провести аналогию Delphi и объектно-ориентированного Паскаля, показать, что свойства и методы любого объекта находятся в инспекторе объектов. На практике полезно кроме стандартных объектов создавать также и свои собственные.
Разработка программы «Калькулятор»	Закрепить теоретический материал на практике		
Контрольная работа по пройденной теме.	Провести контроль знаний и умений учащихся		

Приложение 3

Полиязыковой подход к обучению теме «Массивы»

Содержание системы учебных курсов, относимых к информатике, постоянно расширяется и совершенствуется. В первую очередь, это, безусловно, связано с развитием современных информационных и телекоммуникационных технологий. Это привело не только к появлению большого количества языковых средств кодирования алгоритмов, но и к выделению основных способов разработки самих алгоритмов. Такие способы в специализированной литературе получили название парадигм. В настоящее время сложилось четыре парадигмы, которые диктуют как подходы к разработке программ на современных языках программирования, так и методы изучения информатики. Условно можно выделить процедурную, объектно-ориентированную, логическую и функциональную парадигмы. Разделение всех методов конструирования алгоритмов на четыре парадигмы вполне приемлемо, так как под него попадают все известные языки программирования. Невозможно говорить о явных преимуществах какой-либо одной парадигмы перед остальными. Каждая из них, наряду с большим количеством положительных особенностей, имеет и свои отрицательные аспекты.

Как правило, выбор способа обработки информации определяется спецификой предметной области решаемой задачи. Важно отметить, что использование различных подходов (парадигм) к программированию существенно влияет на эффективность процесса создания компьютерных программ.

Проблема выбора языка и метода программирования гораздо шире, чем проблема обучения нескольким языкам программирования. Как правило, преподаватель, зная один или два языка программирования, изучит третий лишь в том случае, если он относится к той же парадигме. Изучения языка программирования относящегося к другой парадигме вызывает

определенный ряд сложностей, так как при переходе к программированию методами, которые относятся к другой парадигме, обучаемый обязан изменить не только подход к решению поставленной задачи, но и перестроить мыслительную деятельность относительно новой парадигмы.

Задача интеграции обучения различным парадигмам в рамках единой методической системы обучения программированию является сложной. Но, тем не менее, можно определить первые шаги в построении методической системы обучения информатике. Основой при разработке такой системы обучения программированию должно стать объединение различных компонентов, которые составляют сущность методической системы обучения программированию. К таким компонентам относятся цели, содержание, методы, формы и средства обучения. Интеграция различных подходов в программировании должна найти отражение всех перечисленных компонентов методической системы.

Основным достоинством такой интеграционной системы может быть расширение числа подходов и способность выбора наиболее приемлемой парадигмы к решению поставленной задачи. Это даст возможность перехода курса программирования на более высокий уровень.

Рассмотрим методику освоения темы «Структурированный тип данных» для процедурной и объектно-ориентированной парадигмы.

Изучение данной темы необходимо начать с введения понятия массива [45].

Массив — это структура данных, представляющая собой совокупность элементов одного типа (integer , real или char.).

Массивы бывают:

1) одномерными;

2) двумерными;

3) многомерными.

Номер элемента массива называется индексом. Индекс – это значение порядкового типа, определенного как тип индекса данного массива.

Отличительная особенность массивов заключается в том, что все их компоненты можно легко упорядочить и обеспечить доступ к любому из них простым указанием его порядкового номера – индекса. Индексация заслуживает отдельного пояснения. Она не является единственно возможным способом выделения элементов структурированной величины. Для учащихся неочевидно, что границы изменения индексов назначаются произвольно, что тип индексов – интервальный – не обязательно базируется на типе integer (хотя чаще всего это так). Наконец, полезно соотнести представление о линейном (одномерном) массиве с цепочкой ячеек памяти ЭВМ, в которых хранятся элементы массива.

Далее необходимо рассмотреть описание массива, организацию ввода и вывода его элементов.

Навыки использования массивов закрепляются с помощью решения типовых задач. К ним относятся: организация поэлементного ввода и вывод линейного массива (простой цикл), подсчет числа положительных элементов линейного числового массива, нахождение наибольшего элемента линейного числового массива и т. д. Затем переходят к задачам посложнее, требующим организации структур типа «цикл в цикле», и более сложным: упорядочить линейный числовой массив по возрастанию или по убыванию, найти наибольший элемент в двумерном массиве и т.д.

Обучение методам сортировки массива – вставкой, выбором, «пузырьковая» – на основе процедурной парадигмы программирования может проходить в несколько этапов следующим образом:

1 этап - репродуктивный:

На 1 этапе объясняется суть метода сортировки вставкой, который состоит в следующем: массив разделяется на две части: отсортированную и неотсортированную. Элементы из неотсортированной части поочередно

выбираются и вставляются в отсортированную часть так, чтобы не нарушить в ней упорядоченность элементов. В начале работы алгоритма в качестве отсортированной части массива принимают только один первый элемент, а в качестве неотсортированной части – все остальные элементы.

Согласно выше описанному алгоритму представляется схема реализации метода сортировки вставкой (рис.1):

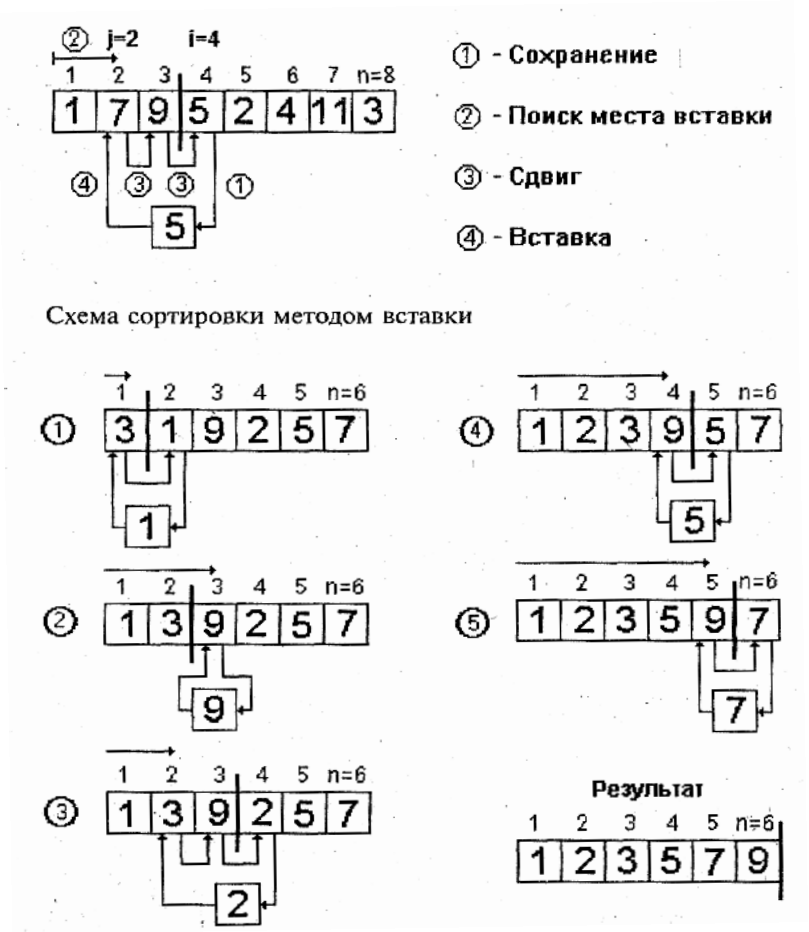


Рис. 1

Далее обучающимся предоставляется возможность рассмотреть метод сортировки на конкретном примере, изучить алгоритм организации метода с использованием кинестетического тренажера и электронного тренажера.

В электронном тренажере (рис. 2) приведен пример задачи: «Имеется массив, состоящий из n элементов с номерами от 1 до n. Отсортировать их методом вставки». Далее представлен алгоритм, по которому происходит сортировка, указаны количество элементов в массиве и их значения.

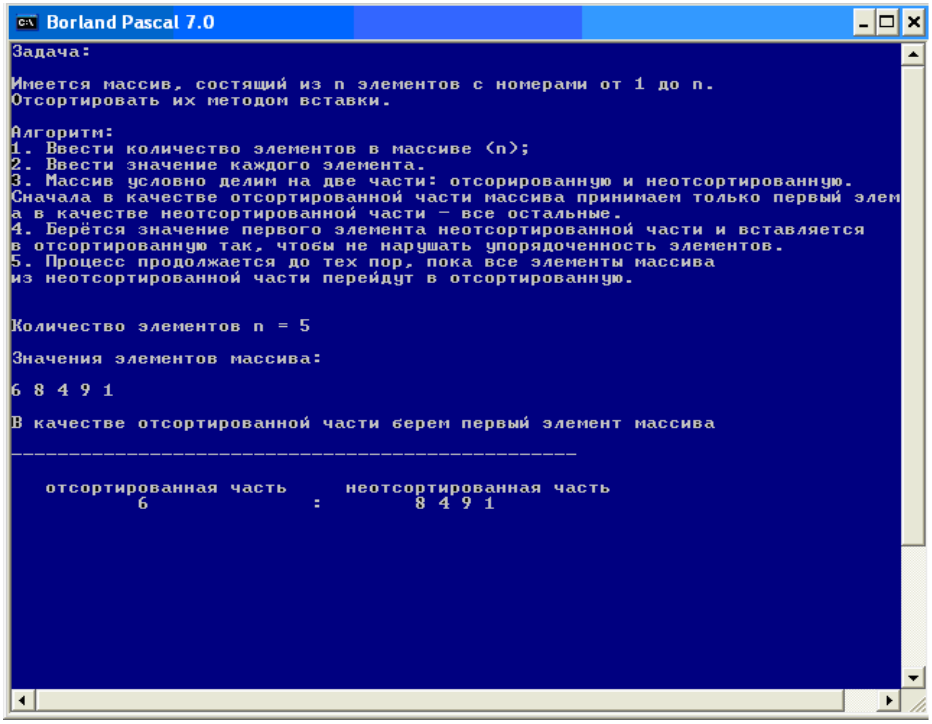


Рис. 2

Обучающиеся прослеживают выполнение каждого последующего шага алгоритма, который выполняется по нажатию любой клавиши на клавиатуре (рис. 3).

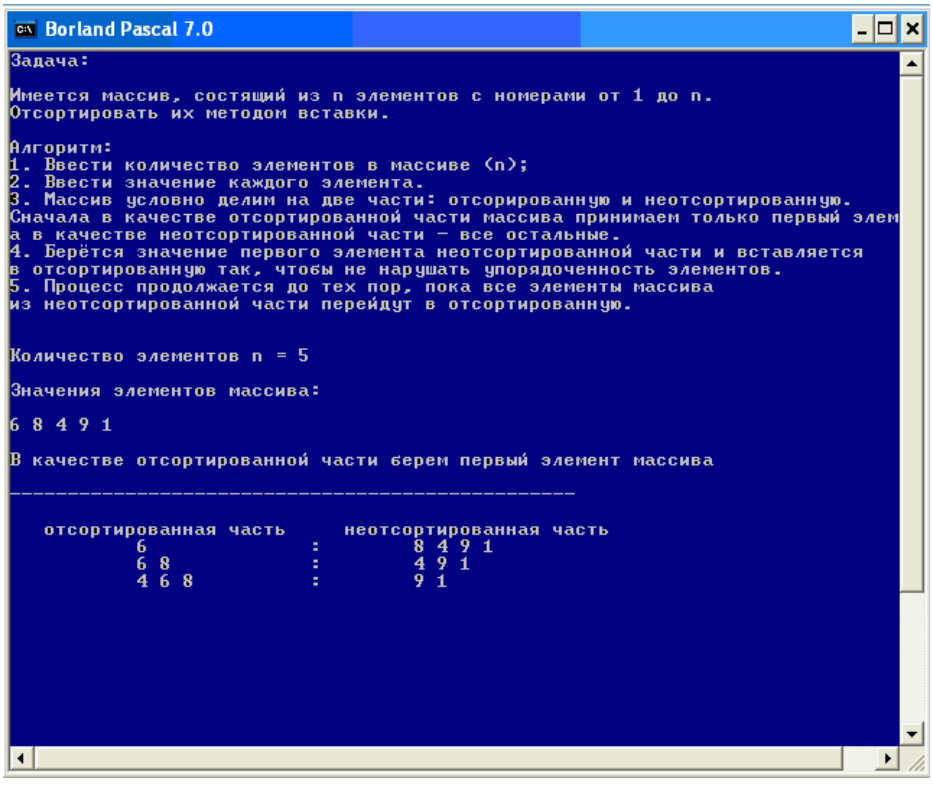


Рис. 3

По окончании выполнения всех пунктов алгоритма, обучающийся может видеть результат – массив отсортирован методом вставки (рис. 4).

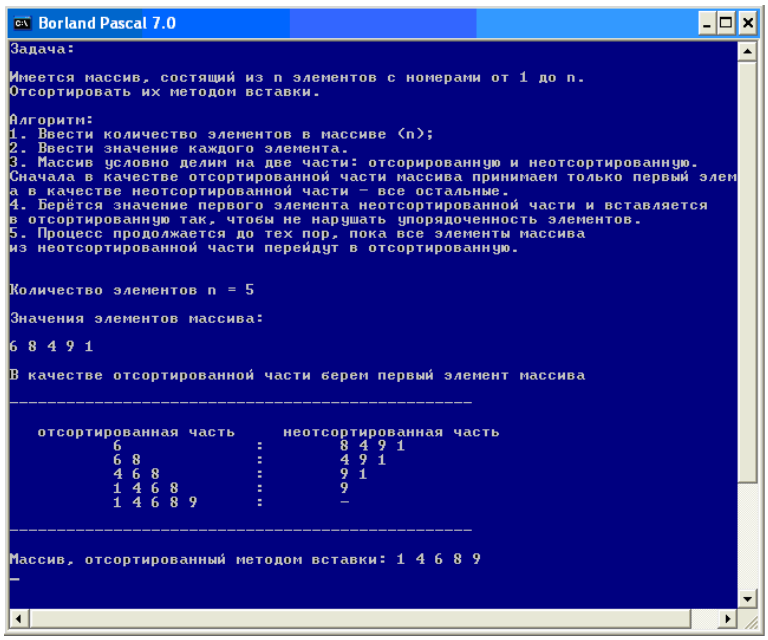


Рис. 4

2 этап: конструктивный.

Изучив алгоритм, обучающийся переходит к выполнению практического задания, которое организовано по рассмотренному ранее примеру. В данном случае описание алгоритма отсутствует, обучающимся необходимо самим вводить значение первого элемента отсортированной части (рис. 5).

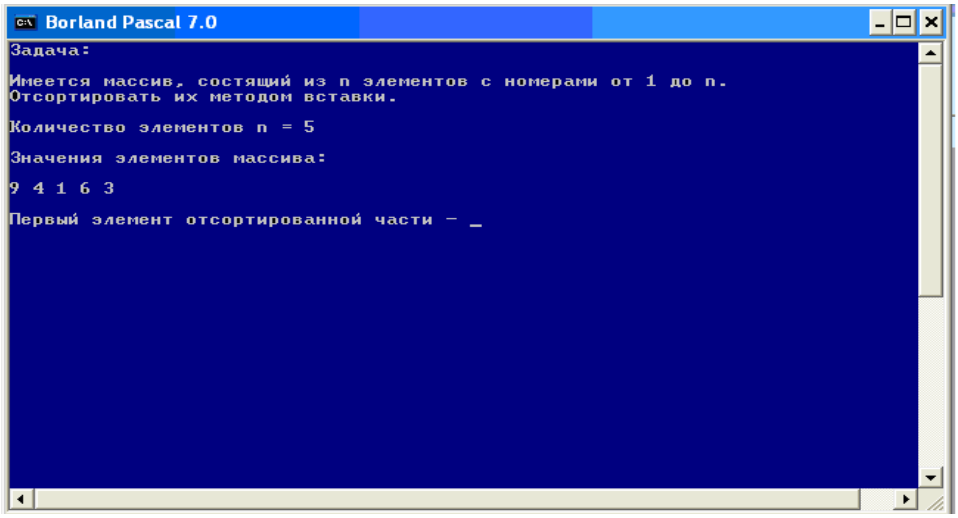


Рис. 5

Переход к следующему шагу происходит только в том случае, если значение введено правильно (рис.6).

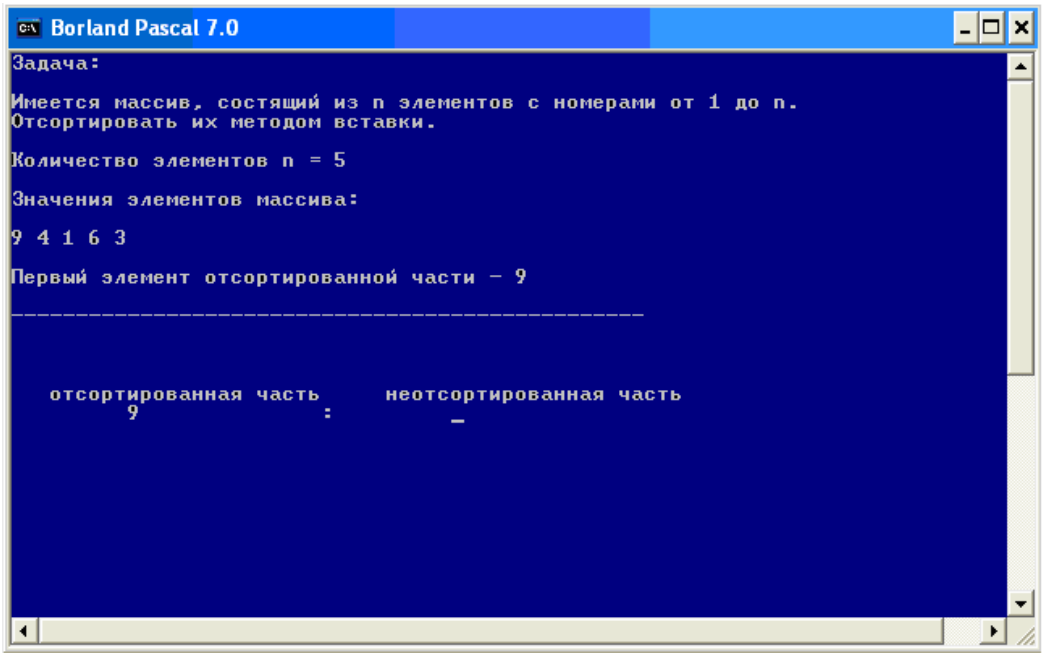


Рис. 6

Каждый элемент неотсортированной части также вводится самостоятельно. И так далее, по алгоритму (рис. 7).

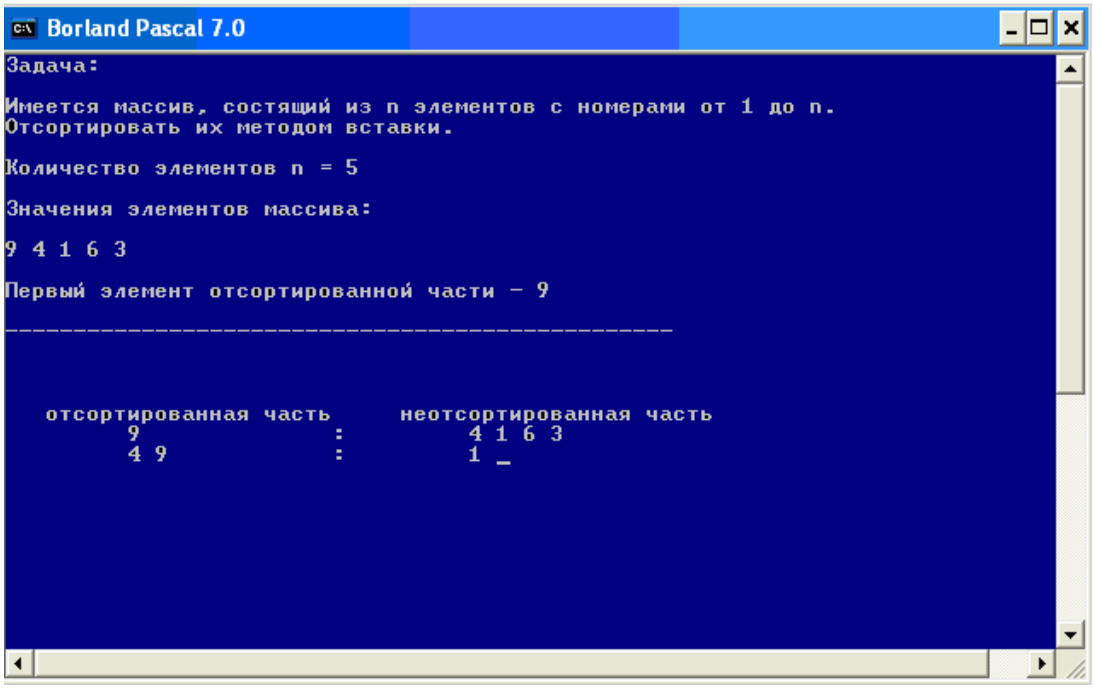


Рис. 7

Таким образом, обучающимся последовательно производится сортировка всего массива методом вставки (рис. 8).

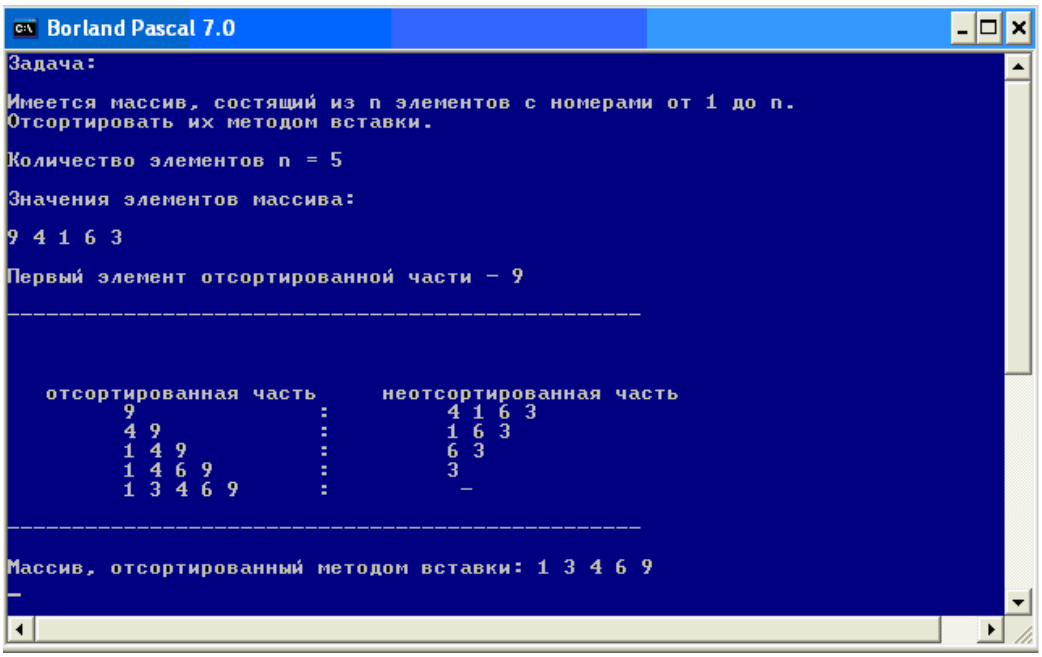


Рис. 8

3 этап - творческий:

На данном этапе обучающиеся самостоятельно разрабатывают программу на процедурном и объектно-ориентированном языке программирования для решения задач по теме «Сортировка массива методом вставки».

Результатом обучения должна стать программа, реализованная на Паскале:

```
Usescrt;
Constnmax=100;
var a:array[1..nmax]ofinteger;
  n,i,j,p,k:integer;
begin
  clrscr;
  randomize;
  repeat
            write('Размермассивадо ',nmax,' n=');
            readln(n);
  until n in[1..nmax];
  writeln('Массив:');
  for i:=1 tondo
```

```
   begin
   a[i]:=random(20);
write(a[i]:4);
   end;
   writeln;
   writeln('Сортировка методом вставки:');
   for i:=2 tondo
   for j:=1 to i-1 do
              if a[i]<a[j]then
                  begin
                   p:=a[i];
                   for k:=i-1 downto j do
                              a[k+1]:=a[k];
                              a[j]:=p;
                 end;
   for i:=1 tondo
   write(a[i]:4);
   readln
end.
```

А также ее визуальное представление с помощью языка Delphi (рис.9):

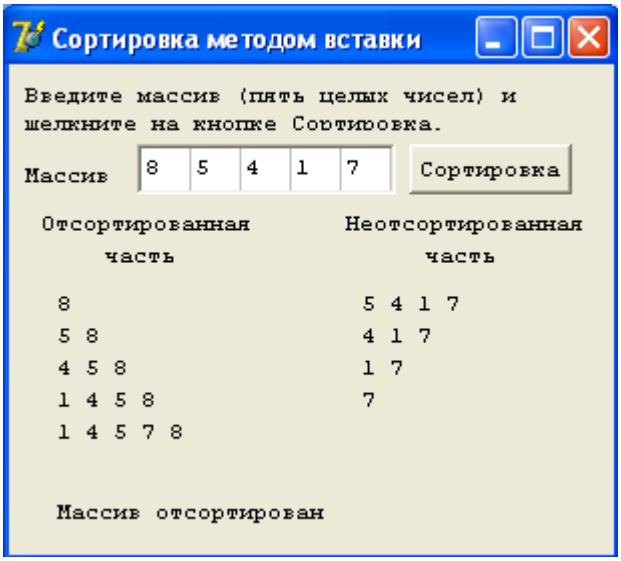


Рис. 9

По такому же принципу организовывается изучение всех остальных методов сортировки массива.

Таким образом, с помощью данной методики происходит обучение программированию с использованием двух парадигм. Использование представленной методики позволит наиболее эффективно организовать образовательный процесс, направленный на формирование компонентов программистского стиля мышления.

Приложение 4

Сравнительная характеристика парадигм программирования

Парадигмы / Элементы	Императивная	Логическая	Объектно-ориентированная	Функциональная
Программа	Алгоритмы + структуры данных	Логика + управление	Совокупность взаимодействующих объектов	Вычислительные выражения
Типы данных	Скалярные, структурные	Атомы, структурные	Скалярные, структурные	Атомы, списки
Обработка данных	Присваивание, передача по значению, передача по ссылке	Связь переменных при помощи унификации (однократное присваивание, передача параметров, размещение записей, доступ к полям записи для одновременного чтения и записи)	Присваивание, действия над объектами	Значение функции, передача по значению, декомпозиция функции
Управление выполнением программы	Последовательность команд, ветвление, циклы, вызов функций, рекурсия	Отсечение, откат, рекурсия	Последовательность команд, ветвление, циклы, вызов функций (методов), рекурсия,	Вызов функции, вычисление функции, условные вычисления, включая рекурсивные
Элементы программы	Команды, блоки процедуры, функции	Факты и правила	Объекты, классы	Функции, LET-блоки, функционалы
Переменная	Выделяется участок памяти для объявленных переменных	Логические переменные соотносятся с объектами, а не с ячейками памяти, Единственный, но не определенный объект	Выделяется участок памяти для объявленных переменных	Переменная обозначает только имя структуры
Область действия переменных	Локальные и глобальные	Область, одно предложение	Глобальные открытые, глобальные закрытые	Локальные, свободные, предложение
Транслятор	Компилятор	Интерпретатор	Компилятор	Интерпретатор, компилятор
Область применения	Программы общего назначения	Системы ИИ, ЭС	Разработка больших пользовательских приложений	Символьная обработка, ИИ системы автоматизированного проектирования, программирование игр