Министерство науки и высшего образования РФ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Уральский государственный педагогический университет»
Институт математики, физики, информатики и технологий
Кафедра информатики, информационных технологий
и методики обучения информатике

Методика обучения школьников решению олимпиадных задач по информатике и ИКТ

Выпускная квалификационная работа
бакалавра по направлению подготовки
44.03.01 – Педагогическое образование
Профиль «Информатика»

Исполнитель: студентка группы ИНФ-1501z
Института математики, физики,
информационных технологий
Пучкова Дарья Андреевна

_____
подпись

Руководитель: к.п.н., доцент кафедры
ИИТиМОИГазейкина Анна
Ивановна

_____
подпись

Квалификационная работа
допущена к защите
«____»_____ 2020 г.

Зав. кафедрой _____

Екатеринбург – 2020

Оглавление

Введение

Одной из важнейших задач в сфере образования является работа со школьниками, имеющими выдающиеся способности в области информатики. В связи с быстрым развитием информационных технологий, усиленным интересом школьников, олимпиады по информатике и ИКТ становятся одни из лидирующих.Терабайты информации и миллионы страниц олимпиадных задач и методов их решения публикуются не первый год в книгах и Интернете. Задания на олимпиадах чаще всего имеют простую формулировку, но не простое решение, требующее творческого мышления.

Таким образом, актуальность исследования состоит в необходимостисформировать и развить интерес у школьников к олимпиадному программированию, при этом развивая творческие и интеллектуальные способности.

Объект исследования: процесс обучения олимпиадной информатике в средней общеобразовательной школе.

Предмет исследования: методические рекомендации по подготовке учащихсяк решению олимпиадных задач по программированию.

Цель исследования: разработатьучебное пособие и методическиерекомендаций по подготовке учащихся 7-х классов к олимпиадам по программированию.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Проанализировать особенности организации внеклассной работы по информатике в современной школе.
2. Изучить особенности олимпиад по информатике и ИКТ как одной из форм внеклассной работы по информатике.

3. Проанализировать педагогический опыт подготовки школьников к олимпиадам по программированию.

4. Разработать учебное пособие и методические указания по подготовке к олимпиадным задачам по информатике.

5. Произвести апробацию разработанных материалов.

Выпускная квалификационная работа состоит из введения, двух глав, заключения, списка литературы.

ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ПОДГОТОВКИ УЧАЩИХСЯ 7-Х КЛАССОВ К ОЛИМПИАДАМ ПО ПРОГРАММИРОВАНИЮ

1.1.Проанализировать особенности организации внеклассной работы по информатике в современной школе

Одним из важных этапов в жизни человека является школа, в которой проводится большая часть времени. Известно, что школьное образование разделено на этапы, формируя информационную культуру и овладевая основами информатики в процессе обучения в школе. Для того чтобы с детства привить информационную культуру, которая, с большой вероятностью, понадобится всем, нужно начинать с начальных классов.

В любом учебном заведении, в каком бы педагог не работал, он знает, что успех в проведении беседы, урока или лекции напрямую зависит от создания активности в обучении. Большой прогресс в обучении создается благодаря интересу у учащихся, приводящий их к активной деятельности. Однако не так-то просто на начальной стадии пробудить глубокий интерес к занятиям и самой потребности в самообразовании. Некоторые учителя убеждены, что если дети слушают учителя, значит, урок проходит хорошо, не задумываясь над тем, как ученики работали на протяжении урока, какие направления выбрали для развития творческой стороны.

При обучении мы должны обращать внимание учащихся на развитие их личностных качеств, когда учащиеся активно действуют, они будут усваивать знания, навыки и умения. В данном случае, задача учителя сделать информацию не только полезной, но и интересной для учащихся.

Информатика – это интенсивно развивающаяся, молодая наука, уже в начальной школе необходимо закладывать фундаментальные основы.

Именно поэтому желательно начинать изучение информатики с первого класса.

Проведение внеклассных мероприятий необходимо для закрепления интереса школьников к предмету информатики. Ежегодно в школах проводят различные олимпиады, конкурсы ученических программ, где участники проявляют свое логическое и абстрактное мышление, показывают свои способности в области программирования.

В частности на уроке воспитывают ответственное отношение к учебе, интереса к знаниям, увлеченности наукой. Школьная программа ограничивает учителя, в основном только через внеклассную работу можно полностью удовлетворить интересы и запросы школьников.

Выделим несколько целей внеклассной работы по информатике:

1. Увлечь учащихся предметом, повысить их интерес к информатике.
2. Выявить способности к информатике, в частности к программированию, и творческий потенциал ребенка.
3. Привить учащимся интерес к исследовательской работе.
4. Сформироватьнавыки работы с компьютером и умения работать с любым видом информации на ПК, при этом грамотно организовывая свою информационную деятельность (подбирать материал на определенную тему, пользоваться дополнительной литературой);

Подразумевается, что учитель должен осуществить реализацию этих целей на уроке. Но, в процессе учебных занятий, которые имеют ограничения в виде рамок учебного времени и программы, сделать это с достаточной полнотой не удается. Следовательно, полная реализация данных целей достигается на внеклассных занятиях.При условии обеспечения учащихся

образовательного процесса доступом к средствам информационных и коммуникационных технологий, а именно, компьютерам, периферии, сети.

Связь между учебно-воспитательной работой, которая проводится на уроках и внеклассным обучением имеет довольно тесную связь. Учебные занятия развивают интерес у учащихся к знаниям, что оказывает помощь в формировании представления о внеклассных занятиях. Внеклассная работа, в свою очередь, позволяет учащимся применять теоретические знания на практике, расширяя и углубляя их. Однако учебно-воспитательная и внеклассная работы должны быть самостоятельными по отношению друг к другу, чтобы не превратить внеклассную работу в дополнительные занятия.

Внеклассные занятия оказывают положительное влияние, ведь учащиеся более углубленно и досконально работают с учебным материалом, изучая дополнительную литературу. Ребята пытаются самостоятельно продолжить изучение информатики и вычислительной техники, осваивают работу с компьютером.

Внеклассная работа не регламентируется государственной программой. Материал, который разбирается на занятиях, составляется в соответствии со знаниями и умениями учащихся. В зависимости от содержания и формы проведения занятия могут длиться от пяти минут до двух часов. Состав учащихся на таких занятиях строго не определен, это могут быть ребят из одного или разных классов. Виды внеклассной работы многообразны, к ним мы относим кружки, викторины, групповые занятия, олимпиады.

Для внеклассной работы по информатике нужен специально оборудованный кабинет, который будет оснащен техническим оборудованием и немало важным аспектом – обычной и научно-популярной литературой. Для такого помещения необходимо, соответствующее современному уровню развития вычислительной техники, наличие

достаточного количества компьютеров;наличие периферийных устройств, таких как принтер, системы мультимедия; локальной и глобальной сети.

Внеклассная работа характеризуется разнообразием форм. Одна из таких форм ведения внеклассных занятий по информатике является разделение на две части. Где первая часть будет посвящена повторению старого и изучению нового материала, и индивидуальной работе учащихся по заданиям теоретико-практического плана. Заключительной частью данного этапа будет задано домашнее задание по изучению теории. Второй же частью такой формы будут являться обсуждения и решения задач повышенной трудности.

Накоплен огромный опыт внеклассной работы по различным дисциплинам. Умелое сочетание опыта внеклассной работы со спецификой предмета информатики обеспечат успех проведения внеклассных занятий по информатике. Формы внеклассной работы классифицируются по разным признакам: охвату учащихся, времени проведения, систематичности, дидактической цели и т.д. по систематичности можно выделить эпизодические внеклассные мероприятия и постоянно действующие внеклассные организации (работающие, по крайней мере, в течение учебного года). [6]

К эпизодическим внеклассным мероприятиям отнесем:

1. Проведение викторин, вечеров, тематических семинаров и конференции.
2. Разработка и выпуск стенгазеты.
3. Подготовка и участие к школьным, городским и др. олимпиадам.
4. Летние компьютерные лагеря.

К постоянно действующим внеклассным занятиям можем отнести следующее:

1. Разнообразные факультативы и кружки.
2. Научные общества в школе.
3. Организация дистанционного обучения.

Приведенное деление внеклассных мероприятий, как свидетельствует практика школ, имеет условный характер: в живой внеклассной работе нередко одни формы порождают другие, разветвляются на несколько новых форм. Например, хорошоработающий кружок может выступить организатором вечера или инициатором выпуска стенной газеты по информатике. [7]

Во всех видах внеклассных занятий есть индивидуальная работа, и выражается она в подготовке материала для стенгазеты, викторины, вечера, конференции, в чтении литературы и другом.

Проведение вечеров, конференций, конкурсов, олимпиад выражается в массовой работе, а многообразие форм внеклассной работы делает ее интересной.

Можно сделать вывод, что внеклассные занятия способствуют формированию исследовательских навыков, творческого воображения, углубленному изучению вопросов учебной программы, вызывающих интерес у учащихся.

На коллегии Министерства образования Российской Федерации, которая состоялась 22 февраля 1995 года, обсуждался ход реализации программы информатизации образования на 1994-1995 гг. был рассмотрен вопрос о совершенствовании организации обучения информатике в общеобразовательной школе на современном этапе. Коллегия постановила признать целесообразной необходимость выделения нескольких этапов в овладении основами информатики и формировании информационной культуры в процессе обучения в школе:

- первый этап (1-6 классы) – пропедевтический;

- второй этап (7-9 классы) – базовый курс;

- третий этап (10-11 классы) – профильные курсы. [1]

На первом этапе прививаются элементы информационной культуры, происходит освоение некоторых прикладных аспектов информатики, что дает возможность использовать полученные знания в дальнейшем обучении информатике. Первый этап посвящен первоначальному знакомству школьников с самим компьютером, простейшими текстовыми программами, тренажерами, играми.

При организации обучения нужно учитывать психолого-педагогические особенности ребят, в младших класс не все ребята готовы к освоению компьютерной техники, так как имеют разную подготовку к ее изучению, индивидуальные особенности, обычно дети в таком возрасте еще не совсем усидчивые.

Учащиеся начальной школы обычно мыслят конкретно, их основным видом деятельности является игра. Следовательно, уроки лучше проводить в игровой форме, которые будут в дальнейшем приводить к перестройке восприятия и памяти, развивая способности абстрагирования. В данном промежутке времени школьник получает знания об информации, ее способах представления, формах, свойствах, кроме этого, предмет развивает логику.

На протяжении второго этапа проходит обучение основному общеобразовательному минимуму подготовки школьников. Школьники формируют представление о хранении, передаче информации в обществе; овладевают методами и средствами технологии решения задач; происходит формирование навыков сознательного пользования ПК в своей учебной деятельности, компьютерной грамотности, подготовка к практической деятельности информационных технологий.

На данном этапе происходит дальнейшее изучение абстрактного мышления: понятий, представлений, умозаключений, суждений. Пробуждение развития логического мышления и их использование школьником происходит через познание теоретических разделов, в которых изучается устройство компьютера, системы счисления, решение алгоритмических задач, программирование. Информатика, а в частности, программирование является интересным инструментом для развития интеллектуальных способностей.

Продолжением базового курса считается третий этап обучения, он же считается профильным, в котором предметно рассматривается информатика и информационные технологии, где разграничиваются по объему и содержанию, обуславливается это дальнейшим профессиональным занятием школьников.На данном этапе происходит углубление в программирование, методы вычислительной математики, моделирование, тестирование, способы представления информации, телекоммуникации и другие профильные занятия. При этом формируется интеллектуальные качества, уникальность склада ума, что в большей степени подготавливает к последующей профессиональной деятельности.

Основными целями курса информатики в средней школе, по мнению доктора педагогических наук А.А. Кузнецова, являются:

1. Овладение школьниками компьютерной грамотностью, которая включает не только навыки работы на компьютере и умения алгоритмизации, но и умение решать задачи с помощью компьютера, используя при этом информационное моделирование;
2. Формирование у школьников основ информационной культуры, куда включено изучение фундаментальных основ информатики. [2]

Информатика сочетает в себе несколько основных направлений в обучении данному предмету и отражает не малозначимые аспекты

образовательной значимости. Так одна из целей связана с формированием представлений об информационном подходе к анализу мира, о роли самоуправляемых систем. Другаяцель связана с формированием компьютерной культуры, творческого подхода школьников, развитием их мышления.

Исходя из вышесказанного, под внеклассной работой подразумевается добровольное внеурочное занятие по предмету учащимися, а именно, это занятие, проходящее в школе после уроков, которое не заключается в усвоении обязательного минимума теоретического материала, предусмотренного программой. Внеклассная работа тесно связанна между необязательными занятиями школьника по предмету и работой учителя по организации этих занятий. Практика показывает, что внеклассная работа является поддержкой в повышении качества подготовки учащихся по информатике, так как количества отведенных часов на занятия информатикой недостаточно.

1.2. Олимпиады, как одна из форм внеклассной работы по информатике

Школьные олимпиады выявляют одаренных учащихся, развивают творческий интерес к решению нестандартных задач, мотивируют к углубленному изучению предмета, и играют важную роль в формировании высокопрофессиональных кадров.

История интеллектуальных конкурсов насчитывает не одно столетие, Астрономическое общество Российской Империи еще в XIX веке устраивало «олимпиады для учащейся молодежи». В Петербурге в начале 1880 года состоялся VI съезд врачей и естествоиспытателей, где собралось более 30 астрономов. В протоколе заседания было занесено следующее:

«Необходимость Русского астрономического общества ощущается уже давно всеми русскими астрономами. В России нет журнала, где бы они могли всегда печатать свои статьи, сообщать свои наблюдения и проч., наконец, нет собраний, где бы астрономы могли сходиться и обмениваться мыслями».[8]

В 1890 году министром народного просвещения был утвержден проект Устава Русского астрономического общества. В 1886 году начали проводить заочные олимпиады. Конкурс по решению задач «Вестник опытной физики и элементарной математики» в 1885-1917 гг. был одним из прообразов современных олимпиад.

Однако только в 1934 году Ленинградский университет организовал состязание юных математиков, и тогда состоялась первая олимпиада в привычной для нас форме.

История олимпиадного движения отражает эволюцию общества и системы общего образования, позволяет увидеть, как были расставлены акценты в системе образования.

Школьные предметные олимпиады – это соревнование учащихся на лучшее выполнение какого-либо задания. За последние два десятилетия олимпиадное движение приобрело по-настоящему массовый характер. Сегодня нет ни одного предмета, по которому бы не проводились олимпиады. В олимпиадном движении появились совершенно новые предметы, такие как экономика, информатика, право, экология.

Олимпиады по информатике начались относительно других олимпиад недавно. В конце 70 годов началось развитие компьютерной техники, и наступила эра новых информационных технологий. Первая Всесоюзная олимпиада по информатике прошла в апреле 1988 года в Свердловске, 80 школьников из всех союзных республик приняли в ней участие. Проведение олимпиады именно в этом городе было выбрано не случайно: на тот момент компьютеры «Роботрон» уже были установлены во многих школах уральской столицы.

Олимпиада состояла из теоретического тура, в котором участникам представлялось четыре задачи, и практического, в ходе которого необходимо было решить две задачи на компьютерах, а языками для написания использовались Бейсик и Паскаль. Членами жюри были приглашены лучшие специалисты в области школьной информатики, которые провели значимую работу в олимпиадном движении, разработали методику проведения олимпиад, их содержание.

Историческое Постановление ЦК КПСС и Совета Министров СССР от 28 марта 1985 г. № 271 «О мерах по обеспечению компьютерной грамотности учащихся средних учебных заведений и широкого внедрения электронно-вычислительной техники в учебный процесс» [9]. Тогда и началось во всех школах страны преподавание курса «Основы информатик и вычислительной техники».

С 1989 года стали проводить Всероссийскую олимпиаду, которую провели в Красноярске, где на тот момент было лучшее компьютерное оснащение. Олимпиада делилась на два тура, один из которых был теоретическим, а другой практическим и предполагал использование компьютеров. Но спустя два года оба тура стали практическими. При проверке заданий специалисты столкнулись с проблемой их апробации. Исходя из того, что задания проверялись вручную, тут же, при ученике, требовалось множество специалистов для их проверки.Только с 1999 года запустили процедуру тестирования полностью автоматической. Начиная с 1992 года, Всероссийская олимпиада проводилась в городах Троицк и Санкт-Петербург, а с 2000 годов ежегодно проводится в разных городах России, это связано с хорошей оснащенностью техникой, со стремительным развитием экономики страны и информационной потребностью.

История олимпиады по программированию берет начало в 2002 году. Данная олимпиада с 2002 по 2006 года проводилась как заочный тур Московской олимпиады школьников по информатике, на очный тур данной олимпиады приглашались только те, кто показал хороший результат. В 2006 году заочный тур был преобразован во Всероссийскую заочную олимпиаду по программированию. Помимо такой олимпиады, в 2005 году, для школьников, недавно начавших программирование (7-9 классов) стали проводить Московскую олимпиаду по информатике. Которая в 2009 году была переименована в Открытую олимпиаду школьников по программированию.

На самом деле олимпиада по информатике очень обширна и включает в себя задания по программированию, информационным технологиям, систем счисления.

Одним из главных показателей развития образования являются международные предметные олимпиады. На данный момент Международная

олимпиада по информатике IOI (IntenationalOlympiadinInformatics, МОИ) занимает важное место среди них в силу интенсивного развития как школьного предмета информатики (computerscience) в большинстве стран мира, так и востребованности современных информационных технологий в подготовке высококвалифицированных кадров в сфере высоких технологий.[3]

Первостепенной задачею олимпиады является выявление и работа с талантливыми учащимися. Взаимная работа учитель-ученик это то,что принесет половину успеха на олимпиаде. В идеальном случае - подготовить учащегося максимально раскрывая его интеллектуально-творческий потенциал, сохраняя мотивацию и психологическое здоровье. Олимпиады по информатике одни из трудоемких, требующие определенных навыков, фантазии, умения сосредоточиться на задании, доведении выполнение некоторых задач до автоматизма. Исходя из всего вышеперечисленного, необходимо нарабатывать (тренировать) ученика на выбор последовательности решения задач, на конкретные приемы, на выполнение задач на время, планирование времени, на овладение приемам решения типовых задач, на умение последовательно работать в условиях стресса.

Из года в год олимпиадное движение становится всё более востребовано, но и вопрос как лучше подготовиться к олимпиаде все еще остается актуальным. Ведь задачи олимпиадного типа принципиально отличаются от задач школьного курса, чтобы подготовить школьника к олимпиаде, как говорится, с «нуля», потребуется не один год. Еще одним важным критерием является то, что олимпиады проходят в ограниченное время, и не каждый учащийся обладает достаточной быстротой принятия решения. Задачи по информатике требуют нестандартного, творческого подхода, не каждому школьнику, учащемуся на оценку «отлично» следует участвовать в олимпиадах. В олимпиадах все чаще встречаются задания на программирование и алгоритмизацию. Школьная программа включает в себя

только несколько разделов, чего совсем недостаточно для получения значимого пласта знаний.

Выявление и развитие у учащихся творческих способностей, заинтересованности к научно-исследовательской деятельности, распространение научных знаний, является одними из целей олимпиад.

Следовательно, для участников олимпиад необходимы дополнительные знания и умения, помимо школьного курса, которые будут отличаться объемом и глубиной материала.

Специфика олимпиад по информатике. Всероссийские олимпиады и их этапы.

По инициативе и под эгидой Министерства образования и науки Российской Федерации ежегодно проводится Всероссийская олимпиада школьников по информатике, что считается более представительным форумом творческой молодежи, обучающейся в общеобразовательных учреждениях России.

Ключевым для школьных олимпиад стал 2007 год, когда на правительственном уровне определили «Порядок проведения олимпиад школьников» и принято «Положение о Всероссийской олимпиаде школьников».Сейчас же Всероссийская олимпиада является неотъемлемой частью системы образования, в которой принимают участие более шести миллионов учащихся, с разных уголков России. Проводится олимпиада ежегодно по 24 предметам.

Всероссийская олимпиада школьников по информатике состоит из четырех этапов и охватывает образование на разных уровнях:

- ✓ школьный;
- ✓ муниципальный;
- ✓ региональный;

- ✓ заключительный.

Организация олимпиад является непростой, а на уровне школ и районов даже сложной. Ведь для проведения требуется существенное количество персональных компьютеров, вовлечение квалифицированных специалистов по информационным технологиям, организация оргкомитета и жюри.

Школьный этап олимпиады организуют образовательные организации, муниципальный этап олимпиады – органы местного самоуправления муниципальных и городских округов в сфере образования, региональный этап – органы государственной власти субъектов Российской Федерации в сфере образования, заключительный – Министерство образования и науки.

Центральный организационный комитет олимпиады, руководствуясь в своей деятельности «Положением о всероссийской олимпиаде школьников» осуществляет организационно-методическое обеспечение Всероссийской олимпиады. Федеральным агентством по образованию утверждается состав комитета, центральной предметно-методической комиссии.

Олимпиада, обычно, проводится в два тура – теоретического и практического. Теоретический тур заключает в себя выполнение заданий по решению задач и составлению алгоритма. Практический тур включает в себя написание программы на одном из языков, отладка его на компьютере и выполнение её апробации.

Участвовать в школьном этапе может любой учащийся, окончивший начальную школу. Для того, чтобы пройти на следующий этап, необходимо преодолеть необходимый порог баллов, устанавливающийся организаторами олимпиады. В муниципальном этапе могут принять участие учащиеся 7-11 классов. Региональный и заключительный этапы предусмотрены для учащихся 9-11 классов.

Школьный этап проводится с сентября по октябрь, по заданиям, разработанным предметно-методическими комиссиями муниципального этапа. Муниципальный этап, в свою очередь, проводится с ноября по декабрь, участие в нем принимают учащиеся, которые стали победителями и призерами предыдущего этапа этого года и предыдущего учебного года, по заданиям, разработанным предметно-методическими комиссиями регионального этапа. В региональном этапе участие принимают с января по февраль учащиеся, ставшие победителями и призерами предыдущего этапа и победители школьного этапа текущего учебного года из числа обучающихся образовательных организаций Российской Федерации, расположенных за пределами территории РФ, по заданиям, разработанным предметно-методической комиссии Олимпиады. Заключительный этап проводят в марте-апреле и участвуют в нем победители и призеры регионального этапа, победители и призеры заключительного этапа предыдущего учебного года.

Согласно пункту 24 Порядка проведения Всероссийской олимпиады школьников, утвержденного приказом Министерства образования науки России от 18 ноября 2013 г.

Центральный оргкомитет олимпиады устанавливает квоты победителей и призеров заключительного этапа олимпиады, которые составляют не более 45 процентов от общего числа участников заключительного этапа олимпиады по каждому общеобразовательному предмету, при этом число победителей заключительного этапа олимпиады не должно превышать 8 процентов от общего числа участников заключительного этапа олимпиады по каждому общеобразовательному предмету[10].

Экзамен государственной итоговой аттестации без ее сдачи будет засчитан победителям и призерам региональных этапов Всероссийской олимпиады. Имея хорошие результаты на заключительном этапе Всероссийской олимпиады школьников можно получить льготы при

поступлении в виде дополнительных баллов за ЕГЭ или портфолио до зачисления в профильные вузы без экзаменов.

1.3 Анализ педагогического опыта подготовки школьников к олимпиадам по программированию.

В настоящее время существует большое количество различных источников для подготовки школьников к олимпиадам, от различных статей до сетевых ресурсов. Так же существует множество методов обучения. Попытаемся проанализировать педагогический опыт подготовки к олимпиадам других учителей.

Первым рассмотрим пособие 2008 года «Подготовка школьников к олимпиадам по информатике с использованием веб-сайта» Алексеева А.В. и Беляева С.Н.В данном пособии описана технология подготовки школьников к олимпиадам по информатике с использованием сайта в сети Интернет.

Пособие ориентировано на школьников 7-11 классов, студентов вузов, школьных учителей информатики и преподавателей программирования различных учебных заведений дополнительного и профессионального образования[12].

Материал разделен на два основных пункта: «Курс начинающего олимпиадника» и «Архив задач».

В «Курс начинающего олимпиадника» для самостоятельного изучения включен набор тем, каждая из которых включает теоретическую часть и практическую, в виде набора задач, сложность которых постепенно возрастает. В пособии задачи разделены на девять тем:

1. Введение. Понятие олимпиадной задачи. Работа с файлами.
2. Разветвляющиеся алгоритмы.
3. Циклические алгоритмы.
4. Массивы.
5. Сортировка.
6. Двумерные массивы.

7. Системы счисления.
8. Теория чисел.
9. Длинная арифметика.

В каждой теме кратко описаны необходимые определения и приведены примеры описания кода, дана ссылка на список задач в архиве задач, которые решаются в этой теме, что очень удобно для самостоятельного изучения. Рассмотрим на примере темы 03: Циклические алгоритмы, как выглядит это структурно.

Тема 03: Циклические алгоритмы.

Сложно представить серьезную программу без циклов. Мы предлагаем, что вы уже знакомы с циклами. В этом разделе будет рассмотрен ряд задач, для решения которых необходимо использовать циклы.

Напомним, что существуют три вида циклов:

1. оператор цикла с параметром

fori = 1..n;

операторы;

2. оператор цикла с предусловием

while (Условие) {

операторы;

}

3. оператор цикла с постусловием

do {

операторы;

} while (Условие);

Список задач:

Задача 106: Монетки.

Задача 081: Арбузы.

Задача 043: Нули.

Задача 063: Загадка.

Задача 002: Сумма [12].

В «Архиве задач» приведены задачи для самостоятельного решения и словесно описаны их решения. Авторы рекомендуют самостоятельно попытаться решить задачи, и в случае, когда не получается, прибегнуть к просмотру решения. В «Архиве задач» каждая задача имеет название, время, память, сложность выполнения, условие задачи и разбор.

Рассмотрим задачу **002: Сумма.**

(Время: 1 сек. Память: 16 Мб.Сложность: 19%)

Требуется посчитать сумму целых чисел от 1 до N.

Входные данные

В единственной строке входного файла INPUT.TXTзаписано единственное целое число N, не превышающее по абсолютной величине 10^4.

Выходные данные

В единственную строку выходного файла OUTPUT.TXTнужно вывести одно целое число – сумму чисел от 1 до N.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	5	15

Разбор

В этой задаче можно воспользоваться формулой арифметической прогрессии $S_n=(a_1+a_n)*n/2$. Но школьники часто забывают эту формулу и в этом случае прибегают к вычислению этой суммы с помощью цикла и решают эту задачу, используя следующий алгоритм:

```
read (n);
s=0;
fori = 1.. n {
s=s+i;
}
write (s);
```

Но сложность этой задачи не в вычислении этой суммы и приведенный выше алгоритм не проходит на третьем тесте! Оказывается, здесь очень внимательно нужно прочитать условия задачи, особенно ограничения на число N. Дело в том, что это число может быть отрицательным! Учитывая это, попробуйте самостоятельно доработать и реализовать верный алгоритм решения этой задачи [12].

Авторами рекомендован раздел «Рекомендации учащимся» при выборе задач. В нем приведена таблица с номером и названием задач, тематикой и сложностью выполнения. По параметру «Тема» можно выбрать нужный раздел, в котором вы хотите порешать задачи или в котором вы не совсем

уверены при решении задач. По параметру «Сложность» рассчитывается рейтинг, и оцениваются силы в решении задач.

Также есть и раздел «Рекомендации учителям», в котором приведена таблица с номером и названием задач, тематикой и сложностью выполнения. Задачи упорядочены по тематике, а потом по сложности. Тематика задач приведена в порядке усложнения тем, достаточных для участия в школьных и муниципальных олимпиадах и которые используются на окружных и более высокого уровня олимпиадах

В конце пособия есть раздел «Тестирующие системы», в котором представлены тестирующие системы, которые существуют на сайтах ведущих вузов России и мира. Название вузов и адреса с задачами приведены в отдельной таблице. Авторы советуют порешать задачи на этих сайтах для совершенствования в области олимпиадного программирования.

Глава 2. РАЗРАБОТКА УЧЕБНОГО ПОСОБИЯ К РЕШЕНИЮ ЗАДАЧ И МЕТОДИЧЕСКИЕРЕКОМЕНДАЦИЙ ПО ПОДГОТОВКЕ К ОЛИМПИАДНЫМ ЗАДАЧАМ ПО ИНФОРМАТИКЕ

2.1 Разработка учебного пособия по подготовке к олимпиадным задачам по информатике.

В пособии рассмотрены задачи, необходимые при подготовке школьников к олимпиадам по информатике. Приведены методические рекомендации по работе с ним. Ориентировано на школьников 7-8 классов школьных учителей информатики. Пособие создано для развития у школьников навыков программирования и способностей, направленных на решение олимпиадных задач.

В последнее время большинство олимпиадных задач строят по такой классификации, как показано в таблице 1.

Направление задач	Обозначение
Задачи без программирования	Задачи на логику и расчеты, где решение не требует составления программного кода.
Арифметические задачи	Задачи требуют знания формул и умения применять их, программный код обычно небольшой.
Геометрические задачи	Задачи на проверку условий выполнения определенных свойств для заданных фигуры или тела, взаимодействия тел на плоскости и в пространстве.
Задачи на динамическое программирование	Задачи, направленные на выявление рекуррентных соотношений.
Задачи на сортировку и последовательность	Задачи на обработку данных, представленных в виде массивов.
Задачи на графы	Задачи со структурами данных, основанными на вершинах и ребрах.
Задачи на рекурсию	Задачи на поиск с рекурсивным перебором вариантов.

Некоторые задачи сложно отнести к какому-то определенному направлению, так как могут сочетать в себе сразу несколько.

Рассмотрим примеры задач по каждому направлению, располагая в порядке возрастания сложности.

Задачи без программирования.

1. Маша читает книги только братьев Гримм, в которых имеются цветные иллюстрации. Кроме того, для нее важен объем книги, в ней должно быть не меньше 200 страниц и не больше 250 страниц. Какие книги возьмет Маша в библиотеке, если ей предложены следующие:

№ п/п	Название книги	Иллюстрации	Число страниц	Автор
1	Красная шапочка	Цветные	230	Братья Гримм
2	Русалка	Цветные	199	Братья Гримм
3	Золотой ключик	Без иллюстрации	240	Братья Гримм
4	УрфинДжюс	Цветные	249	А. Волков
5	Белоснежка	Цветные	255	Братья Гримм
6	Гарри Поттер	Без иллюстрации	210	Д. Роулинг
7	Бременские музыканты	Цветные	180	Братья Гримм

Решение: для начала отсортируем по столбцу «Иллюстрация», нам подходят позиции 1, 2, 4, 5 и 7. Затем отберем по столбцу «Число страниц», нам подходят 1, 3, 4 и 6. Но 3 и 6 позиции не подходят, так как они без иллюстрации. Таким образом, у нас остаются позиции 1 и 4, проверим их по столбцу «Автор», по столбцу «Автор» нам подходит красная шапочка, 1.

Ответ: 1.

2. В доме у Пети установили новый лифт экспериментальной модели. В этом лифте все кнопки с номерами этажей заменены двумя кнопками. При нажатии на одну из них лифт поднимается на один этаж вверх, а при нажатии на вторую – опускается на один этаж вниз. Пете очень понравился новый лифт, и он катался на нем, пока не побывал на каждом из этажей хотя бы по одному разу.

Известна последовательность кнопок, которые нажимал Петя: 1221221221. Каково количество этажей в доме у Пети? С какого этажа Петя начал кататься на лифте? На каком этаже Петя закончил свой эксперимент?

Решение: предположим, что 1 – это подняться наверх, а 2 – это спуститься вниз. Тогда, предположим, что Петя начал движение с 4 этажа, построим таблицу:

	1	2	2	1	2	2	1	2	2	1
5эт	*									
4эт*		*		*						
3эт			*		*		*			
2эт						*		*		*
1эт									*	

Ответ: 5 этажей в доме, Петя начал движение с 4 этажа и закончил движение на 2 этаже.

Арифметические задачи.

1. Сообщение о том, что ваш друг живет на 5 этаже, несет 4 бита информации. Сколько этажей в доме?

Решение: $N=2^i$; $2^4=16$ этажей.

Ответ: 16.

2. Проверить, поместится ли на диске компьютера музыкальная композиция, которая длится mминут и n секунд, если свободное дисковое пространство 6 мегабайт, а для записи одной секунды звука необходимо 16 килобайт.

Решение: переведем мегабайты в килобайты, так как 1 мегабайт = 1024 килобайта, то 6 мегабайт = 6144 килобайт. Обозначим t – время, в секундах, v – объем файла, в килобайтах, тогда:

t=60*m+n;

v=16*t.

Программа на Паскале будет иметь вид:

```
programmuzi;
```

```
varm, n, t, v: integer;
begin
writeln (‘Введитемин’);
readln (m, n);
t:=60*m+n;
v:=16*t;
if v<=6144 then writeln (‘Композицияпоместится’)
elsewriteln (‘Нехватает ’, v-6144, ‘ килобайт’);
end.
```

Геометрические задачи.

1. Даны 2 точки A (x^1, y^1) и B (x^2, y^2). Определить, какой из отрезков, OA или OB, больше.

Решение:

```
programotr;
var x1, x2, x3, x4:integer;
s1, s2:real;
begin
writeln (‘введитех1 иу1’);
readln (x1, y1);
writeln (‘введитех2 иу2’);
readln (x2, y2);
s1:=sqrt(x1*x1+y1*y1);
s2:=sqrt(x2*x2+y2*y2);
if s1
if s1>s2 then writeln(‘OA>OB’);
if s1=s2 then writeln (‘OA=OB’);
end.
```

2. Фермер Иван с юности следит за своим газоном. Газон можно считать плоскостью, на которой в каждой точке с целыми координатами растет один пучок травы.

В одно из воскресений Иван воспользовался газонокосилкой и постриг некоторый прямоугольный участок газона. Стороны этого участка параллельны осям координат, а две противоположные вершины расположены в точках (x1, y1) и (x2, y2). Следует отметить, что пучки травы, находящиеся на границе этого прямоугольника, также были пострижены.

Довольный результатом Иван купил и установил на газоне дождевальную установку. Она была размещена в точке с координатами (x3, y3) и имела радиус действия струи r. Таким образом, установка начала поливать все пучки, расстояние от которых до точки (x3, y3) не превышало r.

Все было хорошо, но Ивана заинтересовал следующий вопрос: сколько пучков травы оказалось и пострижены, и полито в это воскресенье? Требуется написать программу, которая позволит дать ответ на вопрос Ивана.

lawn.in

Имя входного файла:

lawn.out

Формат входных данных

В первой строке входного файла содержатся четыре целых числа: x1, y1, x2, y2 ($-100\,000 \le x1 < x2 \le 100\,000$; $-100\,000 \le y1 < y2 \le 100\,000$).

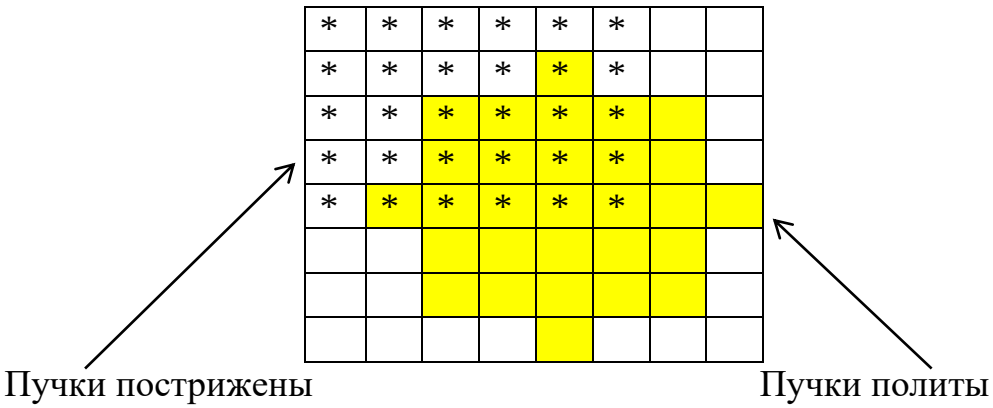
Во второй строке входного файла содержатся три целых числа: x3, y3, r($-100\,000 \le x3, y3 \le 100\,000$; $1 \le r \le 100\,000$).

Формат выходных данных

В выходной файл необходимо вывести одно целое число – число пучков травы, которые были и пострижены, и политы.

Пример входных и выходных данных

Иллюстрация к примеру



Для каждой прямой, содержащей все пучки травы с равной координатой по оси абсцисс (аналогично можно рассматривать и ось ординат), найдем ее точки пересечения с окружностью, отображающей область действия поливальной установки, если, конечно, такие точки существуют. Получив данные точки, не сложно за время O(1) посчитать количество пучков травы на данной прямой одновременно и политых, и постриженных. Это делается с помощью операции вычисления целой части числа. Далее, просуммировав найденные числа для всех прямых, можно найти ответ на поставленную задачу.

```
programgazon;

type

abc=Int64;

var

x1, y1, x2, y2, x3, y3, x:logint;

k, r, y:abc;

input, output:text;

functionkol (x, z1, z2:longint):longint;

var
```

```
min, max, k:longint;
begin
if (xx2) or (z1>y2) or (z2)
k:=0
else
begin
min:=y1;
if z1>y1 then
min:= z1;
max:=y2;
if z2
max:=z2;
k:=max-min+1
end;
kol:=k
end;
begin
assign (input, ‘input.txt’);
reset (unput);
assign (output, ‘output.txt’);
rewrite (output);
```

```
readln (input, x1, y1, x2, y2);
read (input, x3, y3, r);
k:=kol (x3, y3-r, y3+r);
y:=r;
for x:=1 to r-1 do
begin
whilesqr (x) + sqr (y) >sqr (r) do
y:=y-1;
k:=k+kol (x3+x, y3-y, y3+y) + kol (x3-x, y3-y, y3+y)
end;
k:=k+kol (x3+r, y3, y3) + kol (x3-r, y3, y3);
write (output, k);
close (input);
close (output);
end.
```

Динамическое программирование.

1. Вычислить значение суммы S=1/1!+1/2!+…+ 1/k!

Решение: можно, конечно, каждый раз вычислять очередное p!, затем 1/p!, и полученное слагаемое добавлять к сумме. Но обратим внимание на следующее очевидное равенство: 1/(p+1)! = (1/p!)/(p+1),

И программа вычисления запишется следующим образом:

```
programsum;
vari, p: longint;
```

```
S:real;
begin
S:=1;
p:=1;
for i:= 2 to k do
begin
p:= p/I;
S:=S+p;
end;
writeln('S=', S:8:2);
end.
```

2. Начисловой прямой школьник Ваня может идти вправо на одну или две единицы. Первоначально Ваня находится в точке с координатой 0. Определите количество маршрутов Вани, приводящих его в точку с координатой n.

Обозначим количество маршрутов Вани, ведущих в точку с координатой n, как F(n). Теперь вычислим функцию F(n). Заметим, что F(0)=1 (это вырожденный случай, существует ровно один маршрут из точки 0 в точку 0 – он не содержит ни одного прыжка), F(1)=1, F(2)=2. Как вычислить F(n)? В точку n школьник может попасть двумя способами – из точки n–2 при помощи прыжка длиной 2 и из точки n-1 прыжком длины 1. То есть число способов попасть в точку n равно сумме числа способов попасть в точку n-1 и n-2, что позволяет выписать рекуррентное соотношение: F(n) = F (n-2) + F (n-1) , верное для всех $n \geq 2$.

Решение на языке Pascal в виде рекурсивной функции:

```
function f (n: longint): longint;
begin
if n < 2 then
```

```
f:=1
else
f:= f (n-1) + f (n-2);
end;
```

Вычисление решения этой функции, например, для n=20, оказывается, что функция работает крайне медленно. То есть одна из причин неэффективности рекурсивного решения – одно и то же значение функции вычисляется несколько раз, так как оно используется для вычисления нескольких других значений функции.

В данном случае, значения рекурсивной функции совпадает с числами Фибоначчи, так как вычисляются по тем же рекуррентным соотношениям. Для вычисления чисел Фибоначчи можно использовать цикл.

Решение:

```
var
F: array [0..100] of longint;
i, n: longint;
begin
readln (n);
for i:= 1 to n do
F [i]:=0;
F[0]:=1;
F[1]:=1;
for i:= 2 to n do
```

```
F[i]:= F [i – 1] + F [i – 2];

writeln (F [n]);

end.
```

Динамическое программирование использует те же рекуррентные соотношения, что и рекурсивное решение, но в отличии от рекурсии в динамическом программировании значения вычисляются в цикле и сохраняются в списке.

Сортировка и последовательность.

1. Дан числовой массив A(30). Определить, положительных или отрицательных элементов больше в этом массиве.

Решение: учтем, что значение элемента массива может быть равно нулю, его не учитываем.

```
programmassiv;
var a:array [1..30] of integer;
i, n, k:integer;
begin
for i:= 1 to 30 do
begin
readln (a[i]);
if a[i] > 0 then n:= n+1;
if a[i] < 0 then k:= k+1;
end;
ifn>kthenwriteln (‘положительныхбольше’);
ifn<kthenwriteln (‘отрицательныхбольше’);
ifn=kthenwriteln (‘положительных и отрицательныхпоровну’);
end.
```

2. Симметричной называют последовательность чисел, которая одинаково читается как слева направо и наоборот. Например, следующие последовательности являются симметричными:

11222211, 987656789

Вашей программе будет дана последовательность чисел. Требуется определить, какое минимальное количество и каких чисел надо приписать в конец этой последовательности, чтобы она стала симметричной.

Входные данные: в первой строке содержится число N – количество элементов исходной последовательности. Во второй строке записано N чисел, разделенных пробелом – элементы этой последовательности. $1 \leq N \leq 100$, элементы последовательности – натуральные числа от 1 до 9.

Выходные данные: в первой строке число M – минимальное количество элементов, которое надо дописать к последовательности, во второй строке M чисел, которые надо дописать к последовательности.

Решение:

```
programsum;
vara, b: array [1..300] of integer;
i, j, n, k: integer;
functioncheck: boolean; //проверяет последовательность на симметричность
var i: integer;
begin
result:= false;
for i:= 1 to n+k do
if b[i] <> b [n+k-i+1] then exit;
result:= true;
end;
begin
assign (input, ‘input.txt’);
reset (input);
```

```
assign (output, ‘output.txt’);
rewrite (output);
read (n); //считываеммассив
for i:= 1 to n do read (a[i]);
fork:= 0 tondo //перебираем всевозможные добавления
begin
fullchar (b, sizeof (b),0); //очищаеммассивb
b:= a; // копируем массив ав массив b
for i:= n+1 to n+k do //добавляеммассивbкчислу
b[i]:= a{n+k-i+1];
ifcheckthen// проверяем массив на симметричность
begin
writeln (k); // если симметричная последовательность получена, то
fori:= n+1 ton+kdo //выводим количество добавленных чисел и сами числа
write (b[i],’’);
close (input);
close (output);
halt (0); //прерываем работу программы
end;
end;
close (input);
close (output);
end.
```

Графы.

1. Каждый элемент квадратной матрицы размеренности N*N равен нулю, либо единицы. Найти количество групп, образованных единицами. Единицы относятся к одной группе, если из одной из них можно перейти к другой «наступая» на единицы, расположенные в соседних клетках. Соседними являются клетки, граничащие по горизонтали или вертикали.

Входные данные: в первой строке файла input.txtзаписано натуральное число N не больше 100 – размер квадратной матрицы. В следующих N строках задаются элементы матрицы через пробел.

Выходные данные: output.txt выведите единственное число – количество групп.

Решение: необходимо обойти матрицу и вычислить количество групп. После того, как мы попадаем в группу, надо это зафиксировать, увеличив переменную – результат на единицу. Чтобы второй раз не посчитать одну и ту же группу, сразу после посещения необходимо его уничтожить, то есть присвоить всем клеткам значение ноль.

Тест задачи не слишком мал, поэтому необходимо написать процедуру уничтожения группы, назовем ее “del”. Чтобы во время процедуры не выйти за пределы массива, сделаем его размером N+2 * N+2, а не размером N*N. Это дает нам возможность окружить искомый массив размером N*N нулями.

```
programgrup;
const m =102;
var a:array [1..m, 1..m] of integer;
i, j, n, rez: integer;
input, output: text;
procedure del (i, j: integer);
begin
ifa[i, j] <> 1 then
exit;
a [i, j]:= 0;
del (i+1, j);
del (i-1, j);
del (i, j+1);
del (i, j-1);
end;
begin
```

```
rez:= 0;
assing (input, ‘input.txt’);
reset (input);
assing (output, ‘output.txt’);
rewrite (output);
read (input, n); // заполняем массив нулям
for i:= 1 to n+2 do
for j:= 1 to n+2 do
a [i, j]:=0; //  считать матрицу из файла
for i:= 2 to n+1 do
for j:= 2 to n+1 do
read (input, a[i, j]); // обход матрицы в поиске групп
for i:= 2 to n+1 do
for j:= 2 to n+1 do
if a[i,j] =  1 then
begin
inc (rez);
count (i, j);
end;
write (output, rez);
close (input);
close (output);
end.
```

Рекурсия.

1. Вычислить n!

Решение:

```
program factorial;
var N: word;
function F (n:word): longint;
```

```
begin
if n=0 then F:=  else F:= n*F (n-1)
end;
begin
write (‘N=’);
readln (N);
writeln (‘N!=’, F (N))
end.
```

2. Дано целое число A. Вставить между некоторыми цифрами 1, 2, 3, 4, 5, 6, 7, 8, 9, записанными именно в таком порядке, знаки «+» и «-» так, чтобы значением получившегося выражения было число n. Например, если A=122, то подойдет выражение: 12+34-5-6+78+9.

Решение:

```
programzi;
var t: array [2..9] of string;
procedurezi_fra (s, r: longint; o:char; n:integer);
varnewr: longint;
begin
if o = ‘+’ then newr:= r+s else newr:= r-s;
if n <= 9 then begin
t[n]:= ‘+’; zi_fra (n, newr, ‘+’, n+1);
t[n]:= ‘-’; zi_fra (n, newr, ‘-’, n+1);
t[n]:= ‘ ’; zi_fra (s*10+n, r, o, n+1);
end
else//  проверкаравенства
ifnewr = A then begin
for i:= 2 to 9 do write (i-1, t[i]);
writeln (‘9’);
end;
end;
```

```
begin
readln (A);
zi_fra (1, 0, '+', 2);
end.
```

2.2 Методические указания по использованию учебного пособия по подготовке к олимпиадным задачам по информатике.

На занятиях по подготовке к олимпиадам учитель кратко объясняет теорию и предлагает порешать задачи по изученной теме, не объясняя решения ни одной из них. Ученики подумав, начинают предлагать свои идеи с решением этих задач. В случае возникновения затруднений у учеников, учитель может предложить некоторую идею по решению. Приветствуется обсуждение задач между учениками.

Задания по подготовке к олимпиадным задачам по информатике можно использовать для повторения, закрепления самостоятельных работ, соревнований.

В пособии задачи разделены на 6 разделов: задачи без программирования, геометрические задачи, динамическое программирование, сортировка и последовательность, графы, рекурсия. Каждый из 6 разделов содержит 10 задач с разъяснениями ответов.

Самым частым вопросом на олимпиаде у ученика возникает вопрос – с чего начать решение задачи? Процесс решения большинства задач можно разделить на следующие этапы:

1. Разобрать условие задачи;
2. Абстрагировать условие задачи;
3. Разработать алгоритм решения;
4. Реализовать программный алгоритм;
5. Провести апробацию программы.

Между ближайшими друг к другу этапами провести четкие границы довольно проблематично, а иногда и вовсе невозможно. Поэтому чаще всего их можно объединить на каком-то этапе, заменить или вовсе пропустить,

поскольку процесс решения задач является творческим процессом. Но выделить этапы стоит даже только потому, как это помогает увидеть полную картину и начать последовательно действовать при решении задач.

Первый этап является самым важным, ведь при разборе условия происходит определение, как будет решаться задача. Для правильного понимания условия и решения задач необходимо внимательно их читать, иначе можно начать решать совершенно другую задачу. При прочтении необходимо обратить внимание на каждое предложение по отдельности, не редкостью бывает, что в них кроется нужная и важная информация. На этом этапе участник определяет, с какой задачи он начнет, но может произойти и так, что задача, которая кажется легкой, на первый взгляд, в решении, на самом деле будет самой сложной, и лучше будет потратить время на понимание и решение другой задачи, более простой, оставляя больше времени на остальные. Даже если задача, кажется, аналогична той, которую участник уже решал, но на самом деле решение ее лежит совершенно в другом ключе.

На этапе абстрагирования условия задачи необходимо перейти от словесного описания текста к выбору схемы решения. Может быть и такое, что при представлении условия задачи, будут выявлены не состыковки при чтении, и тогда необходимо будет возвратиться на чтение условия. Не один раз лучше представить решение задачи и по возможности упростить. Для начала лучше решить задачу ручным способом, без использования компьютера, используя по максимуму входных и выходных данных, рассматривая решение с самых простых случаев. Чаще всего существует не один подход к решению одной и той же задачи. И сложность заключается в том, чтобы увидеть эти варианты подходов и выбрать наиболее подходящий для этой задачи здесь и сейчас и реализовать его. Всвязи с множеством вариантов решений можно наткнуться на неверно выбранный путь, и чаще

всего увидеть это можно на последующих этапах, что приводит на возврат к этому этапу.

Разработка алгоритма решений – это поиск эффективного алгоритма решения и пути его реализации, на данном этапе получаем ответ на вопрос «как это делать?». Творческий подход к решению задач на этом этапе играет главную роль, как и знание теоретических основ информатики, и опыт решения олимпиадных задач. У каждого исполнителя свой уникальный алгоритм решения для определенной задачи, который он сам придумал или переработал и усовершенствовал. Но существуют общепринятые рекомендации. Как правило, этап начинается с определения ведущей идеи выбранного алгоритма. Для упрощения задачи можно переключиться с ограничивающих обстоятельств (время, компьютерная техника и ее свойства) и постараться найти один из возможно-верных путей решений. В таком случае каждая из возникших идей требует предварительной проработки ручным способом и доведении ее до конкретного результата. Обязательным перед отправкой задания на проверку жюри является использование из условия задачи примеров входных и выходных файлов или тестирование программы с помощью своих тестовых данных. При возникновении трудностей на этапе разработки приемлемой идеи решения задачи, необходимо попробовать разложить ее на более простые задачи. Данный порядок следует последовательно применить к каждой из задач по отдельности.

Более того, вместе с понимание хода решения задач можно разработать структуру алгоритма, и наполнять ее соответствующим содержанием. Только при наличии предварительно проработанной идеи можно приступить к описанию структуры алгоритма, выделяя при этом основные компоненты, выстраивая взаимосвязь между ними. В этом случае могут пригодиться домашние заготовки алгоритмов, используя их проще получить окончательное решение задачи. Во время подготовки к

олимпиадеучастникунеобходимо хорошо освоить определенный набор типовых алгоритмов и умело в дальнейшем их применить. Так же решения задач могут основываться на математических идеях, которые необходимо придумать, в этом случае элементами могут быть типичные процедуры и функции (сортировка, перебор, динамическое программирование и др.).

Исключить нельзя и тот случай, когда было быстро придумано решение, но реализовывалось которое долго и упорно, тогда как существует более эффективный и простой путь для решения данной задачи, но чтобы увидеть его, необходимо чуть больше подумать.

Что будет представлять собой каждая алгоритмическая единица и весь алгоритм в целом? Какой будет структура? Какие массивы данных будут использоваться? Как будет осуществлена передача параметров? Данные вопросы следует тщательней всего проработать после того, как будет выделен алгоритм и определены подходы к его детализации.

На протяжении всего этапа желательно все фрагменты решения записывать на бумагу, при написании мыслей происходит фиксация важных фактов и завершается их упорядочивание, в дальнейшем это помогает избежать ошибок в процессе реализации решения задач. Начальные значения переменных, процедуры это те факты, которые необходимо записать. При прочтении таких записей будет проще обнаружить, все ли данные учтены, правильно ли определены переменные. В записях быстрее получится найти нужное место при отладке программы и позволит избежать нелепых ошибок, допустим, таких как забывание присвоения начальному значению переменной.

Данный этап для самого участника должен заканчиваться корректным получением алгоритма и оценкой его эффективности. Эти компоненты решения олимпиадных задач являются очень важными, так как именно они в большей степени определяют правильность полученного решения. Довольно

часто, кажется, что алгоритм интуитивно работает правильно, но на самом деле при более внимательном рассмотрении могут возникнуть проблемы во внутренней структуре алгоритма, довольно сложного алгоритма, что не так просто сразу понять в голове. Для доказательства корректности алгоритма можно разработать набор тестов, учитывающих особенности алгоритма, и провести тестирование программы на них. Как показывает практика, лучше сразу подумать над доказательством корректности и либо уметь вычислять такие оценки для используемых ими стандартных алгоритмов или знать их. Если при тестировании алгоритма оценки не удовлетворяют нашим ограничениям, то необходимо оптимизировать наше решение полностью или отдельными подзадачами. В такой ситуации, не лишнем будет остановиться и подумать, нужно ли дальше тратит время на усовершенствование программы или стоит найти более простое решение, которое позволит сохранить больше времени. Ведь нередко бывает так, что участник долго разрабатывает точное решение для поставленной задачи, но безуспешно, только потому, что его в принципе не существует.

Большинство участников совмещают этап разработки алгоритма и этап реализации программы, поскольку бывает сложно удержаться от соблазна быстрее начать писать программу, и это оправдывает себя в том случае, когда участник уверен, что задача знакомая или простая.

После того как определились с алгоритмом решения задачи переходим к написанию программы. Для начала надо определить, какое из структурного программирования мы будем использовать: «сверху вниз» или «снизу вверх». Главной задачей структурного программирования является то, что в нем имеется возможность разбиения программы на подпрограммы.

Программирование «сверху-вниз», или нисходящее программирование – это методика разработки программ, при которой разработка начинается с определения целей решения проблемы, после чего идет последовательная

детализация, заканчивающаяся детальной программой. [4]Другими словами при проектировании нисходящем задача рассматривается с целью определения возможности ее разбиения на подзадачи. Далее каждая из выведенных подзадач рассматривается и разбивается еще на подзадачи. Процесс будет завершен только в том случае, когда подзадачу невозможно или нецелесообразно разбивать на подзадачи дальше.

В таком случае программа конструируется сверху вниз, от главной программы к подпрограммам нижнего уровня, то есть иерархически,используя при этом только простые циклы и условные разветвления.

Программирование «снизу-вверх», или восходящее программирование – это методика разработки программ, начинающаяся с разработки подпрограмм (процедур, функций), в то время когда проработка общей схемы не закончилась. Является противоположной методике программирования «сверху-вниз». [4]

В восходящем программировании проектирование начинается с подзадач и приходит к задаче, но в таком случае такой подход может привести к переделкам, увеличению времени и в итоге к нежелательным результатам.

Какая методика лучше, однозначно сказать трудно, так как подход к задачам разный и возможно использование первого и второго подхода, а иногда приходится применять их вместе, комбинируя между собой.

При целостном понимании алгоритма решения задачи используется программирование «сверху вниз», в таком случае пишется основная часть программы, а интерфейс процедур и функции только согласовывается.В случае если разбиение на подпрограммы логично связано со структурой

разработанного алгоритма, то этот подход приводит к хорошим результатам, так же упрощая доработку решения до окончательного вида и ее отладку.

Второй подход используется в том случае, когда при дефиците времени все-таки требуется писать программу, но пока непонятно как именно. Тогда сначала напишем то, что у нас есть, а это входные данные, стандартные функции и процедуры.И пока выполняем рутинные операции, работа над алгоритмом не прекращается, возникают новые идеи, которые, возможно материализовать в виде программных компонентов.

В любом из подходов необходимо писать программы так, чтобы легко потом в них разобраться, одним из вспомогательных способов являются комментарии, в которых описаны переменные, функции и др. Еще важным является описание формата входных и выходных данных, приведенных в условии задачи. При разработке программы следует учитывать, что она должна читать входные данные из входного файла, решать задачу и вывести результат в выходной файл. Неправильное написание имен входных и выходных файлов в программе считается ошибкой.Только внимательное чтение условия задачи и тщательный разбор с приведенными там примерами входных и выходных данных приведет к необходимому результату.

Немаловажным пунктом является сохранение редактируемых файлов во время работы. Ведь не редким бывает и сбои в самой компьютерной станции, и перебои в электричестве, которые могут привести к потере данных. Чтобы предотвратить потерю в начале тура необходимо в используемой среде программирования включить режим автосохранения, и, кроме этого, самому своевременно сохранять свои файлы.

Отладка и апробация программы является значительным и обязательным этапом при разработке любого программного обеспечения. Нужна только та программа, которая работает и работает без ошибок.

Тестирование программного средства – это процесс выполнения программ на некотором наборе данных, для которого заранее известен результат применения или известны правила поведения этих программ. Указанный набор данных называется тестовым или просто тестом. Тестирование программ является одной из составных частей более общего понятия – «отладка программ». Под отладкой понимается процесс, позволяющий получить программу, функционирующую с требующимися характеристиками в заданной области изменения входных данных [5].

Следовательно, отладку можно представить в виде неоднократного повторения процессов: апробации, в процессе которой может быть определена ошибка, ее поиск и исправление текста программы с целью устранения обнаруженной ошибки.

Большинство участников олимпиад по разным причинам считают, что если программа компилируется и правильно работает на нескольких наборах входных данных, то решение получено правильно и высылают его на проверку. Но это не значит, что полученная программа будет удовлетворять ограничениям, заданным в условиях задачи и соответствовать заданной размерности входных данных. Причиной того могут быть и незначительные ошибки в именах переменных, задании размеров массивов и даже то, что разработанный алгоритм окажется принципиально неправильным.

Первым делом необходимо убедиться, что программа компилируется и компилируется правильно. Овладение навыками использования встроенных отладчиков в среду программирования вот то, что поможет при компилировании, тем более что это является неотъемлемой частью подготовки к олимпиаде по информатике. Непосредственно к процессу тестирования программы стоит переходить, только добившись того, что программа компилируется правильно. Если времени до окончания тура

осталось совсем немного, и вы понимаете, что тестирование уже невозможно, то вам следует отправить решенную задачу на проверку жюри.

В первую очередь тестировать программу нужно с примера в условии задачи. Даже если, кажется, что он прост, но даже он с первого раза не проходит. Используя именно тест из условия задачи можно найти существенный процент ошибок, во многих случаях сразу понятно, где ошибка и как ее исправить. Если же сразу не получается, то необходимо изучить, что же происходит в программе на этих входных данных. И при поиске ошибок нужно решить, каким именно процессом займемся - проверим логику работы программы или будем заниматься отладкой готовой программы. Переписывать небольшую часть стоит в том случае, когда у вас большие сомнения. При использовании отладки нельзя забывать, что ошибки могут находиться в разных частях программы, что делает этот процесс довольно долгим. Поэтому необходимо соблюдать определенный баланс и разумно использовать эти подходы.

Во время соревнований, когда участник находится в стрессовом состоянии и присутствует фактор времени, всегда следует помнить, что никто не застрахован от ошибок в элементарных алгоритмах, даже если и использовались при подготовке к олимпиаде неоднократно.

Особого внимания заслуживает одна закономерность: участники, исправляя одни ошибки, часто допускают новые. И чтобы этот процесс не зациклился, необходимо скрупулёзно подходить к исправлению ошибок и помнить об этом. Желательно, и даже необходимо, перед внесением изменений в структуру программы сделать резервную копию. При обнаружении в дальнейшем грубой ошибкиу вас всегда под рукой будет копия, тем самым вы ускорите процесс реализации программы. При успешном тестировании на примере из условия задачи, необходимо придумать свои тесты. Данный процесс не редко бывает творческим и

сложным, чем разработка алгоритма решения задачи. Тесты следует разрабатывать содержательными и ориентированными на проверку логики работы алгоритма, чтобы максимально добиться ее правильности.

2.3 Апробацияучебного пособия по подготовке к олимпиадным задачам по информатике.

Учебное пособие по подготовке к олимпиадным задачам по информатике прошел апробацию школьниками 7-го класса средней общеобразовательной школы.

Целью апробации являлось выявление возможности и целесообразности использования данного учебного практикума в учебном процессе.

Школьникам было предложено решить задачи из пособия и оценить доступность изложения материала и возможность самостоятельного освоения.

Решить все задачи из 25 школьников смогли 18. Остальные 7 решили задания на 70%, это дает возможность сделать вывод, что пособие будет востребовано.

Заключение

Целью исследования была разработка учебного пособия и методических рекомендации по подготовке учащихся 7 – х классов к олимпиадам по программированию.

При написании дипломной работы было разработано учебное пособие по подготовки учащихся 7 – х классов к олимпиадам по программированию.Для этого были разработаны задачи разной степени сложности и разной тематики, которые можно решать как в учебное время, так и во внеурочное. Пособие структурировано, целостно, понятно для использования.

Решены все задачи, поставленные нами для выполнения цели исследования.