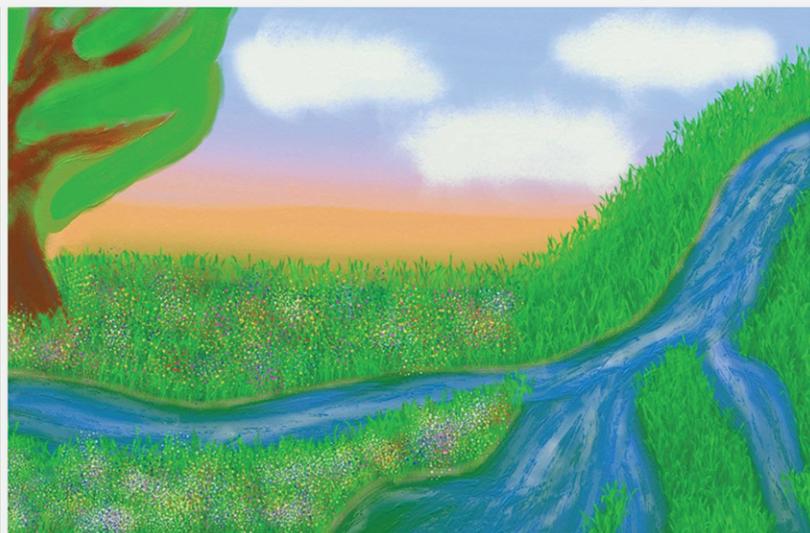


ИНФОРМАТИКА В ШКОЛЕ

№ 8'2014

ISSN 2221-1993

www.infojournal.ru



ИНФОРМАТИКА В ШКОЛЕ

№ 8(101) октябрь 2014

Редакционный совет

Баракина Т. В.
Бешенков С. А.
Босова Л. Л.
Дергачева Л. М.
Жданов С. А.
Заславская О. Ю.
Захарова Т. Б.
Кириченко И. Б.
Кравцова А. Ю.
Кузнецов А. А.
Левченко И. В.
Окулов С. М.
Рыбаков Д. С.
Семакин И. Г.
Христочевский С. А.

Редакция

Дергачева Л. М.
главный редактор
Кириченко И. Б.
*заместитель
главного редактора*
Губкин В. А.
Коптева С. А.
Лукичева И. А.
Меркулова Н. И.
Федотов Д. В.
Шарапкина Л. М.

Адрес редакции:
119121, г. Москва,
ул. Погодинская, д. 8, оф. 222
Телефон/факс: (495) 708-36-15
E-mail: readinfo@infojournal.ru
URL: http://www.infojournal.ru

*Подписные индексы
в каталоге «Роспечать»:*
для индивидуальных подписчиков — 81407
для предприятий и организаций — 81408

Подписано в печать 13.10.2014.
Формат 60×90¹/₈. Усл. печ. л. 8,0.
Тираж 2000 экз. Заказ № 1152.

Отпечатано в типографии
ООО «ГЕО-Полиграф»
141290, Московская область,
г. Красноармейск, ул. Свердлова, д. 1.

Свидетельство о регистрации
средства массовой информации
ПИ № 77-12068 от 11 марта 2002 г.

Все права защищены. Никакая часть журнала
не может быть воспроизведена в любой форме
или любыми средствами, электронными или
механическими, включая фотографирование,
сканирование, магнитную запись, размещение
в Интернете или иные средства копирования
или сохранения информации, без письменного
разрешения издательства.

© «Образование и Информатика», 2014

Содержание

От редакции 2

КОНКУРС ИНФО-2013

Панченко Н. П. Чтобы научиться бегать, сначала нужно научиться ходить... 3

ИТОГОВАЯ АТТЕСТАЦИЯ ПО ИНФОРМАТИКЕ

Каткова А. Л. Некоторые особенности подготовки учащихся к государственной итоговой аттестации по информатике и ИКТ 26

УРОКИ ИНФОРМАТИКИ

Доктор М. В. Урок на тему «Программирование перевода чисел из одной системы счисления в другую»..... 30

Батакова Е. Л. Изучение линейного алгоритма с помощью интерактивных средств обучения..... 35

ЗАДАЧИ

Сулейманов Р. Р. Компьютерное моделирование математических задач 39

Окулов С. М., Пестов О. А. Двоичное дерево поиска..... 46

Патрушев Г. О. Алгоритм организации кругового турнира..... 52

ЗАНИМАТЕЛЬНАЯ ИНФОРМАТИКА

Очков В. Ф., Богомолова Е. П. Римские — арабские 56

Кильдишов В. Д. Создание часов с использованием диаграммы MS Excel... 59



Уважаемые коллеги!

Коллектив редакции журнала «Информатика в школе», члены редакционного совета поздравляют вас с профессиональным праздником — Днем учителя!

Желаем вам здоровья, благополучия, новых успешных свершений. Пусть вам сопутствует удача во всех начинаниях, пусть ваш благородный труд приносит радость и удовлетворение! Мудрости вам и терпения!

Все победы и достижения учеников — это всегда заслуга учителя, результат его каждодневных усилий. Только профессионализм, самоотверженный труд учителя, наставника, воспитателя позволяют детям в полной мере раскрыть свои таланты, реализовать все свои возможности. Учитель своими руками творит характер, индивидуальность, личность ребенка и в конечном счете будущее своего ученика. Еще Сократ более двух тысяч лет назад сказал: «В каждом человеке солнце, только дайте ему светить». Даря ребенку частичку своего тепла, своей любви, учитель получает ученика-творца, ученика-единомышленника, ученика-соратника. В педагогике успех может быть только общим — талантливого учителя и талантливых учеников. Мы учим детей — дети учат нас. Каждый урок — это новая возможность стать лучше, умнее, интереснее, добрее.

От всей души желаем вам, дорогие коллеги, увлеченных, любознательных, благодарных учеников!

Мира, счастья и добра вам и вашим семьям!

*Редакция журнала
«Информатика в школе»*

Роняет старый клен раскрашенные листья,
И шорохом дождя наполнен школьный двор.
Давайте от проблем мы головы очистим,
Неспешно заведем о главном разговор...

Своих учителей встречаю по дороге
И открываю вновь любимой школы дверь.
Мои учителя ко мне бывают строги,
Придирчивы порой — и раньше, и теперь.

Они мне задают нелегкие вопросы,
Но не подскажут, как и где найти ответ.
И я его ищу, порой глотая слезы.
Я понимаю, что — увы! — ответа нет.

В запасе есть у них нелегкие задачи,
Где уравнилья слов должны сложиться в жизнь.
И я решаю их, надеясь на удачу,
И снова говорю: «Давай, не ошибись...»

Но я найду ответ в любимых детских лицах
И в свете ясных глаз, чьи взгляды так легки.

Вы для меня всегда — открытая страница,
Мои учителя — мои ученики...

Н. Ю. Иванова



Н. П. Панченко,
дипломант конкурса ИНФО-2013 в номинации «Мой урок информатики»,
Дворец детского (юношеского) творчества, г. Дзержинск, Нижегородская область

ЧТОБЫ НАУЧИТЬСЯ БЕГАТЬ, СНАЧАЛА НУЖНО НАУЧИТЬСЯ ХОДИТЬ*

Аннотация

В статье представлена методическая разработка лабораторной работы, содержащая комплект тренировочных тестов, задания для обязательного и самостоятельного решения задач целочисленной арифметики, а также задания для самоконтроля.

Ключевые слова: информатика, язык программирования, алгоритм, программа, целочисленная арифметика, олимпиада, ЕГЭ, оценка результата, пословица, занимательные.

Контактная информация

Панченко Надежда Петровна, педагог дополнительного образования Дворца детского (юношеского) творчества, г. Дзержинск, Нижегородская область; *адрес:* 606023, Нижегородская область, г. Дзержинск, пл. Ленина, д. 1; *телефон:* (831-3) 26-19-27; *e-mail:* panchenko@yandex.ru

N. P. Panchenko,
Palace of Children (Youth) Creativity, Dzerzhinsk, Nizhny Novgorod Region

TO LEARN HOW TO RUN, FIRST YOU NEED TO LEARN TO WALK

Abstract

The article presents the laboratory work, comprising a set of training tests, assignments for the compulsory and independent problem solving of integer arithmetic, as well as jobs for the selfcontrol.

Keywords: informatics, programming language, algorithm, program, integer arithmetic, olympiad, Unified state examination, assessment of result, saying, entertaining.

Урок должен быть пронизан творчеством и вместе с тем оставаться уроком.

С. И. Гессен

Новые стандарты образования предъявляют высокие требования не только к учащимся, но и, прежде всего, к педагогу. Основные условия образовательного процесса — сжатые сроки, огромный объем информации, жесткие требования к знаниям и умениям обучающихся. Добиться результата в таких условиях невозможно, если применять только традиционные методы и средства обучения. Необходимо развивать творческую активность учащихся, использовать задания, требующие выполнения школьниками исследовательской работы. Ребята должны вовлекаться в самостоятельный творческий поиск, выполняя постепенно усложняющиеся задания.

Цель любого занятия — постепенное, шаг за шагом, исполнение работы в изучаемой программной среде. Лабораторная работа — это одна из форм, в которой сочетаются теоретический материал, необходимый для выполнения работы, и практическая часть для отработки знаний, умений и навыков. Выполнение лабораторных работ требует предварительной подготовки учащихся, включающей освоение теории изучаемой проблемы, состава и принципа функционирования лабораторной модели, вычислительной техники и программного обеспечения, методики выполнения работы. В лабораторных работах материал учащимся нужно подавать в адаптированной, удобной для понимания форме. Такая практика позволяет существенно ускорить процесс освоения обучающимися программной среды, достаточно быстро формирует у них представление о технологии работы и ее возможностях для решения задач. В процессе выполнения практической работы учащиеся самостоятельно работают на компьютере. Они учатся выделять главное и конспектировать необходимые сведения, что помогает им заложить фундамент для дальнейшей самостоятельной работы. Скорость выполнения заданий зависит от индивидуальных качеств обучающихся и уровня их подготовленности.

Применение средств ИКТ, таких как презентация и имеющаяся в ней возможность посредством гиперссылки вызывать программные среды (например DEV-C++, PascalABC и др.), позволяет выполнять на занятии максимальное количество разнообразных по содержанию и сложности заданий.

В любом языке программирования тема «Целочисленная арифметика» актуальна. Умение решать задачи такого типа пригодится учащимся в дальнейшем: задачи на целочисленную арифметику часто являются подзадачами более сложных задач, а также предлагаются на олимпиадах любого уровня.

* Материалы к статье можно скачать на сайте ИНФО:
<http://www.infojournal.ru/infojournal/school/archive/8-2014>

Тема занятия: Целочисленная арифметика в C/C++.

Цель занятия: формирование навыков практического применения приемов и способов решения задач на целочисленную арифметику.

Задачи занятия:

обучающие:

- закрепление знаний, умений и навыков работы с целочисленной арифметикой в C/C++;
- формирование умения работать с тестовыми материалами;

развивающие:

- развитие алгоритмического и логического мышления;
- развитие творческой активности, познавательного интереса;

воспитательные:

- побуждение к самостоятельной учебной деятельности — от постановки проблемы до ее практического решения;
- профессиональное самоопределение.

Тип занятия: комбинированное занятие.

Вид занятия: лабораторная работа.

Возраст обучающихся: обучающиеся соответствующего профиля учреждений дополнительного образования или обучающиеся профильных групп десятых-одиннадцатых классов.

Оборудование занятия:

- маркерная доска, маркер;
- мультимедийный проектор;
- компьютеры, объединенные в локальную сеть;
- среда программирования — в качестве среды программирования используется бесплатно распространяемая оболочка DEV-C++ с открытым исходным кодом, включающая компилятор GCC и позволяющая посредством гиперссылок демонстрировать работу программ прямо из презентации.

Методическое обеспечение занятия:

- учебно-методический комплекс:
 - текст лабораторной работы (*Приложение 1*);
 - ответы (*Приложение 2*);
 - рейтинговая таблица для обучающихся (*Приложение 1, табл. 1*);
 - критерии оценки программного продукта (*Приложение 1, табл. 2*);
 - сводная рейтинговая таблица с критериями оценки (*Приложение 4**);
 - презентация «Целочисленная арифметика в C/C++» (*Приложение 5*);
 - типичные ошибки (*Приложение 3*);
 - пословицы (*Приложение 1*);
 - тексты программ на языках программирования C/C++, Pascal, QBasic;
- комплекс упражнений для профилактики переутомления;
- план занятия.

Методы и приемы обучения: словесные, наглядные, проблемно-поисковые, практические, поощрение, создание ситуации успеха.

Планируемая деятельность обучающихся:

- проверка теоретических знаний — устный опрос, выполнение заданий самоконтроля;
- заполнение рейтинговой таблицы (*Приложение 1, табл. 1*);
- практическая работа на ПК — применение теоретических знаний, умений и навыков в процессе выполнения заданий различного уровня сложности.

Планируемые действия педагога:

- комментирует электронную презентацию;
- демонстрирует из презентации работу программ — предлагает один из возможных вариантов решения задач;
- просматривает у каждого обучающегося выполнение заданий самоконтроля, обращает внимание учащихся на типичные ошибки (*Приложение 3*), заполняет сводную рейтинговую таблицу (*Приложение 4*);
- консультирует, ведет диалог;
- обсуждает с обучающимися каждый этап самоконтроля, итоги выполненных заданий, отмечает и поощряет тех, кто успешно справился с ними;
- оценивает вместе с обучающимися задания, выполненные на ПК (*Приложение 1, табл. 2*).

Рекомендации по проведению занятия:

- презентация может демонстрироваться на экране для групповой работы или может быть передана по локальной сети на каждое рабочее место (ПК) для работы в индивидуальном режиме;
- во время занятия необходимо соблюдать режим работы в компьютерном классе;
- каждый обучающийся получает для практической работы свой экземпляр текста лабораторной работы (*Приложение 1*), выполненная работа остается у обучающегося;
- в зависимости от уровня подготовки обучающихся педагог может использовать проверочные задания не только в виде самоконтроля, но и используя формы педагогической проверки знаний;
- обучающиеся отмечают выполненные задания в рейтинговой таблице (*Приложение 1, табл. 1*), сравнивают свои ответы с правильными ответами на слайдах;
- педагог может использовать взаимопроверку: обучающиеся обмениваются работами и проверяют работы друг друга;
- в затруднительных случаях задания по самоконтролю выполняются обучающимися на ПК.

Формы контроля и оценки результатов:

- тестовые задания;
- устный опрос;
- самопроверка, взаимопроверка;
- выполнение заданий на ПК.

Ожидаемые образовательные результаты:

предметные:

- сформированность алгоритмического, логического и аналитического мышления; умений обучающихся: писать и отлаживать программы линейной структуры, составлять контрольные примеры;

* Приложения 4–6 можно скачать на сайте ИНФО, приложения 1–3 даны в конце статьи и на сайте ИНФО: <http://www.infojournal.ru/infojournal/school/archive/8-2014>

- закрепление знаний, умений и навыков работы с целочисленной арифметикой в C/C++;
- сформированность умений обучающихся работать с тестовыми материалами;
- владение приемами и способами решения задач на целочисленную арифметику;

личностные:

- готовность и способность к самостоятельному поиску методов решения практических задач, принятию самостоятельных решений, мотивация к познанию нового;
- пробуждение интереса к практической деятельности в целях дальнейшего профессионального самоопределения;
- повышение творческой активности;

метапредметные:

- умение планировать, анализировать и оценивать свою деятельность;
- способность участвовать в обсуждениях, формулировать собственное мнение;
- умение сотрудничать с педагогом и другими обучающимися, определять способы взаимодействия;
- умение осуществлять самоконтроль и определять свой уровень усвоения учебного материала;

интегрированные:

- умение обучающихся управлять собственной деятельностью от цели до планируемого результата;
- умение выбрать способы действий в соответствии со своим индивидуальным стилем деятельности;
- умение ориентироваться в различных ситуациях, владеть стратегией и приемами деятельности.

Рекомендуемая литература для учащихся.

1. *Андреева Е. В.* Программирование — это так просто, программирование — это так сложно. Современный учебник программирования. М.: МЦНМО, 2009.
2. *Динман М. И.* С++. Освой на примерах. СПб.: БХВ-Петербург, 2006.
3. *Крупняк А. Б.* Самоучитель С++. СПб.: Питер, 2005.
4. *Культин Н. Б.* C/C++ в задачах и примерах. СПб.: БХВ-Петербург, 2006.

Рекомендуемая литература для педагога.

1. *Величева Е. П.* Занимательные задачи по программированию // Информатика и образование. 2011. № 6.
2. *Дергачева Л. М.* Решение типовых экзаменационных задач по информатике: учеб. пособие. М.: БИНОМ. Лаборатория знаний, 2013.
3. *Зеленяк О. П.* Практикум программирования на Turbo Pascal: задачи, алгоритмы и решения: 2-е изд., испр. и доп. СПб.: ДиаСофеЮП, 2002.
4. *Меньшиков Ф. В.* Олимпиадные задачи по программированию (+CD). СПб.: Питер, 2007.
5. *Панченко Н. П.* Гимн отрезку прямой // Информатика и образование. 2004. № 6.

Рекомендуемые интернет-источники.

1. *Гущин Д. Д.* Задания В2. Оператор присваивания и ветвления. <http://inf.reshuege.ru/test?theme=176>
2. *Калинина Н. А., Костюкова Н. И.* Основы программирования на языке С. <http://www.intuit.ru/department/pl/c/1/1.html>
3. *Ларина Э. С.* Решение олимпиадных задач. <http://www.intuit.ru/department/school/olympr/0/>
4. Основы программирования для начинающих. <http://iguania.ru/>
5. *Панченко Н. П.* Гимн отрезку прямой. <http://festival.1september.ru/articles/213040/>
6. *Поляков К. Ю.* Задания В2. Оператор присваивания и ветвления. <http://kpolyakov.narod.ru/school/ege.htm/>
7. *Поляков К. Ю.* Программирование на языке Си. <http://kpolyakov.narod.ru/school/c.htm>
8. Уроки Си. <http://programmersclub.ru/01/>

План занятия.

1. Организационный этап.
2. Подготовительный этап.
 - 2.1. Повторение.
 - 2.2. Самоконтроль.
3. Разбор задач.
 - 3.1. Вопросы для обсуждения.
4. Контрольный этап.
 - 4.1. Задачи для самостоятельного решения.
 - 4.2. Олимпиадные задачи.
5. Итоговый этап.
 - 5.1. Подведение итогов занятия.
 - 5.2. Материалы для самоподготовки и самообразования.

Содержание и ход занятия

№ слайда	Задание	Задачи	Действия педагога	Рекомендации к выполнению заданий	Планируемая деятельность обучающихся	Практический результат
1. Организационный этап						
1		Организация внимания, настрой на предстоящую практическую работу	Приветствует обучающихся, отмечает отсутствующих, знакомит с темой занятия		Приветствуют педагога, занимают рабочие места, готовят ручки, тетради	Активизация внимания, настрой на предстоящую работу
2		Знакомство с содержанием и целью занятия, правилами заполнения рейтинговой таблицы и оценкой выполняемых работ	Напоминает условные обозначения, раздает работы с заданиями	Данные условные обозначения используются во всех презентациях автора	Записывают тему занятия, подписывают свой экземпляр лабораторной работы	Готовность обучающихся к активной учебно-познавательной деятельности

Продолжение таблицы

№ слайда	Задание	Задачи	Действия педагога	Рекомендации к выполнению заданий	Планируемая деятельность обучающихся	Практический результат
2. Подготовительный этап						
3	2.1. Повторение	Повторение тем, изученных на предыдущих занятиях	Обращает внимание обучающихся на связь данной темы с темами предыдущих занятий, психологически готовит обучающихся к проведению самоконтроля	Оценивание заданий 1.1–1.3 производится на усмотрение педагога	Слушают объяснение педагога, вспоминают ранее изученный материал	Готовность выполнять задания по темам, изученным ранее
4	Задание 1.1	Проверка умения правильно записывать термины, знания применения операторов	Производит устный опрос по ранее изученным темам	Необходимо обратить внимание на то, что в школах изучаются разные иностранные языки, и это может привести к ошибкам в написании терминов	Выполняют задания, сверяют свои ответы с ответами на слайде	Правильное написание служебных слов в тексте программы
5	Задание 1.2	Проверка умений: <ul style="list-style-type: none"> • понимать алгоритм; • переводить алгоритм в программный код 	К доске по желанию приглашает двух учеников, за проявление инициативы выполнить задание у доски педагог добавляет баллы	В случае затруднения педагог задает наводящие вопросы	Выполняют задание, сверяют с ответом на слайде, отвечают на вопросы педагога, ведут с ним диалог	Понимание заданных алгоритмов и их связи с текстом программы
6	Задание 1.3	Повторение стандартных математических функций	Напоминает об особенностях использования модуля целого и вещественных чисел в C/C++, дает задание на закрепление знаний	В случаях затруднений предлагает обучающимся проверить работу функций на ПК	Выполняют задание и сверяют ответы с ответами на слайде	Закрепление знаний
7	2.2. Самоконтроль	Определение степени усвоения программного материала обучающимися	Рассказывает о содержании данного этапа занятия, предлагает выполнить самоконтроль в форме тестирования. Напоминает о том, что необходимо заполнять личную рейтинговую таблицу, что дает возможность получить дополнительные баллы	Задания самоконтроля идут по нарастанию уровня сложности. Проверку можно осуществлять как по каждому заданию, так и после выполнения всех заданий самоконтроля	Выполняют задания по самоконтролю и сравнивают ответы с ответами на слайдах	Развитие аналитического мышления. Умение применять полученные знания на практике
8	Задание 2.1	Обучение умению находить и исправлять ошибки	Предлагает выполнить задание, после его выполнения обращает внимание учащихся на типичные ошибки, предлагает исправить ошибки самостоятельно, в случае затруднения показывает правильные ответы на слайде		Выполняют задание. Сравнивают свои решения с ответом на слайде, исправляют ошибки, ведут диалог с педагогом, подсчитывают количество правильно выполненных заданий, заполняют рейтинговую таблицу	Отработка навыков и умений находить ошибки, их устранение, повышение мотивации к обучению, уверенности в собственных силах

Продолжение таблицы

№ слайда	Задание	Задачи	Действия педагога	Рекомендации к выполнению заданий	Планируемая деятельность обучающихся	Практический результат
9	Задание 2.2	Проверка умения выполнять операции целочисленного деления в языке программирования C/C++	Предлагает выполнить задание. Отвечает на вопросы. Индивидуально консультирует. Показывает на слайде правильные ответы	Особенности целочисленного деления в C/C++. Подготовка к решению более сложных задач		Понимание особенностей целочисленной арифметики в C/C++
10	Задание 2.3	Проверка умения выполнять целочисленное деление в десятичной системе счисления в языке программирования C/C++		Через практическую работу подготовить к понятию позиционности десятичной СС, умению выделять цифры сотен, десятков, единиц		Понимание структуры числа в десятичной системе счисления
11	Задание 2.4	Применение знаний на практике	Предлагает решить задачи из школьного курса математики и сравнить с возможными вариантами решения	Затруднение может вызвать задача 5, где нужно учесть особенности целочисленного деления в C/C++. Можно воспользоваться функциями округления или представить 0.4 как $4/10$ или $2/5$, тогда $r = m \cdot 2/5 = m \cdot 5/2$		Умение решать практические задачи
12	Задание 2.5	Решение задач ЕГЭ уровня В2	Предлагает решить задачи более сложного уровня. После выполнения заданий открывает правильные ответы, ведет диалог с обучающимися			Знакомство с задачами ЕГЭ. Приобретение опыта решения задач ЕГЭ. Повышение самооценки, уверенности в своих силах
13		Создание ситуации успеха	Поощряет обучающихся и сообщает им, что предложенные задания взяты из заданий ЕГЭ, уровень В2			
14		Акцентирование внимания на типичных ошибках при выполнении вычислений	Рассказывает о возможных ошибках в данном типе задания (<i>Приложение 3</i>)		Записывают информацию в тетради	Предупреждение возможных ошибок при выполнении практических заданий
3. Основной этап						
15	3. Разбор задач	Разбор задач для обязательного решения	Рассказывает о работе на этом этапе		Слушают педагога	Готовность к выполнению практических заданий на ПК
16		Обсуждение типичных ошибок при написании программ	Акцентирует внимание обучающихся на возможных ошибках при написании программ (<i>Приложение 3</i>)		Записывают полученную информацию в тетради	Предупреждение возможных ошибок при написании программ

Продолжение таблицы

№ слайда	Задание	Задачи	Действия педагога	Рекомендации к выполнению заданий	Планируемая деятельность обучающихся	Практический результат
17	3.1. Вопросы для обсуждения	Подготовка к решению более сложных задач	Предлагает обсудить тему о системах счисления	При решении задач данной темы обучающиеся должны знать и четко представлять структуру числа в его позиционности	Отвечают устно, записывают ответы	Закрепление знаний о структуре числа и его позиционности. Готовность обучающихся к решению сложных задач
18	Задача 1	Разбор алгоритма и текста программы, подготовка к формам записи алгоритма: словесному и в виде программного кода	Напоминает о форме записи алгоритма — словесной. Указывает на соответствие алгоритма и текста программы. Подчеркивает, что текст программы — тоже одна из форм представления алгоритма	Важно сообщить: одно из заданий ЕГЭ, а именно С2, звучит так, и процитировать: <i>«...опишите на русском языке или на одном из языков программирования алгоритм, который позволяет...»</i>	Разбирают и анализируют алгоритм в словесном и программном видах, выполняют его на ПК. Проверяют правильность работы программы на контрольных примерах	Знание правил записи алгоритма в словесном и программном кодах
19	Задача 2					
20	Задача 3	Обсуждение алгоритма решения задачи, подведение к пониманию того, что задачи 1 и 3 решаются по одному алгоритму	Обсуждает решение задачи, предлагает ученикам написать и отладить программу. Напоминает о том, что увеличить количество решенных задач и ускорить процесс написания программы можно за счет использования: <ul style="list-style-type: none"> • заготовки программы; • приемов работы с текстом: вставка, копирование и т. д.; • скорости набора текста программы; • знания и правильного написания операторов 	Педагог после обсуждения и решения задачи может вызвать желающих к доске или предложить обучающимся выполнить задание самостоятельно	Обсуждают с педагогом решение задачи, пишут текст программы, показывают педагогу, выполняют работу на ПК	Самостоятельное написание программ с использованием рациональных приемов работы, понимание необходимости приведения контрольных примеров
4. Контрольный этап						
21–24	4.1. Задачи для самостоятельного решения	Самостоятельное решение задач	Предлагает задачи различного уровня сложности для самостоятельного решения	Программы учащиеся должны написать сами. Тексты программ задач 4.5–4.7 рекомендуется сохранить, так как они понадобятся при изучении темы «Условный оператор»	Обучающиеся пишут тексты программ, показывают их педагогу, выполняют их на ПК. Каждый обучающийся работает в индивидуальном режиме	Умение находить свои способы решения задач различного уровня сложности. Самостоятельное написание программы, проверка работы на заданных контрольных примерах и составление своих
25–27	4.2. Олимпиадные задачи	Подготовка к решению олимпиадных задач	Данный тип задач педагог предлагает решить тем, кто успешно справился с предыдущими заданиями	На занятии предлагаются задачи разного уровня сложности. Олимпиадные задачи относятся к повышенному уровню сложности	Каждый обучающийся выбирает задачи соответственно уровню своих подготовленности, возможностей и способностей	Готовность к самостоятельному решению олимпиадных задач и задач повышенной сложности

Окончание таблицы

№ слайда	Задание	Задачи	Действия педагога	Рекомендации к выполнению заданий	Планируемая деятельность обучающихся	Практический результат
5. Итоговый этап						
28	5.1. Подведение итогов занятия	Оценка практической работы обучающихся	Заполняет сводную рейтинговую таблицу по материалам выполнения практических работ обучающихся, добавляет поощрительные баллы, подводит итог занятия. Демонстрирует результаты рейтинга на слайде (<i>Приложение 4</i>)	Необходимо оценить работы обучающихся и сделать выводы: <ul style="list-style-type: none"> • освоили ли они тему, • что вызвало особое затруднение; • что можно использовать на следующем занятии для исправления ошибок 	В конце занятия обучающиеся представляют педагогу данные о выполненных работах	Осознание и адекватная оценка обучающимися собственных возможностей, умение анализировать и применять знания на практике
29–31	5.2. Материалы для самоподготовки и самообразования	Снять эмоциональное напряжение после выполненной работы, мотивировать обучающихся к повышению уровня своих знаний через самообразование	Педагог приводит пословицы и предлагает подумать над их смыслом. Рекомендует материалы для самоподготовки и пополнения знаний	Можно привести притчу о рыбе и удочке (<i>Приложение 1</i>)	Слушают педагога, обсуждают, выражают свое понимание смысла пословиц, притчи	Обучающиеся принимают к сведению информацию. Мотивация обучающихся к дальнейшему пополнению своих знаний

Приложения

Приложение 1

Лабораторная работа «Целочисленная арифметика в C/C++»

Фамилия, имя _____ группа _____ дата _____

Таблица 1

Рейтинг для учащихся

№ п/п	Задание	Баллы	Задачи для самостоятельного решения				
			Баллы	Повышенной сложности	Баллы	Олимпиадные	Баллы
1	1.1		4.1	4.8		5.1	
2	1.2		4.2	4.9		5.2	
3	1.3		4.3	4.10		5.3	
4	2.1		4.4			5.4	
5	2.2		4.5			5.5	
6	2.3		4.6				
7	2.4		4.7				
8	2.5						
9	Итого:						
						Всего баллов:	

Примечание: 1 балл за правильно выполненный пункт задания 2.1–2.5 (максимум — 50 баллов);
 1 балл за решенную задачу (максимум — 7 баллов);
 1–2 балла за решенную задачу (максимум — 6 баллов);
 1–3 баллов за решенную задачу (максимум — 15 баллов).

Критерии оценки выполненной работы для заданий 4 и 5

№ п/п	Критерии	Да	Нет	Частично
1	Программа выполняется			
2	Получен достоверный результат (соответствие поставленной задаче)			
3	Правильно оформлен текст программы (наличие комментариев, отступов)			
4	Использованы материалы новой темы			
5	Грамотно оформлен результат (есть подсказки и рационально использован экран)			
6	Использованы ранее написанные программы (заготовки)			
7	Использованы рациональные приемы работы с текстом программы (копирование, вставка, удаление и т. д.)			
8	Оригинальность решения и творческие находки			
9	Применены операторы, не изученные ранее			
10	Компактность программы			
11	Грамматические ошибки			

Повторение

Задание 1.1. Устный опрос.

- Какие библиотеки используются для организации ввода/вывода? _____
- С какой библиотекой работают операторы ввода/вывода `printf/scanf`? _____
- С какой библиотекой работают операторы ввода/вывода `cin/cout`? _____
- Особенности использования двух типов операторов ввода/вывода _____
- Как помечают комментарии в C/C++? _____
- Для чего нужна функция `getch()`? _____

Задание 1.2. Перевести команды на язык программирования.

Задание 1	Написать фрагмент программы	Задание 2	Написать фрагмент программы
Ввести A и B		Ввести A и B	
Увеличить A в 10 раз		Найти сумму A и B	
Уменьшить B в 3 раза		Уменьшить сумму в 10 раз	
Напечатать A и B		Напечатать результат S	

Вопросы для обсуждения:

- Какой тип данных должен быть у переменной A ? _____
- Какой тип данных должен быть у переменной B ? _____
- Какой тип данных должен быть у переменной S ? _____
- Как это скажется на выводе данных? _____

Задание 1.3. Стандартные функции.

- Какую библиотеку нужно подключить для работы стандартных математических функций? _____
- Вычислить значения функций:

Функция	Пример	Ответ
<code>abs(x)</code>	<code>abs(-10)</code> <code>abs(-10.4)</code>	
<code>fabs(x)</code>	<code>fabs(-10.23)</code> <code>fabs(-10)</code>	
<code>sqrt(x)</code>	<code>sqrt(16)</code> <code>sqrt(0.04)</code>	
<code>pow(x,y)</code>	<code>pow(2,3)</code>	

Самоконтроль

Задание 2.1. Найти и объяснить ошибки, вычислить.

№ п/п	Пример	Ответ	Комментарии	№ п/п	Пример	Ответ	Комментарии
1	$9 / 2 = 4.5$			6	$\text{fab}c(-4) = 4$		
2	$5 \bmod 2 = 1$			7	$\text{abs}(-3,15) = 3.15$		
3	$2^4 = 16$			8	$2ax$		
4	$2,4 + 3,1 = 6,5$			9	$(a: 3 + 4): 2 \text{ а к}$		
5	$\text{ABC}(-3) = 3$			10	$\text{sqrt}(-16) = 4$		

Задание 2.2. Вычислить значения выражений.

№ п/п	Пример	Ответ	№ п/п	Пример	Ответ
1	$10/2 + 10/3 =$		6	$3 / 10 + 3 \% 5 =$	
2	$10/2. + 10/3. =$		7	$1 + 25 / 5 \% 2 =$	
3	$10/2. + 10/3 =$		8	$(1 + 19) \% 5 =$	
4	$10/2 + 10/3. =$		9	$19 - 4 \% 5 =$	
5	$3 / 10 + 25 \% 5 =$		10	$3 / (10 + 3) \% 5 =$	

Задание 2.3. Выполнить целочисленное деление.

№ п/п	Найти целую часть числа	Ответ	Найти остаток от деления	Ответ	№ п/п	Найти целую часть числа	Ответ	Найти остаток от деления	Ответ
1	$21 / 3$		$21 \% 3$		6	$751 / 10$		$751 \% 10$	
2	$3 / 10$		$3 \% 10$		7	$751 / 100$		$751 \% 100$	
3	$17 / 5$		$17 \% 5$		8	$-751 / 10$		$-751 \% 10$	
4	$17 / 10$		$17 \% 10$		9	$751 / -100$		$751 \% -100$	
5	$5 / 10$		$5 \% 10$		10	$-751 / -10$		$-751 \% -10$	

Задание 2.4. Написать инструкцию присваивания и решить задачи.

№ п/п	Задача	Ответ
1	Дано расстояние в см. Найти число полных метров. $s = 345, m = ?$	
2	Дана масса в кг. Найти число полных центнеров. $m = 851, z = ?$	
3	Дана масса в кг. Найти число полных тонн. $m = 3996, t = ?$	
4	Дана масса в кг. Найти число полных пудов (1 пуд = 16 кг). $m = 161, p = ?$	
5	Дана масса в кг. Найти число полных фунтов (1 фунт = 400 г). $m = 405, f = ?$	
6	Дано расстояние в метрах. Найти число полных км. $s = 45672, km = ?$	
7	Дан прямоугольник 400×500 см. Сколько квадратов со стороной 50 см можно из него получить?	
8	Дано трехзначное число a , найти в нем цифру сотен	
9	В трехзначном числе найти цифру единиц	
10	В трехзначном числе сложить цифры десятков и единиц и вычесть цифру сотен	

Задание 2.5. Чему равны значения переменных после выполнения последовательных действий?

№ п/п	Задание	Ответ	№ п/п	Задание	Ответ
1	$a = 15 / (16 \% 7);$ $b = 34 \% a * 5 - 29 \% 5 * 2;$ $a = a \% b + a / b;$		6	$a = 1819;$ $b = (a / 100) * 10 + 9;$ $a = (10 * b - a) \% 100;$	
2	$a = 4 * 7 / 3 \% 2;$ $b = 4 * 5 / (3 * a \% 2 + 1);$ $a = a * b;$ $b = b * b;$		7	$a = 4321;$ $b = (a \% 100) + 22;$ $a = (a * 10) / 100 - b * 10;$ $a = a + b;$	
3	$m = 67;$ $m = m + 13;$ $n = m / 4 - m / 2;$ $c = m - n;$		8	$a = 2025;$ $b = a \% 1000 - a / 100;$ $a = (b + a) \% 1000;$	
4	$a = 6 * 12 + 3;$ $b = a / 10 + 5;$ $a = b \% 10 + 1;$ $c = a * a + b * b - a / 2 * b;$		9	$a = 1234;$ $b = (a / 1000) * 101;$ $a = (b \% 10) + a;$	
5	$a = 2468;$ $b = (a \% 1000) * 10;$ $a = a / 1000 + b;$		10	$a = 7974;$ $b = (a / 100) * 10 - 26;$ $a = (a - b) \% 10 + 7864;$	

Разбор задач для обязательного решения*Вопросы для обсуждения:*

- Какие системы счисления (СС) вы знаете? _____
- Что такое позиционная СС? _____
- Как записать число $a = 345_{10}$ через степень 10? _____
- Как из числа $a = 345_{10}$ получить новое число $b = 543_{10}$? _____

Задача 1. Задано трехзначное число. Найти сумму цифр числа.Например: $a = 341, s = 3 + 4 + 1 = 8.$

Составить свои примеры.

Алгоритм	Программа	Контрольные примеры
1. Подключить библиотеки	<pre>#include <stdio.h> #include <conio.h> main() { //установка русского языка setlocale(LC_CTYPE, "Russian");</pre>	
2. Описать переменные	<pre>int a, z1, z2, z3, s;</pre>	
3. Ввести число a	<pre>printf("задай a="); scanf("%i", &a);</pre>	
4. Найти цифру единиц: $z1=a\%10$	<pre>z1=a%10;</pre>	
5. Найти цифру десятков: $z2=a/10\%10$	<pre>z2=a/10%10;</pre>	
6. Найти цифру сотен: $z3=a/100;$	<pre>z3=a/100;</pre>	
7. Найти сумму цифр: $s=z1+z2+z3$	<pre>s=z1+z2+z3;</pre>	
8. Вывести ответ s	<pre>printf("s = %d\n", s);</pre>	
9. Конец программы	<pre>getch(); }</pre>	

Задача 2. Сколько купюр k потребуется для выдачи суммы s ? В наличии имеются только купюры достоинством n и купюры по одному рублю.

Алгоритм	Программа	Контрольные примеры
1. Задать сумму s . 2. Задать достоинство купюры n . 3. Вычислить количество купюр достоинством n : $k1=s/n$ 4. Вычислить кол-во рублевых купюр: $k2=s\%n$	<pre>//установка русского языка #include <stdio.h> #include <conio.h> main() {</pre>	$s = 54; n = 10;$ Ответ: $k = 9.$ $s = 203; n = 50;$ Ответ: $k = 7.$

Окончание таблицы

Алгоритм	Программа	Контрольные примеры
5. Вычислить общее кол-во купюр: $k=k1+k2$ $k=s/n+s*n$ 6. Вывести ответ	<pre>//установка русского языка setlocale(LC_STYPE, "Russian"); int s, n, k, k1, k2; printf("задай сумму="); scanf("%i", &s); printf("задай дост. купюры="); scanf("%i", &n); k1=s/n; k2=s*n; // или k=s/n+s*n; printf("кол-во куп. = %d\n", k); getch(); }</pre>	

Задача 3. Задано время в секундах $t = 12015$. Перевести в часы, минуты, секунды (h, m, c).

Вопросы для обсуждения:

- К какой СС относится время? _____
- Как записать время в этой системе счисления, т. е. перевести время в секунды? Например, задано время: 3 часа 20 мин 15 с _____
- Алгоритм какой задачи может быть использован в данном случае? _____
- Как получить время в часах? _____
- Как получить время в минутах? _____
- Как получить время в секундах? _____
- За счет чего можно сократить время написания программы? _____

Задание. Написать программу и выполнить на компьютере.

Входные данные	Выходные данные
12015	3 20 15
23723	6 35 23
85915	23 51 55

Задачи для самостоятельного решения

Задача 4.1. В трехзначном числе найти произведение его цифр.

Например: $a = 234, p = 2 * 3 * 4 = 24$.

Входные данные	Выходные данные
234	24
111	1
512	10

Задача 4.2. В трехзначном числе поменять местами цифры сотен и единиц.

Например: было 742, стало 247.

Входные данные	Выходные данные
234	432
115	511
518	815

Задача 4.3. Если в трехзначном числе есть одинаковые цифры: напечатать — цифру «0», в противном случае — любое число.

Входные данные	Выходные данные
234	2
115	0
511	0

Задача 4.4. Известно время начала и конца соревнования в часах, минутах, секундах. Сколько времени проходило соревнование в часах, минутах, секундах?

Входные данные	Выходные данные
1 10 10 2 30 30	1 20 20
2 55 45 5 15 10	2 19 25
3 45 15 6 10 15	2 25 00

Задача 4.5. Задать день рождения (дд). Остаток от деления на 9 укажет вам, какой цветок ваш: 0 — роза, 1 — подсолнух, 2 — пион, 3 — вика, 4 — василек, 5 — лилия, 6 — орхидея, 7 — красный мак, 8 — маргаритка.

Программу сохранить и записать ее имя в тетрадь.

Задача 4.6. Задать год рождения (гггг). Из года рождения вычесть 3, остаток от деления на 12 укажет вам, кто вы по знаку восточного гороскопа: 0 — кабан, 1 — крыса, 2 — буйвол, 3 — тигр, 4 — кот (заяц), 5 — дракон, 6 — змея, 7 — лошадь, 8 — коза, 9 — обезьяна, 10 — петух, 11 — собака.

Программу сохранить и записать ее имя в тетрадь.

Задача 4.7. Последняя цифра вашего года рождения определит стихию и цвет по японскому календарю: 0 — белый металл, 1 — матовый или серый металл, 2 — черная вода, 3 — серая вода, 4 — синее дерево, 5 — голубое дерево, 6 — красный огонь, 7 — розовый огонь, 8 — желтая земля, 9 — светло-зеленая земля.

Задачи для самостоятельного решения повышенной сложности

Задача 4.8. Ввести с клавиатуры два целых числа: m , $n > 0$. Если m делится на n или n делится на m , то вывести 1, в противном случае — любое другое число.

Задача 4.9. Ввести с клавиатуры два целых числа: m , $n > 0$. Если $n \leq m$, то вывести 1, в противном случае — любое другое число.

Задача 4.10. Если в четырехзначном числе, введенном с клавиатуры, есть одинаковые цифры, то вывести 1, в противном случае — любое другое число.

Олимпиадные задачи

Задача 5.1. Определить номера подъезда и этажа по номеру квартиры девятиэтажного дома, считая, что на каждом этаже расположены четыре квартиры, а нумерация начинается с первого подъезда.

Входные данные	Выходные данные
1	1
10	1
16	8
23	5

Входные данные	Выходные данные
2012	5
2013	7
2014	8
2015	9

Входные данные	Выходные данные
1988	8
2000	0
1997	7

Входные данные	Выходные данные
15 3	1
15 4	13
15 15	1
3 15	1

Входные данные	Выходные данные
2 4	1
2 2	1
5 3	3

Входные данные	Выходные данные
1223	0
2132	0
1234	12

Входные данные	Выходные данные	
	подъезд	этаж
№ кв.		
1	1	1
4	1	1
36	1	9
37	2	1
40	2	2
72	2	9
135	4	7

Задача 5.2. Решить эту же задачу, задавая этажность и количество квартир на этаже. Придумать контрольные примеры.

Задача 5.3. Бутылка воды стоит 45 коп. Пустые бутылки сдаются по 20 коп., на полученные деньги опять покупается вода. Какое наибольшее количество бутылок воды можно купить, имея некоторую сумму денег — s копеек?

Входные данные	Выходные данные
45	1
90	2
95	3
150	5
300	11

Задача 5.4. Часовая стрелка образует угол u с лучом, проходящим через центр и точку, соответствующую 12 часам на циферблате. $0 \leq u \leq 360$. Ввести с клавиатуры значение u и определить значение угла для минутной стрелки, а также количество часов и полных минут.

Входные данные	Выходные данные
90	3 0 0
160	5 20 120
180	6 0 0
270	9 0 0

Задача 5.5. Определить угол, который образуют часовая и минутная стрелка.

Входные данные	Выходные данные
3 ч	90
1 ч 30 мин	135
6 ч	180

Материалы для самообразования и самоподготовки

Как вы понимаете смысл следующих пословиц и притчи?

- Китайские:
 - Учителя открывают дверь. Входишь ты сам.
 - Можно привести верблюда к воде, но нельзя заставить его пить.
 - Многие жалуются на свою внешность, и никто — на мозги.
- Русские:
 - Без труда не вынешь и рыбку из пруда.
 - Дорогу осилит идущий.
- Народные:
 - Чтобы научиться бегать, нужно сначала научиться ходить.
- Притча о рыбе и удочке:
 - «Научи человека ловить рыбу, дай ему удочку, и он, будучи научен, будет ловить...»

Литература.

1. *Андреева Е. В.* Программирование — это так просто, программирование — это так сложно. Современный учебник программирования. М.: МЦНМО, 2009.

2. *Динман М. И.* С++. Освой на примерах. СПб.: БХВ-Петербург, 2006.

3. *Крупняк А. Б.* Самоучитель С++. СПб.: Питер, 2005.

4. *Культин Н. Б.* С/С++ в задачах и примерах. СПб.: БХВ-Петербург, 2006.

Интернет-источники.

1. *Гущин Д. Д.* Задания В2. Оператор присваивания и ветвления. <http://inf.reshuege.ru/test?theme=176>

2. *Калинина Н. А., Костюкова Н. И.* Основы программирования на языке С. <http://www.intuit.ru/departament/pl/c/1/1.html>

3. *Ларина Э. С.* Решение олимпиадных задач. <http://www.intuit.ru/departament/school/olymp/0/>

4. Основы программирования для начинающих. <http://iguania.ru/>

5. *Поляков К. Ю.* Задания В2. Оператор присваивания и ветвления. <http://kpolyakov.narod.ru/school/ege.htm/>

6. *Поляков К. Ю.* Программирование на языке Си. <http://kpolyakov.narod.ru/school/c.htm>

7. Уроки Си. <http://programmersclub.ru/01/>

Приложение 2

Ответы

Повторение

Задание 1.1. Устный опрос.

- Какие библиотеки используются для организации ввода/вывода? (*stdio.h, iostream. h.*)
- С какой библиотекой работают операторы ввода/вывода printf/scanf? (*stdio. h.*)
- С какой библиотекой работают операторы ввода/вывода cin/cout? (*iostream. h.*)
- Особенности использования двух типов операторов ввода/вывода. (*Не рекомендуется осуществлять ввод одним типом оператора (scanf), а вывод — другим (cout).*)
- Как помечают комментарии в С/С++? (*// или /* */*)
- Для чего нужна функция getch()? (*Ждать нажатия на любую клавишу.*)

Задание 1.2. Перевести команды на язык программирования.

Задание 1	Написать фрагмент программы	Задание 2	Написать фрагмент программы
Ввести A и B	<pre>printf("задай a="); scanf("%i", &a); printf("задай b="); scanf("%f", &b);</pre>	Ввести A и B	<pre>printf("задай a="); scanf("%i", &a); printf("задай b="); scanf("%f", &b);</pre>
Увеличить A в 10 раз	<pre>a=a*10;</pre>	Найти сумму A и B	<pre>s=a+b;</pre>
Уменьшить B в 3 раза	<pre>b=b/3;</pre>	Уменьшить сумму в 10 раз	<pre>s=s/10;</pre>
Напечатать A и B	<pre>printf("a = %d, b = %3.2f", a, b);</pre>	Напечатать результат S	<pre>printf("s = %3.1f", s);</pre>

Вопросы для обсуждения:

- Какой тип данных должен быть у переменной A ? (*Любой, так как целый тип данных вводит в вещественный.*)
- Какой тип данных должен быть у переменной B ? (*float.*)
- Какой тип данных должен быть у переменной S ? (*float.*)
- Как это скажется на выводе данных? (*Целые — %d, вещественные — %3.2f.*)

Задание 1.3. Стандартные функции.

- Какую библиотеку нужно подключить для работы стандартных математических функций? (*math. h.*)
- Вычислить значения функций:

Функция	Пример	Ответ	Функция	Пример	Ответ
abs(x)	abs(-10)	10	sqrt(x)	sqrt(16)	4
	abs(-10.4)	10		sqrt(0.04)	0.2
fabs(x)	fabs(-10.23)	10.23	pow(x,y)	pow(2,3)	8
	fabs(-10)	10			

Самоконтроль**Задание 2.1.** Найти и объяснить ошибки, вычислить.

№ п/п	Пример	Ответ	Комментарии
1	$9 / 2 = 4.5$	$9 / 2 = 4$	Ошибка в вычислении
2	$5 \bmod 2 = 1$	$5 \% 2 = 1$	Неправильно обозначена операция нахождения остатка
3	$2^4 = 16$	$\text{pow}(2, 4) = 16$	Неправильно обозначена операция возведения в степень
4	$2,4 + 3,1 = 6,5$	$2.4 + 3.1 = 5.5$	В языках программирования символ «.» отделяет целую часть числа от дробной
5	$\text{ABC}(-3) = 3$	$\text{abs}(-3) = 3$	Ошибка синтаксиса
6	$\text{fab}c(-4) = 4$	$\text{fabs}(-4) = 4$	Ошибка синтаксиса
7	$\text{abs}(-3,15) = 3.15$	$\text{abs}(-3.15) = 3$	Функция abs в C/C++ модуль вещественного числа находит как целое число
8	$2ax$	$2 * a * x$	Не поставлен знак умножения
9	$(a: 3 + 4): 2 a k$	$(a / 3+4) / (2 * a * k)$	Использованы знаки арифметических операций из математики
10	$\text{sqrt}(-16) = 4$	$\text{sqrt}(16) = 4$	Из отрицательного числа нельзя извлечь корень

Задание 2.2. Вычислить значения выражений.

№ п/п	Пример	Ответ	№ п/п	Пример	Ответ
1	$10/2 + 10/3 =$	8	6	$3 / 10 + 3 \% 5 =$	3
2	$10/2. + 10/3. =$	8.33	7	$1 + 25 / 5 \% 2 =$	2
3	$10/2. + 10/3 =$	8	8	$(1 + 19) \% 5 =$	0
4	$10/2 + 10/3. =$	8.33	9	$19 - 4 \% 5 =$	15
5	$3 / 10 + 25 \% 5 =$	0	10	$3 / (10 + 3)\% 5 =$	0

Задание 2.3. Выполнить целочисленное деление.

№ п/п	Найти целую часть числа	Ответ	Найти остаток от деления	Ответ	№ п/п	Найти целую часть числа	Ответ	Найти остаток от деления	Ответ
1	21 / 3	7	21 % 3	0	6	751 / 10	75	751 % 10	1
2	3 / 10	0	3 % 10	3	7	751 / 100	7	751 % 100	51
3	17 / 5	3	17 % 5	2	8	-751 / 10	-75	-751 % 10	-1
4	17 / 10	1	17 % 10	7	9	751 / -100	-7	751 % -100	-51
5	5 / 10	0	5 % 10	5	10	-751 / -10	75	-751 % -10	1

Задание 2.4. Написать инструкцию присваивания и решить задачи.

№ п/п	Задача	Ответ
1	Дано расстояние в см. Найти число полных метров. $s = 345, m = ?$	$m = s / 100 = 3$
2	Дана масса в кг. Найти число полных центнеров. $m = 851, z = ?$	$z = m / 100 = 8$
3	Дана масса в кг. Найти число полных тонн. $m = 3996, t = ?$	$t = m / 1000 = 3$
4	Дана масса в кг. Найти число полных пудов (1 пуд = 16 кг). $m = 161, p = ?$	$p = m / 16 = 10$
5	Дана масса в кг. Найти число полных фунтов (1 фунт = 400 г). $m = 405, f = ?$	$f = m * 10 / 4 = 1012$ $0.4 = 4 / 10$
6	Дано расстояние в метрах. Найти число полных км. $s = 45672, km = ?$	$km = s / 1000 = 45$
7	Дан прямоугольник 400×500 см. Сколько квадратов со стороной 50 см можно из него получить?	$k = a * b / pow(c, 2) = 80$
8	Дано трехзначное число a , найти в нем цифру сотен	$z3 = a / 100$
9	В трехзначном числе найти цифру единиц	$z1 = a \% 10$
10	В трехзначном числе сложить цифры десятков и единиц и вычесть цифру сотен	$z1 = a \% 10; z2 = a / 10 \% 10; z3 = a / 100;$ $b = z1 + z2 - z3$

Задание 2.5. Чему равны значения переменных после выполнения последовательных действий?

№ п/п	Задание	Ответ	№ п/п	Задание	Ответ
1	$a = 15 / (16 \% 7);$ $b = 34 \% a * 5 - 29 \% 5 * 2;$ $a = a \% b + a / b;$	$a = 7$ $b = 22$	6	$a = 1819;$ $b = (a / 100) * 10 + 9;$ $a = (10 * b - a) \% 100;$	$a = 71$ $b = 189$
2	$a = 4 * 7 / 3 \% 2;$ $b = 4 * 5 / (3 * a \% 2 + 1);$ $a = a * b;$ $b = b * b;$	$a = 10$ $b = 100$	7	$a = 4321;$ $b = (a \% 100) + 22;$ $a = (a * 10) / 100 - b * 10;$ $a = a + b;$	$a = 45$ $b = 43$
3	$m = 67;$ $m = m + 13;$ $n = m / 4 - m / 2;$ $c = m - n;$	$m = 80$ $n = -20$ $c = 100$	8	$a = 2025;$ $b = a \% 1000 - a / 100;$ $a = (b + a) \% 1000;$	$a = 30$ $b = 5$
4	$a = 6 * 12 + 3;$ $b = a / 10 + 5;$ $a = b \% 10 + 1;$ $c = a * a + b * b - a / 2 * b;$	$a = 3$ $b = 12$ $c = 141$	9	$a = 1234;$ $b = (a / 1000) * 101;$ $a = (b \% 10) + a;$	$a = 1235$ $b = 101$
5	$a = 2468;$ $b = (a \% 1000) * 10;$ $a = a / 1000 + b;$	$a = 4682$ $b = 4680$	10	$a = 7974;$ $b = (a / 100) * 10 - 26;$ $a = (a - b) \% 10 + 7864;$	$a = 7864$ $b = 764$

Разбор задач для обязательного решения

Вопросы для обсуждения:

- Какие системы счисления (СС) вы знаете? (*Позиционные СС — двоичная, десятичная и т. д., римская СС.*)
- Что такое позиционная СС? (*Каждой цифре числа соответствует свой разряд — позиция.*)

- Как записать число $a = 345_{10}$ через степень 10? ($a = 3 * 100 + 4 * 10 + 5 = 3 * 10^2 + 4 * 10^1 + 5 * 10^0$.)
- Как из числа $a = 345_{10}$ получить новое число $b = 543_{10}$? ($b = 5 * 10^2 + 4 * 10^1 + 3 * 10^0$.)

Задача 1. Листинг программы.

С/С++	Паскаль	Бейсик
<pre>//c3_1_z1 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { //установка русского языка setlocale(LC_STYPE, "Russian"); int a, z1, z2, z3, s; printf("задай a="); scanf("%i", &a); z1=a%10; z2=a/10%10; z3=a/100; s=z1+z2+z3; printf("s = %d\n", s); getch(); }</pre>	<pre>{p3_1_z1} var a, z1, z2, z3, s: integer; begin write('Задай a -->'); readln(a); z1:=a mod 10; z2:=a div 10 mod 10; z3:=a div 100; s:=z1+z2+z3; writeln('Ответ - ', s:3); end.</pre>	<pre>'b3_1_z1 DIM s, z1, z2, z3, s AS INTEGER CLS INPUT "Задай a -->"; a z1=a mod 10 z2=a\10 mod 10 z3=a\100 s=z1+z2+z3 PRINT "Ответ - "; s END</pre>

Задача 2. Листинг программы.

С/С++	Паскаль	Бейсик
<pre>//c3_1_z2 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { //установка русского языка setlocale(LC_STYPE, "Russian"); int s, n, k, k1, k2; printf("задай сумму="); scanf("%i",&s); printf("задай дост. купюры="); scanf("%i", &n); //k1=s/n; k2=s%n; k = k1 + k2; k=s/n+s%n; printf("кол-во куп. = %d\n", k); getch(); }</pre>	<pre>{p3_1_z2} var s, n, k, k1, k2: integer; begin write('Задай сумму = '); readln(s); write('Задай дост. купюры = '); readln(n); {k1:=s div n; k2:=s mod n; k:=k1 + k2;} k:=s div n+s mod n; writeln('кол-во куп. = ',k:3); end.</pre>	<pre>'b3_1_z2 DIM s, n, k, k1, k2 AS INTEGER CLS INPUT "Задай сумму = "; S INPUT "Задай дост. купюры ="; n 'k1=s\n 'k2=s mod n 'k=k1+k2 k=s\n+s mod n; PRINT "кол-во куп. = ";k END</pre>

Задача 3.

Вопросы для обсуждения:

- К какой СС относится время? (*Шестидесятеричной*).
- Как записать время в этой системе счисления, т. е. перевести время в секунды? Например, задано время: 3 часа 20 мин 15 с ($t_{сек} = 3 * 60^2 + 20 * 60^1 + 15 = 12015$.)
- Алгоритм какой задачи может быть использован в данном случае? (*Алгоритм задачи 1 — нахождения цифр трехзначного числа* $t_{сек} = 3 * 60^2 + 20 * 60^1 + 15 = 12015$.)

сотни
десятки
единицы
- Как получить время в часах? ($h = t / 3600$.)
- Как получить время в минутах? ($m = t / 60 \% 60$.)
- Как получить время в секундах? ($c = t \% 60$.)
- За счет чего можно сократить время написания программы? (*Наличие заготовки программы, скорость набора, знание правильности написания операторов, использование приемов работы с текстом: копирование, вставка и т. д.*)

Листинг программы.

С/С++	Паскаль	Бейсик
<pre>//c3_1_z3 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() {</pre>	<pre>{p3_3_z3} var h, m, s: integer; t: longint; begin write('Задай t = '); readln(t);</pre>	<pre>'b3_1_z3 DIM h, m, s AS INTEGER DIM t AS LONG CLS INPUT "Задай t="; t s=t MOD 60 m=t\60 MOD 60</pre>

Окончание таблицы

C/C++	Паскаль	Бейсик
<pre>setlocale(LC_CTYPE, "Russian"); int h, m, s; long t; printf("задай t="); scanf("%i", &t); s=t%60; m=t/60%60; h=t/3600; printf("h = %d, m = %d, s = %d", h, m, s); getch(); }</pre>	<pre>s:=t mod 60; m:=t div 60 mod 60; h:=t div 3600; writeln('h= ', h:3, ' m=', m:3, ' s = ', s:3); end.</pre>	<pre>h=t\3600 PRINT "h= "; h, " m ="; m, " s = "; s END</pre>

Задачи для самостоятельного решения

Задача 4.1. Алгоритм. Использовать алгоритм задачи 1. $p = z_1 * z_2 * z_3$.

C/C++	Паскаль	Бейсик
<pre>//c4_1 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_CTYPE, "Russian"); int a, p, z1, z2, z3; printf("задай a="); scanf("%i", &a); z1=a%10; z2=a/10%10; z3=a/100; p=z1*z2*z3; printf("p = %d", p); getch(); }</pre>	<pre>{p4_1} var a, z1, z2, z3, p: integer; begin write('Задай a = '); readln(a); z1:=a mod 10; z2:=a div 10 mod 10; z3:=a div 100; p:=z1*z2*z3; writeln(' p= ', p:3); end.</pre>	<pre>'b4_1 DIM a, z1, z2, z3, p AS INTEGER CLS INPUT "Задай a "; a z1=a MOD 10 z2=a\10 MOD 10 z3=a\100 p=z1*z2*z3 PRINT " p= "; p END</pre>

Задача 4.2. Алгоритм. Использовать алгоритм задачи 1 и позиционность десятичной системы счисления, тогда $r = z_1 * 100 + z_2 * 10 + z_3$.

C/C++	Паскаль	Бейсик
<pre>//c4_2 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_CTYPE, "Russian"); int a, b, z1, z2, z3; printf("задай a="); scanf("%i",&a); z1=a%10; z2=a/10%10; z3=a/100; b=z1*100+z2*10+z3; printf("b = %d", b); getch(); }</pre>	<pre>{p4_2} var a, z1, z2, z3, r: integer; begin write('Задай a = '); readln(a); z1:=a mod 10; z2:=a div 10 mod 10; z3:=a div 100; r:=z1*100+z2*10+z3; writeln(' r = ',r:3); end.</pre>	<pre>'b4_2 DIM a, z1, z2, z3, r AS INTEGER CLS INPUT "Задай a="; a z1=a MOD 10 z2=a\10 MOD 10 z3=a\100 r=z1*100+z2*10+z3 PRINT " r = "; r END</pre>

Задача 4.3. Алгоритм. Разность двух одинаковых цифр даст 0. Найти цифры числа, тогда $r = (z_1 - z_2) * (z_1 - z_3) * (z_2 - z_3)$ даст ответ.

C/C++	Паскаль	Бейсик
<pre>//c4_3 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() {</pre>	<pre>{p4_3} var a, z1, z2, z3, r: integer; begin write('Задай a = '); readln(a); z1:=a mod 10; z2:=a div 10 mod 10;</pre>	<pre>'b4_3 DIM a, z1, z2, z3, r AS INTEGER CLS INPUT "Задай a="; a z1=a MOD 10 z2=a\10 MOD 10 z3=a\100</pre>

Окончание таблицы

C/C++	Паскаль	Бейсик
<pre>setlocale(LC_CTYPE, "Russian"); int a, r, z1, z2, z3; printf("задай a="); scanf("%i", &a); z1=a%10; z2=a/10%10; z3=a/100; r=(z1-z2)*(z1-z3)*(z2-z3); printf("r = %d", r); getch(); }</pre>	<pre>z3:=a div 100; r:=(z1-z2)*(z1-z3)*(z2-z3); writeln(' r = ',r:3); end.</pre>	<pre>r=(z1-z2)*(z1-z3)*(z2-z3) PRINT " r = ";r END</pre>

Задача 4.4. Алгоритм. Перевести время начала и конца соревнования в секунды, найти разность, использовать алгоритм решения задачи 3.

C/C++	Паскаль	Бейсик
<pre>//c4_4 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_CTYPE, "Russian"); int h, m, s, h1, m1, s1, h2, m2, s2; long t, t1, t2; printf("задай t начала соревнований h1, m1, s1 -> "); scanf("%i %i %i", &h1, &m1, &s1); printf("задай t конца соревнований h2, m2, s2 -> "); scanf("%i %i %i", &h2, &m2, &s2); t1=h1*3600+m1*60+s1; t2=h2*3600+m2*60+s2; t=abs(t1-t2); //или //t=abs(h1-h2)*3600+abs(m1- m2)*60+abs(s1-s2); s=t%60; m=t/60%60; h=t/3600; printf("\n h=%d, m=%d, s=%d", h, m, s); getch(); }</pre>	<pre>{p4_4} var h, m, s, h1, m1, s1, h2, m2, s2: integer; t, t1, t2: longint; begin write('Задай t начала соревно- ваний ='); readln(h1, m1, s1); write('Задай t конца соревнова- ний ='); readln(h2, m2, s2); t1:=h1*3600+m1*60+s1; t2:=h2*3600+m2*60+s2; t:=abs(t1-t2); {или t:=abs(h1-h2)*3600+abs(m1-m2)*60 +abs(s1-s2);} s:=t mod 60; m:=t div 60 mod 60; h:=t div 3600; writeln('h=', h:3, ' m=', m:3, 's =', s); end.</pre>	<pre>'b4_4 DIM h, m, s, h1, m1, s1, h2, m2, s2 AS INTEGER DIM t, t1, t2 AS LONG CLS INPUT " Задай t начала соревнова- ний"; h1, m1, s1 INPUT " Задай t конца соревнова- ний"; h2, m2, s2 t1=h1*3600+m1*60+s1 t2=h2*3600+m2*60+s2 t=ABS(t1-t2) 'или t:=abs(h1-h2)*3600+abs(m1- m2)*60+abs(s1-s2) s=t MOD 60 m=t\60 MOD 60 h=t\3600 PRINT "h="; h, " m=", m, "s =", s END</pre>

Задача 4.5. Алгоритм. $r = dr \% 9$.

C/C++	Паскаль	Бейсик
<pre>//c4_5 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_CTYPE, "Russian"); int dr, r; printf("задай день рождения (дд) --> "); scanf("%i",&dr); r=dr%9; printf("\n Ваш цветок - %d", r); getch(); }</pre>	<pre>{p4_5} var dr, r: integer; begin write('Задай день рождения (дд) --> ='); readln(dr); r:=dr mod 9; writeln('Ваш цветок - ', r:3); end.</pre>	<pre>'b4_5 DIM dr, r AS INTEGER CLS INPUT "Задай день рождения (дд)"; dr r=dr MOD 9 PRINT "Ваш цветок - "; r END</pre>

Задача 4.6. Алгоритм. $r = (gr - 3) \% 12$.

C/C++	Паскаль	Бейсик
<pre>//c4_6 #include <conio.h> #include <iostream.h></pre>	<pre>{p4_6} var gr, r: integer; begin</pre>	<pre>'b4_6 DIM gr, r AS INTEGER CLS</pre>

Окончание таблицы

C/C++	Паскаль	Бейсик
<pre>#include <stdlib.h> #include <math.h> main() { setlocale(LC_STYPE, "Russian"); int gr,r; printf("задай год рождения (гггг) --> "); scanf("%i", &gr); r=(gr-3)%12; printf("\n Ваш знак зодиака - %d", r); getch(); }</pre>	<pre>write('Задай год рождения (гггг) --> ='); readln(gr); r:=(gr-3) mod 12; writeln('Ваш знак зодиака - ', r:3); end.</pre>	<pre>INPUT "Задай год рождения (гггг) "; gr r=(gr-3) MOD 12 PRINT "Ваш знак зодиака - "; r END</pre>

Задача 4.7. Алгоритм. $r = gr \% 10$.

C/C++	Паскаль	Бейсик
<pre>//c4_7 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_STYPE, "Russian"); int gr, r; printf("задай год рождения (гггг) --> "); scanf("%i",&gr) ; r=gr%10; printf("\n Ваша стихия - %d", r); getch(); }</pre>	<pre>{p4_7} var gr, r: integer; begin write('Задай год рождения (гггг) --> ='); readln(gr); r:=gr mod 10; writeln('Ваша стихия - ', r:3); end.</pre>	<pre>'b4_7 DIM gr, r AS INTEGER INPUT "Задай год рождения (гггг)->"; gr r=gr MOD 10 PRINT "Ваша стихия - "; r END</pre>

Задачи для самостоятельного решения повышенной сложности

Задача 4.8. Алгоритм. Число делится нацело, когда остаток от деления равен 0, тогда $r = (n \% m) * (m \% n) + 1$.

C/C++	Паскаль	Бейсик
<pre>//c4_8 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_STYPE, "Russian"); int n, m, r; printf("задай m, n --> "); scanf("%i %i",&m,&n); r=(n%m)*(m%n)+1; printf("\n Ответ - %d", r); getch(); }</pre>	<pre>{p4_8} var n, m, r: integer; begin write('Задай m, n --> ='); readln(m, n); r:=(n mod m)*(m mod n)+1; writeln('Ответ - ', r:3); end.</pre>	<pre>'b4_8 DIM n, m, r AS INTEGER CLS INPUT "Задай m, n"; m, n r=(n MOD m)*(m MOD n)+1 PRINT "Ответ - "; r END</pre>

Задача 4.9. Алгоритм. При делении меньшего числа на большее $n / m = 0$, при равенстве чисел $n - m = 0$, тогда $r = (n / m) * (n - m) + 1$.

C/C++	Паскаль	Бейсик
<pre>//c4_9 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_STYPE, "Russian"); int n, m, r; printf("задай n, m --> "); scanf("%i %i", &n, &m); r=(n/m)*(n-m)+1; printf("\n Ответ - %d", r); getch(); }</pre>	<pre>{p4_9} var n, m, r: integer; begin write('Задай m, n --> ='); readln(m, n); r:=(n div m)*(n-m)+1; writeln('Ответ - ',r:3); end.</pre>	<pre>'b4_9 DIM n, m AS INTEGER CLS INPUT " m, n --> ="; m, n r=(n\m)*(n-m)+1 PRINT "Ответ - "; r END</pre>

Задача 4.10. Алгоритм. Применить алгоритм задачи 3.3, тогда $r = (z_1 - z_2) * (z_1 - z_3) * (z_1 - z_4) * (z_2 - z_3) * (z_2 - z_4) * (z_3 - z_4)$.

C/C++	Паскаль	Бейсик
<pre>//c4_10 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_STYPE, "Russian"); int n, z1, z2, z3, z4, r; printf("задай n --> "); scanf("%i", &n); z1=n%10; z2=n/10%10; z3=n/100%10; z4=n/1000; r=(z1-z2)*(z1-z3)*(z1-z4)*(z2- z3)*(z2-z4)*(z3-z4); printf("\n Ответ-%d", r); getch(); }</pre>	<pre>{p4_10} var n, z1, z2, z3, z4, r: integer; begin write('Задай n --> '); readln(n); z1:=n mod 10; z2:=n div 10 mod 10; z3:=n div 100 mod 10; z4:=n div 1000; r:=(z1-z2)*(z1-z3)*(z1-z4)* (z2-z3)*(z2-z4)*(z3-z4); writeln('Ответ- ', r:3); end.</pre>	<pre>'b4_10 DIM n, z1, z2, z3, z4, r AS INTEGER CLS INPUT "Задай n -->";n z1=n mod 10 z2=n\10 mod 10 z3=n\100 mod 10 z4=n\1000 r=(z1-z2)*(z1-z3)*(z1-z4)* (z2-z3)*(z2-z4)*(z3-z4); PRINT "Ответ- ";r END</pre>

Олимпиадные задачи

Задача 5.1. Алгоритм.

Обозначим:

n — номер квартиры;

n_p — номер подъезда;

n_i — номер этажа.

Найдем номер подъезда: $n_p = (n - 1) / 36 + 1$.

Найдем, сколько квартир осталось: $n_o = n - (n_p - 1) * 36$.

Найдем номер этажа: $n_i = (n_o - 1) / 4 + 1$.

Примечание. Наглядная задача для введения понятия — составление контрольных примеров.

C/C++	Паскаль	Бейсик
<pre>//c5_1 #include <conio.h> #include <iostream.h> #include <stdlib.h> main() { // установка русского языка setlocale(LC_STYPE, "Russian"); int n, n_p, n_i, n_o; cout<<"Задай № квартиры --> "; cin>>n; n_p=n-(n_p-1)*36; n_o=n%36; n_i=(n_o-1)/4+1; cout<<"n_p="<< n_p<<endl; cout<< " n_i="<< n_i<<endl; getch(); }</pre>	<pre>{p5_1} var n, n_p, n_i, n_o: integer; begin write('Задай № квартиры --> '); readln(n); n_p:=(n-1) div 36+1; n_o:=n-(n_p-1)*36; n_i:=(n_o-1) div 4+1; writeln('№ подъезда =', n_p:3); writeln('№ этажа =', n_i:3); end.</pre>	<pre>'b5_1 DIM n, np, ni, no AS INTEGER CLS INPUT "N кв. "; n np=(n-1)\36+1 no=n-(np-1)*36 ni=(no-1)\4+ 1 PRINT "N подъезда =" ; np PRINT "N этажа =" ; ni END</pre>

Задача 5.2. Алгоритм. Задача 5.1 — частный случай задачи 5.2. Для проверки работы программы следует использовать контрольный пример предыдущей задачи.

C/C++	Паскаль	Бейсик
<pre>//c5_2 #include <conio.h> #include <iostream.h> #include <stdlib.h> main() { setlocale(LC_STYPE, "Russian"); int n, n_p, n_i, n_o, kk, ki; cout<<"Задай кол-во квартир на этаже --> "; cin>>kk; cout<<"Задай кол-во этажей --> "; cin>>ki; cout<<"Задай № квартиры --> "; cin>>n; }</pre>	<pre>{p5_2} var n, n_p, n_i, n_o, kk, ki: integer; begin write('Задай кол-во квартир на этаже --> '); readln(kk); write('Задай кол-во этажей --> '); readln(ki); write('Задай № квартиры --> '); readln(n); n_p:=(n-1) div (kk*ki)+1; n_o:=n-(n_p-1)*kk*ki;</pre>	<pre>'b5_2 DIM n, n_p, n_i, n_o, kk, ki AS INTEGER CLS INPUT "Задай кол-во кв. на эта- же --> "; kk INPUT "Задай кол-во этажей --> "; ki INPUT "Задай № кв. --> "; n np=(n-1)\(ki*kk)+1 no=n-(np-1)*36 ni=(no-1)\4+1 PRINT "№ подъезда =" ; n_p</pre>

Окончание таблицы

C/C++	Паскаль	Бейсик
<pre>n_p=(n-1)/(kk*ki)+1; n_o=n-(n_p-1)*kk*ki; n_i=(n_o-1)/kk+1; cout<<"№ подъезда = "<< n_p<<endl; cout<< "№ этажа ="<< n_i<<endl; getch(); }</pre>	<pre>n_i:=(n_o-1) div kk+1; writeln('№ подъезда =', n_p:3); writeln('№ этажа =', n_i:3); end.</pre>	<pre>PRINT "№ этажа ="; ni END</pre>

Задача 5.3. Алгоритм. $n = (s - 20) / 25$. В случае затруднения надо разобрать задачу на конкретном примере.

C/C++	Паскаль	Бейсик
<pre>//c5_3 #include <conio.h> #include <iostream.h> #include <stdlib.h> main() { setlocale(LC_CTYPE, "Russian"); int n, s; cout<<"Задай сумму денег в копейках --> ";cin>>s; n=(s-20)/25; cout<<"мак кол-во бутылок = "<< n<<endl; getch(); }</pre>	<pre>{p5_3} var n, s: integer; begin writeln('Задай сумму денег в копейках --> '); readln(s); n:=(s-20) div 25; writeln('мак кол-во бутылок = ', n:3); end.</pre>	<pre>'b5_3 DIM n, s AS INTEGER CLS INPUT "Задай сумму денег в коп.";s n=(s-20)\25 PRINT " мах кол-во бутылок ="; n END</pre>

Задача 5.4. Алгоритм. Часовая стрелка, перемещаясь между двумя соседними цифрами, пройдет путь в $360^\circ/12 = 30^\circ$. За это время минутная стрелка сделает полный круг, т. е. 60 мин. Поэтому, пока часовая стрелка поворачивается на 1° , минутная проходит 2 минутных деления. Минутное деление соответствует $360^\circ/60 = 6^\circ$.

Количество часов $h = u / 30$; количество минут $m = u \% 30 * 2$; угол минутной стрелки $um = m * 6$.

C/C++	Паскаль	Бейсик
<pre>//c5_4 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_CTYPE, "Russian"); int u, h, m, um; printf("задай угол --> "); scanf("%i",&u) ; h=u/30; m=u%30*2 ; um=m*6; printf("\n h=%d, m=%d, um=%d", h, m, um); getch(); }</pre>	<pre>{p5_4} var u, h, m, um: integer; begin write('задай угол --> '); readln(u); h:=u div 30; m:=u mod 30*2 ; um:=m*6; writeln('h=', h:3, ' m=', m:3, ' um=', um:3); end.</pre>	<pre>'b5_4 DIM u, h, m, um AS INTEGER CLS INPUT "задай угол"; u h=u\30 m=u MOD 30*2 um=m*6 PRINT "h="; h, "m="; m, "um="; um END</pre>

Задача 5.5. Алгоритм. Найти, на какой угол uh повернется часовая стрелка, затем минутная um . Модуль разности даст искомый угол $u = \text{abs}(uh - um)$. Часовая стрелка за час перемещается на 30° , минутная делает полный оборот, т. е. 360° , тогда искомый угол минутной стрелки $um = 360 / 60 * \text{показание минутной стрелки}$. Часовая стрелка при перемещении минутной на 2° переместится на 1° , т. е. искомый угол будет в два раза меньше показания минутной стрелки + время в часах * 30.

C/C++	Паскаль	Бейсик
<pre>//c5_5 #include <conio.h> #include <iostream.h> #include <stdlib.h> #include <math.h> main() { setlocale(LC_CTYPE, "Russian"); int uh, m, t, u, um; printf("задай час --> ");</pre>	<pre>{p5_5} var uh, m,t,u, um: integer; begin write('задай час --> '); readln(t); write('задай минуты --> '); readln(m); um:=60*m; uh:=t*30+m div 2; u:=abs(uh-um);</pre>	<pre>'b5_5 DIM uh, m, t, u, um AS INTEGER CLS INPUT "задай час -->"; t INPUT "задай минуты --> "; m um=60*m uh=t*30+m\2 u=abs(uh-um) PRINT "u="; u) END</pre>

Окончание таблицы

C/C++	Паскаль	Бейсик
<pre>scanf("%i",&t) ; printf("задай минуты --> "); scanf("%i",&m) ; um=60*m; uh=t*30+m/2; u=abs(uh-um); printf("\n u=%d", u); getch(); }</pre>	<pre>writeln('u=', u); end.</pre>	

Приложение 3

Типичные ошибки

При выполнении операций целочисленного деления.

- Арифметические ошибки.
- Ошибки в обозначении операций целочисленного деления и деления с остатком.
- Нарушен приоритет выполнения арифметических операций.
- В процессе выполнения программы происходит переприсвоение/обновление данных.

При написании программ.

- Не подключена одна из библиотек.
- В стандартном C/C++ текст программы рекомендуется писать строчными буквами, в DEV-C++ — любыми.
- В C/C++ библиотеки ввода/вывода — стандартная (stdio.h) и потоковая (iostream.h) — могут конфликтовать. Ввод/вывод должен осуществляться операторами, относящимися только к одной из библиотек.
- Вместо знака арифметической операции деление с остатком (%) использован другой символ или обозначение из другого языка программирования.
- В C/C++ целочисленное деление обозначается символом «/».
- Не стоит знак «&» в операторе ввода scanf.
- Отсутствуют «()» в операторе getch().
- Отсутствуют «()» у функции main().
- После main() нельзя ставить «;».
- Пропущен один из символов «{, }», ограничивающий тело функции main.
- Пропущен знак «%», с которого начинаются форматы вывода и ввода.

НОВОСТИ

Американская школьница разработала метод борьбы с интернет-троллингом

Школьница из американского города Чикаго Триша Прабху вышла в финал конкурса Global Science Fair, который проводит корпорация Google, благодаря разработанному ей проекту по борьбе с интернет-троллингом.

Метод 13-летней американки основан на том факте, что подростки, которые часто оскорбляют друг друга в виртуальной среде, действуют импульсивно и отправляют обидные и раздражающие собеседника сообщения, не задумываясь о последствиях.

По словам Триши Прабху, у людей младше 25 лет не развита так называемая префронтальная кора головного мозга — участок, отвечающий за самоконтроль и помогающий думать, прежде чем действовать. Поэтому, считает финалистка конкурса Google, если подростку, который пытается затроллить собеседника, дать время

на раздумья о целесообразности этого поступка, то он, вероятнее всего, откажется от своих действий.

Чтобы проверить свою гипотезу, американка создала специальную систему под названием Rethink (в переводе с английского «переосмысление»), которая предлагала участвующим в эксперименте подросткам подумать о возможных последствиях, прежде чем они отправят комментарий. В более чем 93 % случаев участники исследования меняли свое мнение и отказывались от публикации сообщения.

После того как гипотеза Триши Прабху подтвердилась, она намерена создать реальный продукт, который сможет работать в социальных сетях и мобильных приложениях, предотвращая троллинг. Система по ключевым словам будет определять, является ли комментарий обидным для собеседника, и предлагать его автору подумать лишнюю секунду, прежде чем отправлять его.

(По материалам «Российской газеты»)

XI КОНКУРС НАУЧНО-ПРАКТИЧЕСКИХ РАБОТ ПО МЕТОДИКЕ ОБУЧЕНИЯ ИНФОРМАТИКЕ И ИНФОРМАТИЗАЦИИ ОБРАЗОВАНИЯ ИНФО-2014

**Издательство «Образование и Информатика»,
Всероссийское научно-методическое общество педагогов
объявляют о проведении
в 2014 году конкурса по следующим номинациям:**

- **Облачные технологии в учебном процессе.**
- **Активные методы обучения на уроках информатики.**
- **Совершенствование подготовки учителей информатики в свете требований ФГОС общего образования.**
- **Опыт внедрения программных продуктов на платформе «1С:Предприятие» в практику деятельности образовательной организации.**
- **Лучший ИУМК по внедрению программных продуктов на платформе «1С:Предприятие».**

Руководит конкурсом **Организационный комитет** (далее — Оргкомитет), состоящий из представителей Российской академии образования, ведущих методистов, членов Всероссийского научно-методического общества педагогов, членов редакционных советов журналов «Информатика и образование» и «Информатика в школе», сотрудников объединенной редакции журналов.

Цели и задачи конкурса

1. Выявление и поддержка талантливых педагогов, методистов, руководителей образовательных учреждений и органов управления образованием, использующих в профессиональной деятельности информационно-коммуникационные технологии.
2. Включение педагогов, методистов, руководителей образовательных учреждений и органов управления образованием в деятельность по разработке нового содержания образования, новых педагогических технологий, методик обучения и управления образованием.
3. Создание информационно-образовательного пространства на страницах журналов «Информатика и образование» и «Информатика в школе» по обмену и распространению опыта использования средств информационно-коммуникационных технологий в педагогической деятельности и в области управления образованием.
4. Повышение информационной культуры и информационно-коммуникационной компетентности всех участников образовательного процесса — учащихся, педагогов, родителей.

Конкурс проводится с 20 сентября по 20 декабря 2014 года.

Работы на конкурс принимаются до 20 декабря 2014 года включительно. Работы, присланные позже этой даты, к участию в конкурсе допускаться не будут.

Итоги конкурса будут опубликованы на сайтах Всероссийского научно-методического общества педагогов (<http://www.vnmpor.ru/>) и издательства «Образование и Информатика» (<http://www.infojournal.ru/>), а также в номерах 1–2015 журналов «Информатика и образование» и «Информатика в школе».

Лучшие работы будут опубликованы в журналах «Информатика и образование» и «Информатика в школе».

Победители получают призы от партнеров конкурса, а также:

- диплом от Всероссийского научно-методического общества педагогов и издательства «Образование и Информатика» (один групповой диплом — если работа представлена группой авторов);
- по одному экземпляру журналов «Информатика и образование» № 1–2015 и «Информатика в школе» № 1–2015, в которых будут опубликованы итоги конкурса;
- авторский экземпляр журнала с опубликованной работой.

Подробную информацию о конкурсе вы можете найти на сайтах организаторов:

<http://www.vnmpor.ru/> — Всероссийское научно-методическое общество педагогов

<http://www.infojournal.ru/> — Издательство «Образование и Информатика»

Контакты Оргкомитета

Телефон: (495) 708-36-15

E-mail: readinfo@infojournal.ru

<http://www.vnmpor.ru/> — Всероссийское научно-методическое общество педагогов

<http://www.infojournal.ru/> — Издательство «Образование и Информатика»



А. Л. Каткова,
Шадринский государственный педагогический институт

НЕКОТОРЫЕ ОСОБЕННОСТИ ПОДГОТОВКИ УЧАЩИХСЯ К ГОСУДАРСТВЕННОЙ ИТОГОВОЙ АТТЕСТАЦИИ ПО ИНФОРМАТИКЕ И ИКТ

Аннотация

В статье рассматривается задание 20 части 3 ГИА по информатике и ИКТ, представляющее собой практическое упражнение, которое необходимо выполнить на компьютере. Автор показывает различные варианты составления алгоритмов, приводящих к заданному результату.

Ключевые слова: ГИА по информатике и ИКТ, алгоритм, учебная среда разработки «КуМир», исполнитель Робот.

Контактная информация

Каткова Алла Леонидовна, канд. пед. наук, доцент кафедры прикладной информатики и экономики Шадринского государственного педагогического института; *адрес:* 641882, Курганская обл., г. Шадринск, ул. К. Либкнехта, д. 3; *телефон:* (919) 591-76-77; *e-mail:* AllaKatkova@mail.ru

A. L. Katkova,
Shadrinsk State Pedagogical Institute

TRAINING STUDENTS FOR THE STATE FINAL CERTIFICATION IN INFORMATICS AND ICT

Abstract

The article deals with the task 20 of the third part of SFC in informatics and ICT that is the practical exercise to be performed on the computer. The author shows various variants of making algorithms leading to the desired result.

Keywords: SFC in informatics and ICT, algorithm, learning environment "KuMir", performer Robot.

Выпускные экзамены никого не оставляют равнодушными: ни школьников, ни родителей, ни учителей. Учащиеся старательно готовятся к государственной итоговой аттестации (ГИА) — основному государственному экзамену (ОГЭ) в девятых классах и единому государственному экзамену (ЕГЭ) — в одиннадцатых, листают учебники, нанимают репетиторов, находят необходимые сайты, анализируют полученную информацию.

Основной государственный экзамен — это форма государственного контроля освоения выпускниками девятого класса основных общеобразовательных программ [2]. Сдача ОГЭ необходима для перехода в десятый класс или поступления в учреждения среднего профессионального образования (колледжи и техникумы). Два экзамена являются обязательными: русский и математика, экзамены по другим предметам обучающиеся сдают на добровольной основе по своему выбору. Именно с этим и возникают сложности у школьников — встает вопрос: какой предмет выбрать и как подготовиться. Рассмотрим некоторые особенности подготовки к ГИА по информатике и ИКТ.

Прежде всего следует обратить внимание на количество вопросов в экзамене. Часть 1 включает шесть заданий (1–6) с выбором ответа. К каждому заданию дается четыре варианта ответа, из которых только один верный. Часть 2 включает двенадцать заданий (7–18) с кратким ответом. Часть 3 представляет собой практическое задание, которое необходимо выполнить на компьютере. Она содержит два задания (19, 20), на которые следует дать развернутый ответ. Решением для каждого задания является файл, который необходимо сохранить.

Занимаясь подготовкой к ГИА, ученики отмечают, что информации по первым двум частям достаточно, но встает вопрос, как разобраться с третьей, где необходимы практические навыки работы на компьютере в специальных программах. В помощь учащимся проанализируем подробно задание 20 части 3 и представим его решение несколькими различными способами. Для примера используем демонстрационный вариант ГИА по информатике и ИКТ 2014 года [1].

Задание.

На бесконечном поле есть горизонтальная и вертикальная стены. Левый конец горизонтальной стены соединен с нижним концом вертикальной стены. Длины стен неизвестны. В вертикальной стене есть ровно один проход, точное место прохода и его ширина неизвестны. Робот находится в клетке, расположенной непосредственно над горизонтальной стеной у ее правого конца.

На рисунке 1 указан один из возможных способов расположения стен и Робота (Робот обозначен буквой «Р»).

Напишите для Робота алгоритм, закрашивающий все клетки, расположенные непосредственно левее и правее вертикальной стены. Проход должен остаться незакрашенным. Робот должен закрасить только клетки, удовлетворяющие данному условию. Например, для приведенного выше рисунка Робот должен закрасить следующие клетки (рис. 2).

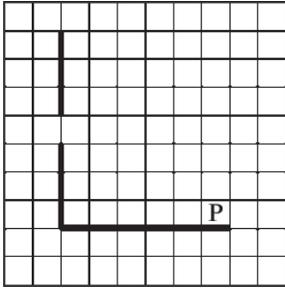


Рис. 1

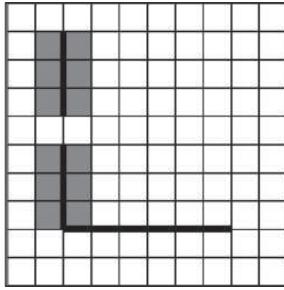


Рис. 2

При выполнении алгоритма Робот не должен разрушиться, выполнение алгоритма должно завершиться. Конечное расположение Робота может быть произвольным.

Алгоритм должен решать задачу для любого допустимого расположения стен и любого расположения и размера прохода внутри стены.

В спецификаторе для выполнения задания 20.1 рекомендуется использование учебной среды исполнителя Робот. В качестве такой среды может использоваться, например, учебная среда разработки «КуМир» (НИИСИ РАН, <http://www.niisi.ru/kumir>) или любая другая среда, позволяющая моделировать работу исполнителя Робот [3].

Для решения задач такого типа установим и запустим учебную среду разработки «КуМир» (рис. 3).

Укажем, что будем использовать исполнителя Робот, для этого удалим знак | перед словосочетанием «использовать Робот». При этом ранее подсвеченное серым цветом слово «Робот» изменит окраску на зеленую (рис. 4).

Как видно, структура программы уже задана исполнителем. Прописаны начало и конец алгоритма. Остается только верно написать тело программы. Но прежде следует ввести указанную в задании информацию: построить начальную обстановку поля и установить исполнителя в позицию.

Для этого выберем в верхнем меню вкладку **Инструменты** и команду **Редактировать стартовую обстановку Робот**. На появившемся поле прорисовываем стены и устанавливаем исполнителя в нужное положение (рис. 5). Сохраняем обстановку как стартовую (**Робот, Сменить стартовую обстановку**).

Чтобы дойти до стены, можно использовать команду «влево» несколько раз, но в задании сказано, что «алгоритм должен решать задачу для любого допустимого расположения стен», т. е. стена может располагаться на

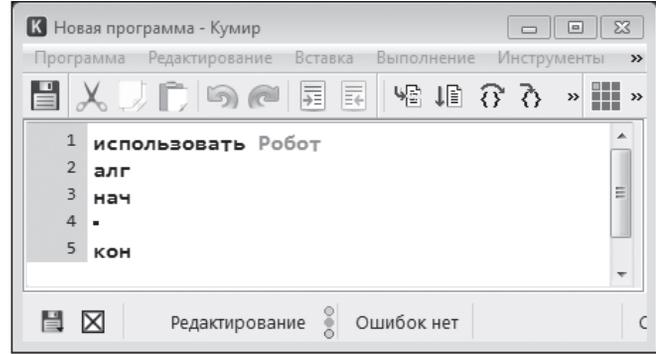


Рис. 4

любом расстоянии от исполнителя, поэтому следует двигаться неизвестное число шагов влево, пока не встретим стену. Эту задачу можно реализовать с помощью создания цикла с условием «пока слева свободно» (рис. 6).

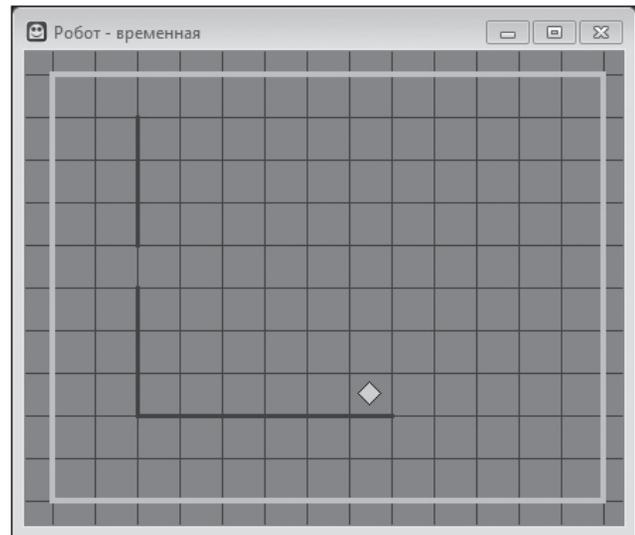


Рис. 5

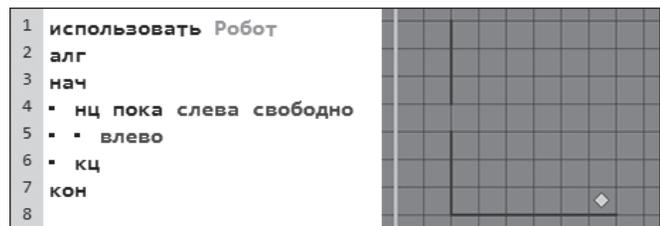


Рис. 6

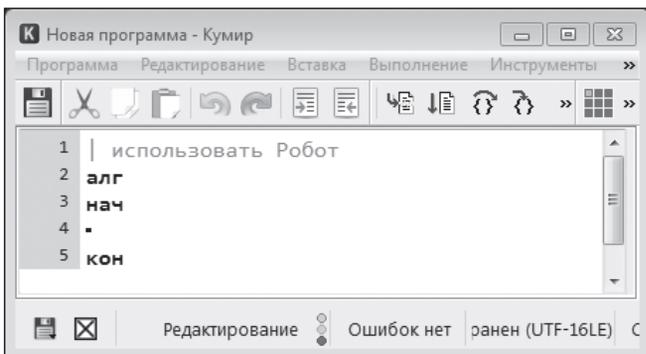


Рис. 3

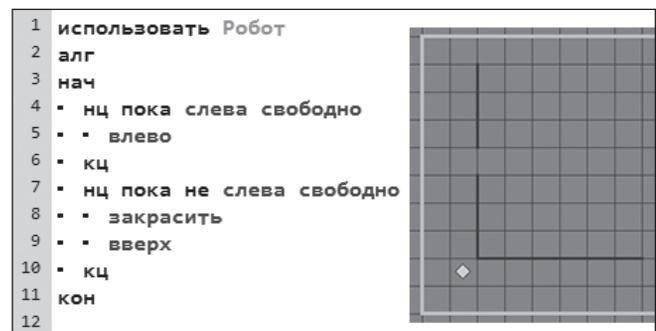


Рис. 7

Теперь нужно двигаться вверх и закрашивать клетки, значит, исполнителю придется изменить направление «вверх» и, распознавая стену, построенную слева, закрашивать поле до тех пор, пока слева несвободно (рис. 7).

Подобным образом, используя цикл, проходим там, где нет стены, не закрашивая клетки, и снова повторяем ту часть программы, которая отвечает за заливку клеток цветом около стены. Затем разворачиваем исполнителя двумя дополнительными командами «влево» и «вниз», приготовив Робота для движения вниз рядом со стеной, которая теперь оказывается справа. В итоге получается следующая программа:

```
использовать Робот
алг
нач
. нц пока слева свободно
. . влево
. кц
. нц пока не слева свободно
. . закрасить
. . вверх
. кц
. нц пока слева свободно
. . вверх
. кц
. нц пока не слева свободно
. . закрасить
. . вверх
. кц
. влево
. вниз
. нц пока не справа свободно
. . закрасить
. . вниз
. кц
. нц пока справа свободно
. . вниз
. кц
. нц пока не справа свободно
. . закрасить
. . вниз
. кц
кон
```

Отметим, что учащимся выражение «не справа свободно» логически непонятно, но исполнитель Робот позволяет использовать команду «справа стена», которая является эквивалентом первой команды. При изменении алгоритма исполнитель дает такой же результат, но команды интуитивно понятны учащимся, поскольку в них нет отрицания.

Хорошим тоном в программировании считается алгоритм, дающий верный результат за меньшее количество шагов. В предыдущем примере количество строк алгоритма было 31. Их можно уменьшить за счет добавления в цикл условий, например, представленным ниже способом.

```
использовать Робот
алг
нач
. нц пока слева свободно
. . влево
. кц
. нц пока сверху свободно
. . если слева стена то
. . . . закрасить
. . . . вверх
. . . иначе
. . .
```

```
. . . . вверх
. . все
. кц
. влево
. нц пока снизу свободно
. . если справа стена то
. . . . закрасить
. . . . вниз
. . . иначе
. . . . вниз
. . все
. кц
кон
```

Одним из интересных способов решения задачи может предстать добавление логической переменной, которая будет менять направление движения исполнителя. Назовем ее «направление». Этот вариант алгоритма для тех, кто любит поразмышлять и добавить «изюминку» в программу. При прохождении Роботом стены, расположенной слева от него, переменная имеет значение «да», затем при повороте и обратном ходе исполнителя значение логической переменной изменяется на «нет».

```
использовать Робот
алг
нач
. лог направление
. направление:=да
. нц
. . если (слева свободно) и (справа свободно) и
(сверху свободно) и (снизу свободно) то
. . . . если направление то
. . . . . вверх
. . . . . иначе
. . . . . вниз
. . . . . если снизу стена то
. . . . . . выход
. . . . . все
. . . . все
. . . иначе
. . . . если сверху стена то
. . . . . влево
. . . . . вниз
. . . . . направление:=нет
. . . . все
. . все
. . если слева стена или справа стена то
. . . . закрасить
. . . . если направление то
. . . . . вверх
. . . . . иначе
. . . . . вниз
. . . . все
. . . иначе
. . . . если снизу стена то
. . . . . влево
. . . . все
. . все
. кц
кон
```

При изменении стартовой обстановки, не изменяющей условия задачи (удлинение стены, расширение и даже добавление дополнительного прохода), результат во всех вышеприведенных решениях останется прежним (рис. 8). Представленные программы универсальны.

Подводя итог, следует заметить, что задачи части 3 ГИА по информатике и ИКТ не столь сложно решаемые, как может показаться при прочтении задания. Школьникам требуется немного практики в написании алгоритмов, и обучающая среда «КуМир» подходит для этой

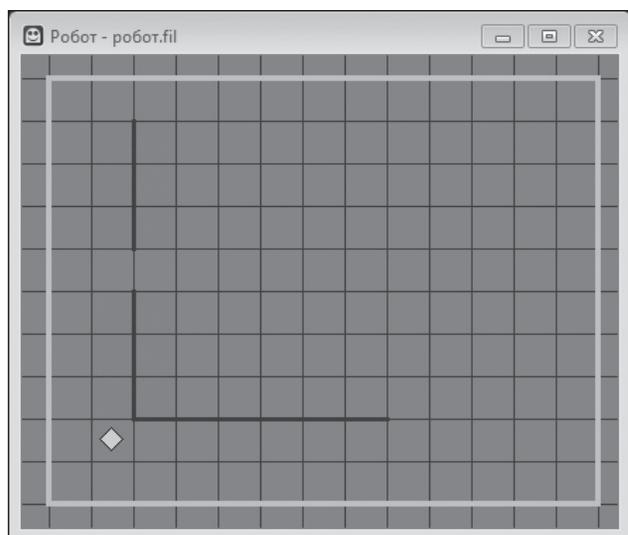


Рис. 8

цели. В отличие от частей 1 и 2 здесь учащийся может полностью раскрыться, проявить все свои знания и умения написания программ, а также показать достойные, не шаблонные способы решения.

Интернет-источники

1. Демонстрационный вариант контрольных измерительных материалов для проведения в 2014 году государственной (итоговой) аттестации (в новой форме) по информатике и ИКТ обучающихся, освоивших основные общеобразовательные программы основного общего образования. http://gia.edu.ru/ru/graduates_classes/demonstration/

2. Официальный информационный портал государственной итоговой аттестации. <http://gia.edu.ru/ru/main/brief-glossary/>

3. Спецификация контрольных измерительных материалов для проведения в 2014 году государственной (итоговой) аттестации (в новой форме) по информатике и ИКТ обучающихся, освоивших основные общеобразовательные программы основного общего образования. http://gia.edu.ru/ru/graduates_classes/demonstration/

НОВОСТИ

С TeamViewer интерактивное обучение стало доступно школьникам в удаленных регионах России

TeamViewer, приложение для удаленной поддержки и организации интерактивных конференций, используется в дистанционном обучении учащихся школы МиКЭБИ компании «Тренинг Сити». С помощью TeamViewer организовано одновременное интерактивное обучение двух-трех учеников, проживающих в российских регионах, с возможностью полного доступа к их компьютеру, сообщили CNews в компании TeamViewer.

«Мы пробовали разные платформы для организации дистанционного обучения, — рассказал генеральный директор компании «Тренинг Сити» Евгений Кузнецов, — однако только TeamViewer по своим возможностям устроил нас полностью. Мы смогли реализовать в обучении все необходимые функции. Следует отдельно отметить стабильность работы программы: при разрыве интернет-соединения, что, к сожалению, в регионах России пока не редкость, происходит автоматическое восстановление сеанса».

Компания «Тренинг Сити» работает на рынке профессионального обучения. Несколько лет назад «Тренинг Сити» открыла новое направление — обучение одаренных детей. Открытие дистанционного обучения по методике МиКЭБИ в учебном центре «Тренинг Сити» позволило получить к ней доступ детям из общеобразовательных школ, не работающих по данной программе. Ученики компании в основном проживают в российских столицах — Москве и Санкт-Петербурге, но также есть обучающиеся в регионах.

Преподаватели и учащиеся обеспечены полным набором современных технологий для коммуникации с эффектом присутствия в классе. Каждый ученик школы пользуется планшетом с выходом в Интернет,

выполняющим роль рабочей тетради, веб-камерой, а также монитором и телевизором. Программа удаленного доступа позволяет соединять в один сеанс два или три ученических планшета. Ученики в своих «рабочих тетрадях» выполняют письменные и устные задания преподавателя, получают от него файлы, просматривают видеоролики, а педагог, в свою очередь, может контролировать правильность выполнения заданий, рассказали в компании.

«С помощью TeamViewer получается очень динамичное общение — педагог не только общается с учениками, но также может внести корректирующие правки в «рабочей тетради» в режиме реального времени», — подчеркнул Евгений Кузнецов.

В дистанционной школе, занятия в которой проходят во внеурочное время, обучаются школьники младших классов, но проблем с техникой представители юного цифрового поколения не имеют. Родителям нужно лишь установить на планшет клиентскую часть программы, инструкции по настройке высылаются по электронной почте. Одна из полезных опций TeamViewer — запрет на закрытие сеанса удаленного доступа. Это позволяет избежать риска закрытия программы ребенком, отметили в компании.

«Машина педагога может контролироваться специалистом ИТ-службы также с помощью TeamViewer. Системный администратор имеет возможность либо напрямую подключиться к компьютерам обучающихся, либо, если это необходимо, сделать это посредством компьютера преподавателя (например, если требуется помощь в пересылке файла педагога детям)», — добавил Евгений Кузнецов.

(По материалам CNews)



М. В. Дактор,
средняя общеобразовательная школа № 16, Калининград

УРОК НА ТЕМУ «ПРОГРАММИРОВАНИЕ ПЕРЕВОДА ЧИСЕЛ ИЗ ОДНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ДРУГУЮ»*

Аннотация

В статье представлена методическая разработка урока на тему «Программирование перевода чисел из одной системы счисления в другую» (IX класс). Урок построен на основе системно-деятельностного подхода, предполагает реализацию личностно-ориентированного обучения, имеет практико-ориентированный характер.

Ключевые слова: ФГОС, системно-деятельностный подход, УУД, личностно-ориентированное обучение, урок открытия нового знания, учебный диалог, коллективное создание алгоритма, индивидуальная работа учащихся.

Контактная информация

Дактор Мария Валериевна, учитель информатики средней общеобразовательной школы № 16, Калининград; *адрес:* 236039, г. Калининград, ул. Багратиона, д. 107а; *телефон:* (401) 264-25-72; *e-mail:* MariaValerievna@mail.ru

M. V. Daktor,
School 16, Kaliningrad

LESSON ON THEME "PROGRAMMING THE CONVERSION NUMBERS FROM ONE NUMBER SYSTEM TO ANOTHER"

Abstract

The article presents the lesson on theme "Programming the conversion numbers from one number system to another" (grade 9). The lesson is based on a system-activity approach and involves the implementation of student-centered learning, has a practice-oriented nature.

Keywords: FSES, system-activity approach, universal educational actions, student-centered learning, lesson of discovering new knowledge, educational dialogue, collective creation of algorithm, individual student work.

Одна из целей изучения информатики в основной школе — развитие алгоритмического мышления, которое играет важную роль в формировании личности. Большинство специалистов разных стран сходятся во мнении, что основной акцент в содержании курса информатики основной школы необходимо делать на изучении фундаментальных основ информатики, выработке навыков алгоритмизации, знакомстве со средами программирования.

Изучение раздела «Программирование» начинается с понятия алгоритма. В начальной школе происходит знакомство с этим понятием на интуитивном уровне. В основной школе это понятие уточняется. При решении учебных задач обучающиеся знакомятся с разными способами записи алгоритмов, изучают свойства алгоритмов, знакомятся с алгоритмическими структурами и их кодированием на языках программирования. Формальное определение алгоритма формулируется при изучении основ теории алгоритмов в профильных классах старшей школы.

Вопросы по алгоритмизации и программированию входят в содержание контрольно-измерительных материалов ГИА и ЕГЭ.

Различные этапы всероссийской олимпиады школьников по информатике фактически представляют собой олимпиады по алгоритмизации и программированию.

Изучение программирования, направленное на понимание сути алгоритмов, их свойств, способов описания, способствует развитию четкости мышления, умению предельно уточнять предмет мысли, развивает такие важные качества, как внимательность, аккуратность, обстоятельность, убедительность в суждениях, умение абстрагироваться от конкретного содержания и сосредоточиться на структуре своей мысли.

В соответствии с программой курса «Информатика и ИКТ» авторского коллектива под руководством И. Г. Семкина разделы «Управление и алгоритмы» и «Введение в программирование» изучаются в девятом классе в количестве 12 и 15 часов соответственно. Для практической работы используются два вида учебных исполнителей алгоритмов, разработанных авторами и входящих в комплект ЦОР. Для изучения основ программирования используется язык Паскаль.

Тема «Программирование перевода чисел из одной системы счисления в другую» изучается в разделе «Введение в программирование». К этому времени обучающиеся:

- знают структуру программы на языке Паскаль;
- знакомы с основными алгоритмическими структурами;
- умеют реализовывать алгоритмические структуры средствами языка Паскаль;
- имеют представления о системах счисления;
- владеют алгоритмами перевода недесятичных чисел в десятичную систему счисления и алгоритмами перевода десятичных чисел в другие системы счисления.

* Материалы к статье можно скачать на сайте ИНФО: <http://www.infojournal.ru/infojournal/school/archive/8-2014>

В данной статье мы рассмотрим пример урока информатики на тему «Программирование перевода чисел из одной системы счисления в другую».

Раздел программы: Введение в программирование.

Тема урока: Программирование перевода чисел из одной системы счисления в другую.

Цели урока:

- создание условий для самостоятельной деятельности обучающихся по составлению программ перевода чисел;
- *деятельностная цель:* формирование способности обучающихся к новому способу действия — составлению программ для перевода двоичного числа в десятичную систему счисления и десятичного числа — в двоичную систему счисления;
- *содержательная цель:* расширение понятийной базы за счет включения в нее новых элементов — освоение техники работы с входными и выходными файлами при написании программ.

Решаемые учебные задачи:

- систематизировать и обобщить сведения, полученные обучающимися при изучении темы «Системы счисления»;
- систематизировать и обобщить сведения, полученные обучающимися при изучении темы «Программирование циклов»;
- расширить представления учащихся о возможностях ввода исходных данных и вывода результатов в среде программирования Turbo Pascal.

Тип урока: урок открытия нового знания.

Класс: девятый.

Формы работы на уроке: индивидуальная, групповая, коллективная.

Планируемые образовательные результаты:

предметные:

- умение составлять программы на языке программирования Паскаль для автоматического перевода чисел из двоичной системы счисления в десятичную и из десятичной системы счисления — в двоичную;
- умение осуществлять считывание данных из файла и производить запись данных в файл;

метапредметные:

- умение самостоятельно планировать пути достижения целей;
- умение соотносить свои действия с планируемыми результатами, осуществлять контроль своей деятельности, определять способы действий в рамках предложенных условий, корректировать свои действия в соответствии с изменяющейся ситуацией;
- умение оценивать правильность выполнения учебной задачи;
- навыки применения теоретических знаний в практической деятельности;

личностные:

- алгоритмическое мышление, необходимое для профессиональной деятельности в современном обществе;
- представление о программировании как сфере возможной профессиональной деятельности;

- формирование коммуникативной компетенции в общении и сотрудничестве со сверстниками в процессе творческой деятельности;
- формирование способности к саморазвитию и самообучению.

Основные понятия, изучаемые на уроке:

- системы счисления;
- алгоритм перевода двоичного числа в десятичную систему счисления;
- алгоритм перевода десятичного числа в другие системы счисления;
- формат оператора цикла с предусловием в Паскале;
- формат оператора цикла с постусловием в Паскале;
- техника работы с файлами при написании программ:
 - описать переменную файлового типа в разделе описания переменных;
 - связать файловую переменную с именем файла с помощью процедуры;
 - открыть файл для чтения/записи процедурами;
 - произвести чтение/запись из файла/в файл.

Оснащение урока:

оборудование:

- компьютер учителя;
- компьютеры учащихся;
- интерактивная доска;
- мультимедийный проектор;

программное обеспечение:

- презентация по теме урока, подготовленная в MS PowerPoint;
- авторское электронное учебное пособие: *Доктор М. В., Пузевич А. В.* Подготовка к олимпиадам по математике и информатике. 2012.

образовательные ресурсы из Единой коллекции ЦОР:

- Перевод недесятичных чисел в двоичную систему счисления. http://files.school-collection.edu.ru/dlrstore/1a264912-eca9-4b45-8d77-c3655b199113/9_110.swf
- Перевод десятичных чисел в другие системы счисления. http://files.school-collection.edu.ru/dlrstore/78ba290c-0f7c-4067-aaf4-d72f40f49f3b/9_109.swf

Литературные и интернет-источники, использованные при подготовке урока.

1. *Виноградова Н. Ф.* Современный взгляд на дидактику общеобразовательной школы в условиях введения новых ФГОС. М.: Педагогический университет «Первое сентября», 2013.
2. Информатика и ИКТ. Задачник-практикум: в 2 т. Т. 1 / под ред. И. Г. Семакина, Е. К. Хеннера. М.: БИНОМ. Лаборатория знаний, 2012.
3. Конструирование технологических карт урока в соответствии с требованиями ФГОС: сборник метод. материалов. Нижний Тагил: НТФ ГАОУ ДПО СО «ИРО», 2012.
4. *Семакин И. Г.* Авторская мастерская. <http://metodist.lbz.ru/authors/informatika/2/>
5. *Семакин И. Г. и др.* Информатика и ИКТ: учебник для 9 класса. М.: БИНОМ. Лаборатория знаний, 2012.
6. *Семакин И. Г., Шеина Т. Ю.* Преподавание базового курса информатики в средней школе: метод. пособие. М.: БИНОМ. Лаборатория знаний, 2012.

План урока.

1. Этап мотивации (самоопределения) учебной деятельности.
2. Этап актуализации и пробного учебного действия.
3. Этап выявления места и причины затруднения.
4. Этап построения проекта выхода из затруднения.
5. Этап реализации построенного проекта.
6. Этап первичного закрепления с проговариванием во внешней речи.
7. Физкультминутка. Упражнение для глаз.
8. Этап самостоятельной работы с самопроверкой по эталону.
9. Этап включения в систему знаний и повторения.
10. Домашнее задание.
11. Этап рефлексии учебной деятельности на уроке.

Ход урока

Урок сопровождается демонстрацией слайдов мультимедийной презентации, подготовленной учителем в MS PowerPoint.

1. Этап мотивации (самоопределения) учебной деятельности

Методический прием «Исправь ошибку». На экране демонстрируется слайд, на котором представлено соответствие чисел в десятичной и двоичной системах счисления. В записи есть ошибки. От учащихся требуется их исправить.

Учитель. Ребята, сегодня на уроке мы реализуем одну из функций инженерного калькулятора — перевод двоичного числа в десятичную систему счисления и перевод десятичного числа в двоичную систему счисления. На слайде представлено соответствие чисел в десятичной и двоичной системах счисления. В этой записи есть ошибки. Ваша задача — их исправить. (Демонстрируется слайд 1.)

2. Этап актуализации и пробного учебного действия

Учитель. Вспомнить алгоритмы перевода вам помогут цифровые образовательные ресурсы, размещенные в сети Интернет.

Исправь ошибки
$10 = 1010_2$
$11 = 1111_2$
$111_2 = 6$
$1101_2 = 13$

Слайд 1

Исправь ошибки
$10 = 1010_2$
$11 \neq 1111_2$
$111_2 \neq 6$
$1101_2 = 13$

Слайд 2

Исправь ошибки
$10 = 1010_2$
$11 = 1011_2$
$111_2 = 7$
$1101_2 = 13$

Слайд 3

Работа в группах
■ Группа 1. Перевод из десятичной системы счисления в двоичную
■ Группа 2. Перевод из двоичной системы счисления в десятичную
■ Группа 3. Цикл с предусловием. Формат оператора цикла в Паскале
■ Группа 4. Цикл с постусловием. Формат оператора цикла в Паскале

Слайд 4

Работа в группах	
Группа 1	Группа 2
$121 - x_2$ $1025 - x_2$ $1011011_2 - x_{10}$	$125 - x_2$ $1024 - x_2$ $1011111_2 - x_{10}$
Группа 3	Группа 4
Формат оператора цикла с предусловием в Паскале	Формат оператора цикла с постусловием в Паскале

Слайд 5

Учащиеся работают с ЦОР из Единой коллекции ЦОР: «Перевод недесятичных чисел в двоичную систему счисления» и «Перевод десятичных чисел в другие системы счисления».

Учитель. Итак, вы вспомнили алгоритмы перевода чисел. Исправим ошибки. (Демонстрируются слайды 2 и 3.)

На следующем слайде представлены задания для четырех групп. Вы можете самостоятельно выбрать задание и соответствующую группу. (Демонстрируются слайды 4 и 5.)

Учащиеся объединяются в группы, выбирают задание и выполняют его. Выбор консультанта в группе осуществляется обучающимися самостоятельно.

3. Этап выявления места и причины затруднения

Каждая группа отчитывается о выполнении задания.

Группы 1 и 2 показывают последовательность действий решения задачи, используя интерактивную доску. (Проверка — на слайде 6.)

Группы 3 и 4 представляют запись на Паскале операторов цикла с предусловием и постусловием с соблюдением синтаксиса. Демонстрация записи производится в среде программирования Turbo Pascal. (Проверка — на слайдах 7, 8.)

4. Этап построения проекта выхода из затруднения

Учитель. Алгоритмы перевода чисел и формат записи циклов в Паскале потребуются нам для решения главной задачи урока — составления программы перевода чисел.

Работа в группах

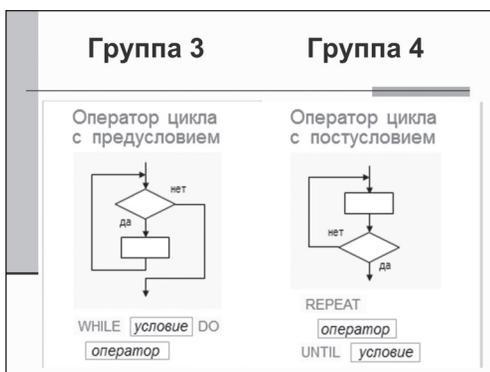
Группа 1	Группа 2
$121 - x_2$ $1025 - x_2$ $1011011_2 - x_{10}$	$125 - x_2$ $1024 - x_2$ $1011111_2 - x_{10}$
Группа 1 Ответы	Группа 2 Ответы
$121 = 1111001_2$ $1025 = 10000000001_2$ $1011011_2 = 91_{10}$	$125 = 1111101_2$ $1024 = 10000000000_2$ $1011111_2 = 95_{10}$

Слайд 6

Работа в группах

Группа 3	Группа 4
while <выражение> do <оператор> Пока истинно условие цикла, повторяется выполнение тела цикла	repeat <оператор> until <условие> Тело цикла всегда выполняется хотя бы один раз
While означает «пока», do – «делать», «выполнять». Оператор, стоящий после слова do, называется телом цикла. Телом цикла может быть простой или составной оператор, т. е. последовательность операторов между словами begin и end	Сначала выполняются операторы, составляющие тело цикла, затем проверяется условие, которое в данном случае является условием выхода из цикла, т. е. если оно ложно, то операторы цикла повторяются, иначе (если условие истинно) цикл завершается

Слайд 7



Слайд 8

Перевод двоичного числа в десятичную систему счисления	Перевод десятичного числа в двоичную систему счисления
<pre> k:=1; n10:=0; while (n2<>0) do begin n10:=n10+(n2 mod 10)*k; k:=k*2; n2:=n2 div 10; end; </pre>	<pre> k:=1; n2:=0; repeat n2:=n2+(n10 mod 2)*k; k:=k*10; n10:=n10 div 2 until (n10=0); </pre>

Слайд 9

Какой цикл мы будем использовать при составлении программы перевода двоичного числа в десятичную систему счисления?

Какой цикл мы будем использовать при составлении программы перевода десятичного числа в двоичную систему счисления?

(Коллективное обсуждение.)

5. Этап реализации построенного проекта

Совместно составляется фрагмент программного кода для:

- 1) перевода двоичного числа в десятичную систему счисления;
- 2) перевода десятичного числа в двоичную систему счисления (см. слайд 9).

Подготовка к олимпиадам по математике и информатике

Задачи для начинающих

Содержание

- ▶ Задачи для начинающих
 - ▶ Сумма чисел
 - ▶ Журавлики
 - ▶ Два бандита
 - ▶ Зарплата
 - ▶ Монетки
 - ▶ Нули
 - ▶ Конечные автоматы
- ▶ Задачи для самостоятельного решения
 - ▶ Неглухой телефон
 - ▶ Игра
 - ▶ Больше-меньше
 - ▶ Арбузы
 - ▶ Баскетбол
 - ▶ Эния
 - ▶ Разворот

Решения задач

```

program z_1_5;
var f1,f2:text;
a,b,s1,s2,i:byte;
begin
assign(f1,'input.txt');
reset(f1);
assign(f2,'output.txt');
rewrite(f2);
s1:=0; s2:=0;
for i:=1 to 4 do
begin
read(f1,a,b);
s1:=s1*a; s2:=s2*b;
end;
if s1=s2 then write(f2,'DRAW') else
if s1>s2 then write(f2,1) else write(f2,2);
close(f1);
close(f2);
end.
                
```

Фрагмент электронного учебного пособия «Подготовка к олимпиадам по математике и информатике»

6. Этап первичного закрепления с проговариванием во внешней речи

Работа с учебником (§ 2.1, с. 134).

Учитель. Проанализируем трассировочную таблицу, в которой представлено изменение значений переменных на каждом шаге выполнения алгоритма, представленного в программе.

7. Физкультминутка — упражнение для глаз

8. Этап самостоятельной работы с самопроверкой по эталону

Учащиеся записывают программный код на Паскале в среде программирования Turbo Pascal (ввод числа производится с клавиатуры) (см. Приложение).

9. Этап включения в систему знаний и повторения

Учитель. В разработанной программе ввод исходного числа производится с клавиатуры. Измените программу так, чтобы число для перевода считывалось из входного файла и результат выводился в выходной файл. Для выполнения данного задания повторите технологию чтения/записи из файла/в файл. Для этого откройте электронное учебное пособие «Подготовка к олимпиадам по математике и информатике» и выберите раздел «Техника работы с файлами при написании программ».

Формулирование выводов. (Демонстрируются слайды 10 и 11.)

Техника работы с файлами при написании программ

- Описать переменную файлового типа в разделе описания переменных
- Связать файловую переменную с именем файла с помощью процедуры
- Открыть файл для чтения/записи процедурами
- Произвести чтение/запись из файла/в файл

Слайд 10

Текстовый файл	
Чтение	Read (<Имя файловой переменной>, <Список переменных>); Readln (<Имя файловой переменной>, <Список переменных>); Оператор вводит все значения переменных из списка и передвигает указатель ввода до начала следующей строки.
Запись	Write (<Имя файловой переменной>, <Список переменных>); Writeln (<Имя файловой переменной>, <Список переменных>); Оператор выводит все значения переменных из списка и передвигает указатель вывода до начала следующей строки.

Слайд 11

10. Домашнее задание

Подготовить учебный проект по одной из следующих тем:

- Сложность алгоритмов.
- Языки программирования и трансляторы.
- История языков программирования.

11. Этап рефлексии учебной деятельности на уроке

Методический прием «Хочу спросить».

Учитель. Вы задаете вопрос, начиная со слов: «Хочу спросить...» Если вы довольны ответом, то поднимаете зеленую карточку. Если ответ вас не удовлетворил, то поднимаете красную карточку.

Методический прием «Оцени свое эмоциональное состояние».

Учитель. У каждого из вас на столе лежат смайлики — веселый и грустный. В конце урока вы мне отдаете тот смайлик, который соответствует вашему настроению.

Приложение

Программа перевода двоичного числа в десятичную систему счисления.

```

program z_1;
var n10, n2, k: longint;
begin
  write('n2=');
  readln(n2);      {ввод исходного двоичного числа}
  k:=1;
  n10:=0;
  while (n2<>0) do {цикл выполняется, пока n2
                  не равно нулю}
  begin
    n10:=n10+(n2 mod 10)*k; {суммирование
                            развернутой формы}
    k:=k*2;                 {вычисление степеней двойки}
    n2:=n2 div 10; {отбрасывание младшей цифры}
  end;
  writeln('n10=', n10); {вывод десятичного числа}
end.

```

Программа перевода десятичного числа в двоичную систему счисления.

```

program z_2;
var n10, n2, k: longint;
begin
  write('n10=');
  readln(n10); {ввод исходного десятичного числа}
  k:=1;
  n2:=0;
  repeat
    n2:=n2+(n10 mod 2)*k; {суммирование
                          развернутой формы}
    k:=k*10;              {вычисление базиса}
    n10:=n10 div 2       {целочисленное деление на 2}
  until (n10=0);        {цикл заканчивает выполнение
                        при n10=0}
  writeln('n2=', n2);  {вывод двоичного числа}
end.

```

Е. Л. Батакова,
лицей «Синтон», г. Чайковский, Пермский край

ИЗУЧЕНИЕ ЛИНЕЙНОГО АЛГОРИТМА С ПОМОЩЬЮ ИНТЕРАКТИВНЫХ СРЕДСТВ ОБУЧЕНИЯ

Аннотация

В статье представлена методическая разработка урока информатики, на котором учащиеся знакомятся с линейным алгоритмом при активном применении интерактивных средств обучения.

Ключевые слова: интерактивные средства обучения, информатика, новая информационно-образовательная среда, образовательные результаты, решение познавательных задач.

Контактная информация

Батакова Евгения Леонидовна, учитель информатики лицея «Синтон», г. Чайковский, Пермский край; *адрес:* 617762, Пермский край, г. Чайковский, ул. Азина, д. 1/1; *телефон:* (342-41) 7-68-50; *e-mail:* hermanny@mail.ru

E. L. Batakova,
Lyceum "Synthon", Tchaikovsky, Perm Region

STUDYING LINEAR ALGORITHM USING INTERACTIVE LEARNING TOOLS

Abstract

The article presents an informatics lesson on which students learn the linear algorithm, with the active use of interactive learning tools.

Keywords: interactive learning tools, informatics, new information educational environment, educational outcomes, decision of cognitive problems.

Раздел программы: Алгоритмы.

Тема урока: Линейный алгоритм.

Цели урока: формирование у учащихся понятия «линейный алгоритм» и освоение обучающимися принципа его использования при решении задач.

Задачи урока:

образовательные:

- ученики должны знать определение понятия «линейный алгоритм»;
- ученики должны овладеть принципами использования линейного алгоритма, уметь его применять при решении задач и реализовывать предложенный порядок действий;
- учитель должен стимулировать интерес учащихся к изучаемому разделу программы с помощью интерактивных средств обучения;

развивающие:

- развивать умение анализировать условие задачи, опираясь на знания, полученные в ходе изучения новой темы «Линейный алгоритм» и предыдущей темы «Что такое алгоритм»;
- развивать умение моделировать решение задачи, используя линейный алгоритм с целью получения требуемого результата;
- развивать умение программировать линейный алгоритм при решении задач, что к концу урока должно переквалифицироваться в навык;

воспитательные:

- развивать умение слушать, корректно задавать вопросы;
- развивать навыки взаимопомощи и самоконтроля;
- формировать умение работать самостоятельно, сосредотачиваться на поставленной задаче;
- способствовать освоению учащимися навыков контроля и самоконтроля в процессе нахождения решения задачи.

Тип урока: урок изучения и первичного закрепления новых знаний.

Предполагаемые образовательные результаты:

- *личностные:* освоены новые знания; развито умение слушать, корректно задавать вопросы; сформированы навыки взаимопомощи;
- *метапредметные:* развито умение анализировать условие задачи, точно и последовательно выполнять цепочку действий, опираясь на знания, полученные на уроке;
- *предметные:* сформировано определение понятия «линейный алгоритм» и усвоен принцип использования линейного алгоритма; развито умение применять данный алгоритм при решении задач и реализовывать любой предложенный порядок действий; освоены навыки программирования линейного алгоритма при решении задач.

Применяемые интерактивные средства обучения:

- *технические ИСО:*
 - компьютер;
 - интерактивная доска SMART Board (или экран с проектором);
- *программные ИСО:* программа Notebook 10;
- *тематические (предметные) ИСО:*
 - демонстрация принципа работы линейного алгоритма (слайд 1);
 - рабочее поле «Выполните задания...» (слайд 2);
 - диалоговые окна, определяющие условия заданий (слайды 3–6);

- анимационные элементы (структура и определение линейного алгоритма, ответ для задания 2);
- блок-схема линейного алгоритма (слайд 7).

План урока.

1. Организационный момент.
2. Актуализация знаний с элементами знакомства с новым материалом и повторением полученных ранее знаний.
3. Изучение нового материала посредством решения задач.
4. Подведение итогов урока.
5. Домашнее задание.

Ход урока

1. Организационный момент

Учащиеся занимают места в классе и настраиваются на продуктивную работу.

На протяжении всего урока учащихся сопровождает комплекс ИСО (разработан автором статьи), содержащий семь слайдов, воспроизводимых на интерактивной доске.

В начале урока учитель выдает учащимся форму «Бланк ответов» (рис. 1), которую они заполняют в процессе изучения темы. Она помогает учащимся проследить выполненную работу (входные, выходные данные) и формализует мышление учащихся, т. е. умение формулировать мысль или действие двумя-тремя словами (в виде алгоритма).

2. Актуализация знаний с элементами знакомства с новым материалом и повторением полученных ранее знаний

Слайд 1 (рис. 2, 3) — это титульный слайд изучаемой темы, с помощью которого учитель демонстрирует работу линейного алгоритма (на слайде представлена реализация линейного алгоритма).

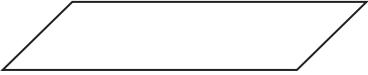
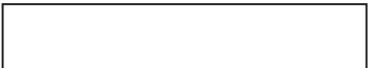
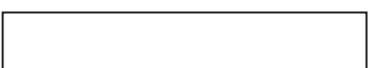
БЛАНК ОТВЕТОВ	
Вписать в соответствующие фигуры краткий (2–3 слова) ответ на вопрос: «Что требуется сделать в задаче?»	
Начальные условия	
Задача 1	
Задача 2	
Задача 3	
Задача 4	
Результат	

Рис. 1. Бланк ответов

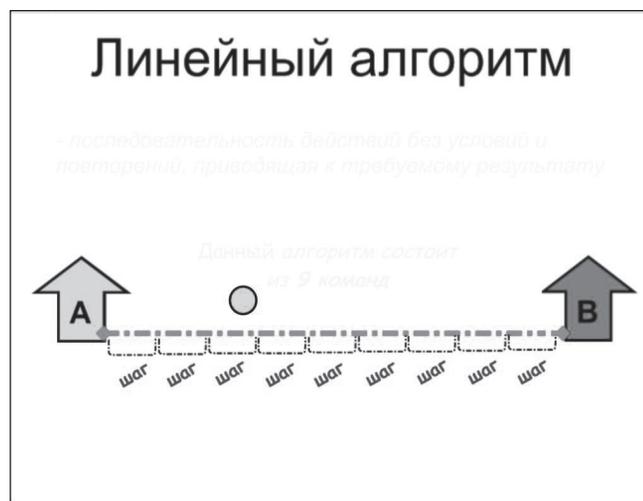


Рис. 2. Реализация линейного алгоритма



Рис. 3. Определение линейного алгоритма

После визуализации алгоритма учитель анализирует с учащимися работу линейного алгоритма, его особенности и формулирует определение (на данный объект настроена анимация, для того чтобы учащиеся сами или с помощью педагога вывели определение линейного алгоритма).

3. Изучение нового материала посредством решения задач

На **слайде 2** (рис. 4) представлена задача, состоящая из четырех заданий (этапов) и начальных условий, конечный результат которой можно получить, только выполнив все задания (пройдя все этапы). В качестве главного инструмента реализации алгоритма предлагается исполнитель алгоритмов «Стрелочка». На слайде изображена блок-схема линейного алгоритма — для того чтобы учащиеся изначально привыкали к работе с данной структурой. На графические фигуры «Задание n » ($n = 1..4$) настроены гиперссылки, направляющие на слайды с текстами соответствующих заданий.

Слайд 3 (рис. 5) — это текст **задания 1**. В этом задании учащимся дан готовый линейный алгоритм, их задача — определить конечный результат работы этого алгоритма в среде исполнителя «Стрелочка». Цели зада-

Линейный алгоритм
С помощью исполнителя алгоритма "Стрелочка":

начальное положение: левый верхний угол, направление - налево

Задание 1

Задание 2

Задание 3

Задание 4

Результат= ????????

выполните задания...

Рис. 4. Главная задача

Задание 1

Реализуйте алгоритм:

Начало алгоритма

шаг	поворот
шаг	прыжок
поворот	поворот
поворот	поворот
шаг	поворот
шаг	шаг
поворот	шаг
поворот	поворот
поворот	поворот
шаг	поворот
шаг	шаг
поворот	шаг

Конец алгоритма

26 команд

Рис. 5. Задание 1

Задание 2

Усовершенствуйте алгоритм "Цифра три" (состоит из 26 команд) так, чтобы программа состояла из **24 команд**

Ответ:

Рис. 6. Задание 2 — постановка задачи

Задание 2

Усовершенствуйте алгоритм "Цифра три" (состоит из 26 команд) так, чтобы программа состояла из **24 команд**

Ответ:

Начало алгоритма	
шаг	шаг
шаг	шаг
поворот	поворот
поворот	поворот
поворот	поворот
шаг	прыжок
шаг	поворот
шаг	поворот
поворот	поворот
поворот	шаг
поворот	шаг
Конец алгоритма	

Рис. 7. Задание 2 — ответ

ния: закрепить у учащихся знание структуры линейного алгоритма, подготовить их к самостоятельной разработке последующей программы, исключить чувство страха и неуверенности перед началом программирования.

Слайд 4 (рис. 6, 7) — это **задание 2**. На слайде представлено условие алгоритмической задачи: требуется усовершенствовать алгоритм предыдущего задания таким образом, чтобы результат реализации программы был тот же, а количество используемых команд было бы равно 24. Это задание дает возможность учащимся взглянуть на полученное решение с другой стороны, что будет способствовать развитию вариативного и творческого мышления.

Для самопроверки на слайде размещено правильное решение поставленной задачи, которое скрыто анимацией (рис. 7).

Слайд 5 (рис. 8) — это **задание 3**, в котором учащиеся, используя знания и навыки, полученные при решении предыдущих заданий, должны составить свой алгоритм по условию задачи.

Данное задание состоит из двух частей: 1) нахождение самого числа и 2) составление алгоритма для его изображения. Обе части задания предполагают для решения использование линейного алгоритма.

Слайд 6 (рис. 9) — это **задание 4**, которое является конечным этапом для достижения главного результата общей задачи.

Задание 3

К числу 3 прибавьте 9 и составьте алгоритм для изображения полученного числа.

Рис. 8. Задание 3

X

Задание 4

Приставьте первую цифру конечного числа предыдущего задания в конец этого числа и составьте алгоритм для получившегося...

*Например, было: 23
стало: 232*

Рис. 9. Задание 4

Условие задания состоит из двух частей:

- 1) нахождение заданного числа с помощью линейного алгоритма (найденное число — конечный результат общей задачи);
- 2) программирование изображения полученного числа в среде исполнителя «Стрелочка». В процессе программирования учащиеся демонстрируют все знания и умения, приобретенные в ходе изучения линейного алгоритма.

4. Подведение итогов урока

Выполнив все предложенные задания, учащиеся попадают на **слайд 7**, где весь ход решения задач представлен в виде блок-схемы линейного алгоритма (рис. 10).

Учитель совместно с учениками перечисляет последовательно все действия, выполненные ими на протяжении урока и записанные в форму «Бланк ответов», которые привели к получению результата общей задачи. По мере озвучивания учитель открывает на слайде соответствующие действия-этапы, и учащиеся сверяют и корректируют результаты, отраженные в их бланках (рис. 11).

Как только вся блок-схема открылась, учитель задает вопрос: «Что это? Внутри чего мы находились на протяжении всего урока?» Ученики делают вывод, что они не только решали задания с помощью линейного алгоритма, но и сами находились внутри изучаемого объекта.

5. Домашнее задание

1. Ответьте устно на вопрос: в чем заключается особенность линейного алгоритма?
2. Ответьте на вопрос письменно в тетради: можно ли усовершенствовать программы, составлен-



Рис. 10. Блок-схема линейного алгоритма



Рис. 11. Заполненная блок-схема линейного алгоритма

ные с помощью линейного алгоритма? Если да, то как?

3. Составьте задачу (письменно в тетради), при решении которой был бы использован линейный алгоритм и которая бы состояла из:
 - изображения пути вашего героя от начального положения до конечной точки;
 - описания вашей системы команд исполнителя;
 - программы с количеством команд от 20 до 25.

Литература

1. Батакова Е. Л., Соболев Е. В. Использование интерактивных средств обучения на уроках информатики: учеб. пособие. Киров: Радуга-ПРЕСС, 2013.
2. Кузнецов А. А. Еще раз о школьных стандартах // Информатика и образование. 2012. № 6.



Р. Р. Сулейманов,

Башкирская республиканская гимназия-интернат № 1 имени Рами Гарипова, г. Уфа

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ МАТЕМАТИЧЕСКИХ ЗАДАЧ

Аннотация

В статье рассматривается компьютерное моделирование математических задач. Выделяются учебные действия формирования навыков компьютерного моделирования средствами программирования. Приводятся конкретные примеры, в частности компьютерное моделирование целочисленных треугольников и фигурных чисел. В качестве языка программирования используется язык Pascal, кроме того, алгоритмы описаны на школьном алгоритмическом языке и приведены блок-схемы алгоритмов.

Ключевые слова: компьютерное моделирование, математические задачи, целочисленные треугольники, фигурные числа.

Контактная информация

Сулейманов Ринат Рамилович, канд. пед. наук, доцент, член-корреспондент Академии информатизации образования, учитель информатики высшей категории Башкирской республиканской гимназии-интерната № 1 имени Рами Гарипова, г. Уфа; адрес: 450098, г. Уфа, ул. Росийская, д. 88; телефон: (347) 284-66-09; e-mail: rin-suleimanov@yandex.ru

R. R. Sulejmanov,
Bashkir Republican Gymnasium Boarding
School 1 named after Rami Garipov, Ufa

COMPUTER MODELING OF MATHEMATICAL PROBLEMS

Abstract

The article deals with computer modeling of mathematical problems. The learning activities for developing skills of computer modeling by tools of programming are described. Specific examples, particularly computer modeling of integer numbers of triangles and figured numbers are given. As a programming language Pascal is used, moreover, the algorithms are described on the school algorithmic language and flowcharts of algorithms are given.

Keywords: computer modeling, mathematical problems, integer triangles, figured numbers.

Компьютерное моделирование достаточно широко используется при изучении различных тем курса информатики. Под *моделированием* в данном случае мы подразумеваем специально организованный учителем комплекс задач и действий по их решению, который самостоятельно выполняется учащимися и завершается созданием творческого продукта. *Компьютерное моделирование* в таком понимании — это комплексный обучающий метод, который позволяет индивидуализировать учебный процесс, дает возможность обучающемуся проявить самостоятельность в планировании, организации и контроле своей деятельности, творческий подход при выполнении учебных заданий.

Компьютерное моделирование математических задач предполагает:

- наличие проблемы, требующей интегрированных знаний и исследовательского поиска ее решения;
- практическую, теоретическую, познавательную значимость предполагаемых результатов;
- самостоятельную деятельность ученика;
- структурирование содержательной части с указанием поэтапных результатов;
- подведение итогов, корректировку, выводы.

Учебные действия, которые учащийся выполняет при компьютерном моделировании, тесно взаимосвязаны. Перечислим эти действия в порядке их освоения при формировании навыков компьютерного моделирования учебных задач средствами программирования:

- преобразование условий учебной задачи с целью обнаружения общего отношения, лежащего в основе изучаемой системы теоретических знаний;
- моделирование выделенного отношения в графической и знаковой формах;
- преобразование модели отношения с целью изучения ее общих свойств;
- выделение и решение конкретно-практических вопросов на основе общего способа решения задачи;
- контроль за выполнением предыдущих действий;
- достижение результата (создание продукта);
- оценка достижения результата.

При выполнении перечисленных действий учащийся овладевает определенной системой теоретических знаний и обобщенным способом решения определенного класса конкретных практических задач.

Приведем примеры компьютерного моделирования математических задач.

Целочисленные треугольники

Задача.

Выделить целочисленные треугольники.

Целочисленными треугольниками называются такие треугольники, длины сторон которых выражаются натуральными числами.

По теореме косинусов, для произвольного треугольника с длинами сторон a, b, c и углом α , противоположной стороне c , выполняется равенство:

$$a^2 - 2ab \cos \alpha + b^2 = c^2. \quad (1)$$

Равенство (1) позволяет «забыть» о геометрии: задачу поиска целочисленных треугольников с углом 120° мы свели к задаче решения уравнения (1) в натуральных числах. На поиске решений в натуральных числах уравнения (1) мы сейчас и сосредоточим свои усилия.

Очевидно, что целочисленными треугольниками будут треугольники с углами:

$$\begin{aligned} \alpha &= 90^\circ (\cos 90^\circ = 0), \\ \alpha &= 120^\circ (\cos 120^\circ = -1/2), \\ \alpha &= 60^\circ (\cos 60^\circ = 1/2). \end{aligned}$$

Таблица 1

α	$\cos \alpha$	Теорема косинусов для данного угла α
90°	0	$a^2 + b^2 = c^2$
60°	1/2	$a^2 - ab + b^2 = c^2$
120°	-1/2	$a^2 + ab + b^2 = c^2$

Решим уравнение (1) в натуральных числах для конкретных треугольников.

Целочисленные прямоугольные треугольники

По теореме Пифагора (и теореме косинусов при $\alpha = 90^\circ$), стороны прямоугольного треугольника связаны формулой $a^2 + b^2 = c^2$, где a, b — катеты, c — гипотенуза (рис. 1). Выполним поиск целочисленных прямоугольных треугольников, длины сторон которых не больше 100.

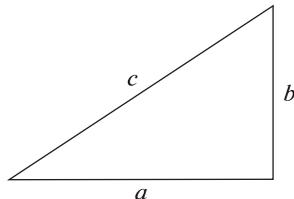
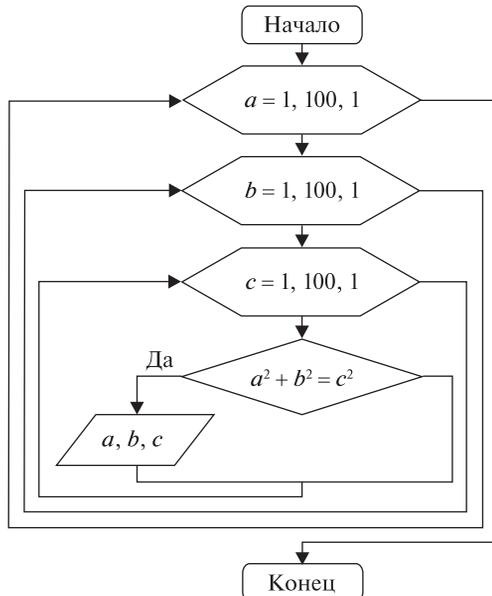


Рис. 1

Блок-схема 1.



Алгоритм 1.

```

алг 1 (нат А, В, С)
арг А, В, С
рез А, В, С
нач
нц для А от 1 до 100
нц для В от 1 до 100
нц для С от 1 до 100
если А2+В2=С2
то вывод А, В, С
все
кц
кц
кц
кон
    
```

Программа 1.

```

program pr1;
uses crt;
var a, b, c: word; {задание типа переменных a, b, c}
begin
for a:=1 to 100 do {перебор всех возможных
(целых) значений катета a}
for b:=1 to 100 do {перебор всех возможных
(целых) значений катета b}
for c:=1 to 100 do {перебор всех возможных
(целых) значений гипотенузы c}
if a*a+b*b = c*c
then writeln('a=', a, ' b=', b, ' c=', c);
{проверка выполнения теоремы Пифагора для треугольника со сторонами a, b, c}
end.
    
```

Дополнительные сведения.

Длины сторон целочисленных прямоугольных треугольников представляют собой «пифагоровы» тройки чисел (a, b, c) , значения которых можно вычислить по формулам:

$$\begin{aligned} a &= l(m^2 - n^2), \\ b &= 2lmn, \\ c &= l(m^2 + n^2), \end{aligned} \quad (2)$$

где l, m, n — произвольные натуральные числа ($m > n$), а выражения для вычисления a и b можно менять местами.

Тогда длины сторон целочисленных прямоугольных треугольников, применяя соотношения (2), можно вычислять, применяя алгоритм и программу 2. Диапазон изменения значений переменных l, m, n — от 1 до 10.

Алгоритм 2.

```

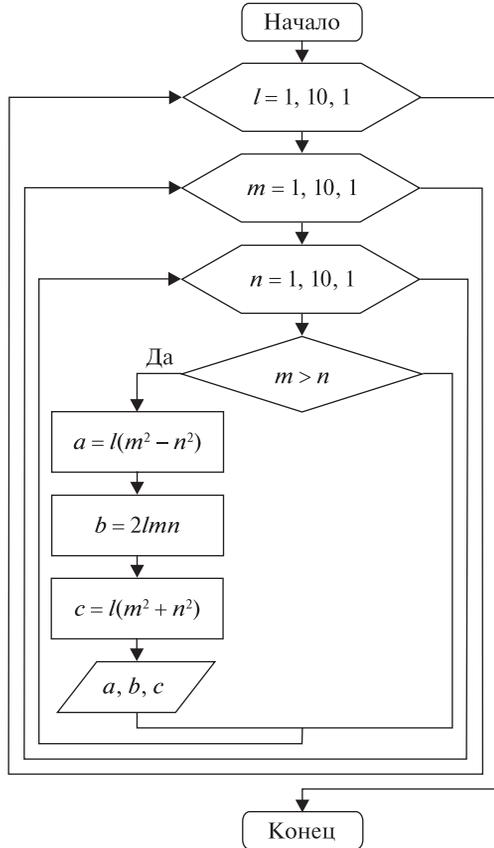
алг 2 (нат l, m, n, А, В, С)
арг l, m, n
рез А, В, С
нач
нц для l от 1 до 10
нц для m от 1 до 10
нц для n от 1 до 10
если m > n
то А:=l(m2-n2)
В:=2lmn
С:=l(m2+n2)
вывод А, В, С
все
кц
кц
кц
кон
    
```

Программа 2.

```

program pr 2;
uses crt;
var l, m, n, a, b, c: word; {задание типа
переменных l, m, n, a, b, c}
    
```

Блок-схема 2.



```

begin
  for l:=1 to 10 do      {перебор возможных (целых)
                        значений переменной l}
    for m:=1 to 10 do   {перебор возможных (целых)
                        значений переменной m}
      for n:=1 to 10 do {перебор возможных (целых)
                        значений переменной n}
        begin
          if m>n then
            {проверка условия m>n, если оно выполняется, то
             вычисляются значения переменных a, b, c и вывод
             полученной целочисленной тройки чисел на экран}
            begin
              a:=l*(m*m-n*n);
              b:=2*l*m*n;
              c:=l*(m*m+n*n);
              writeln(' a=', a, ' b=', b, ' c=', c);
            end;
          end;
        end;
      end;
    end;
  end.
  
```

Целочисленные треугольники с углом 120 градусов

В целочисленных треугольниках со сторонами *a*, *b*, *c*, содержащих углы 60° или 120°, договоримся обозначать буквой *c* сторону, противолежащую углу в 60° или 120° (рис. 2).

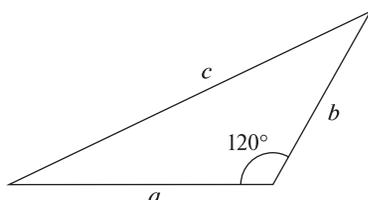


Рис. 2

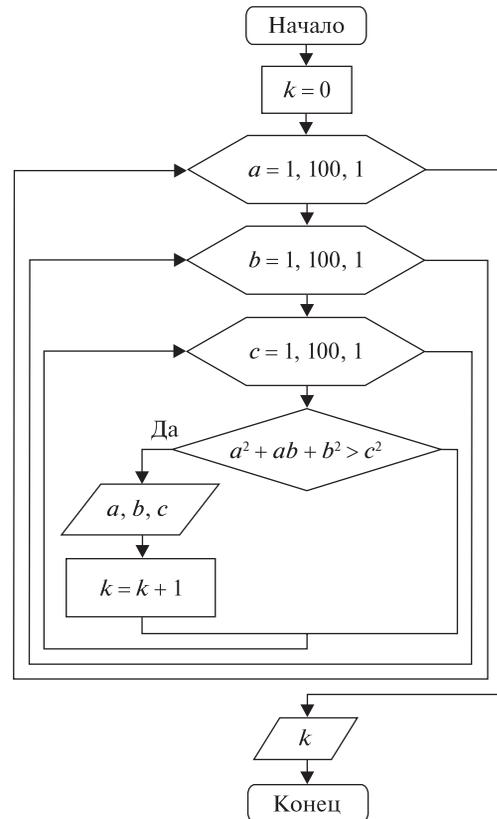
Если треугольник имеет угол 120°, то его стороны *a*, *b*, *c* связаны равенством:

$$a^2 + ab + b^2 = c^2. \tag{3}$$

И наоборот, если для треугольника выполняется соотношение (3), то в этом случае косинус угла α , противолежащего стороне *c*, согласно (3) и теореме косинусов, равняется $-1/2$. Это возможно только в случае, если $\alpha = 120^\circ$.

Алгоритм 3 и программа 3 позволяют определить наборы *a*, *b*, *c* длин сторон целочисленного треугольника с углом 120° и их количество. Диапазон изменения длин сторон треугольника — от 1 до 10.

Блок-схема 3.



Алгоритм 3.

```

алг 42 (нат А, В, С, К)
арг А, В, С
рез А, В, С, К
нач
К:=0
нц для А от 1 до 100
нц для В от 1 до 100
нц для С от 1 до 100
если А2+АВ+В2>С2
то вывод А, В, С
К:=К+1
все
кц
кц
кц
вывод К
кон
  
```

Программа 3.

```

program pr3;
uses crt;
var a, b, c, k: word; {задание типа переменных
a, b, c, k}
  
```

```

begin
k:= 0; {задание начального значения счетчика k}
for a:=1 to 100 do {перебор всех возможных
                   (целых) значений стороны a}
  for b:=1 to 100 do {перебор всех возможных
                     (целых) значений стороны b}
    for c:=1 to 100 do {перебор всех возможных
                       (целых) значений стороны c}
      if a*a+a*b+b*b=c*c then
        begin
          writeln ('a=',a,' ','b=',b,' ','c=',c);
          {проверка выполнения теоремы косинусов для треу-
            гольника со сторонами a, b, c и углом α = 120°
            и вывод полученных троек чисел на экран}
          k:= k+1;          {подсчет количества
                           целочисленных троек}
        end;
      writeln('k=', k);   {вывод количества
                           целочисленных троек на экран}
    end.
end.

```

Целочисленные треугольники с углом 60 градусов

Длины сторон a, b, c треугольников с углом 60° связаны соотношением:

$$a^2 - ab + b^2 = c^2,$$

которое получается из теоремы косинусов.

Если в программе 3 изменить условие $a*a+a*b+b*b=c*c$ на $a*a-a*b+b*b=c*c$, то программа будет подсчитывать длины сторон целочисленного треугольника с углом 60° и количество пифагоровых троек для данного типа треугольников.

Дополнительные сведения.

Теорема. Пусть числа a, b, c связаны соотношением $a^2 + ab + b^2 = c^2$. Тогда найдутся натуральные взаимно простые числа n и m ($n > m$), такие, что $n - m$ не делится на 3 и

$$\begin{aligned} a &= n^2 - m^2, \\ b &= m^2 + 2mn, \\ c &= m^2 + mn + n^2 \end{aligned} \quad (4)$$

Доказательство данной теоремы можно предложить учащимся выполнить самостоятельно.

Составим программу вычисления чисел a, b, c — длин сторон целочисленного треугольника с углом 60° и $b > a$, когда можно подобрать взаимно простые натуральные числа m и n , для которых справедливы соотношения (4). При составлении алгоритма и программы можно воспользоваться алгоритмом Евклида для нахождения взаимно простых чисел (это числа, которые не имеют общих делителей, кроме единицы), а также учесть, что $n > m$ и выполняется условие $(n-m) \bmod 3 \neq 0$.

Алгоритм 4 и программа 4 реализуют приведенную выше теорему для n и m , не больших 10.

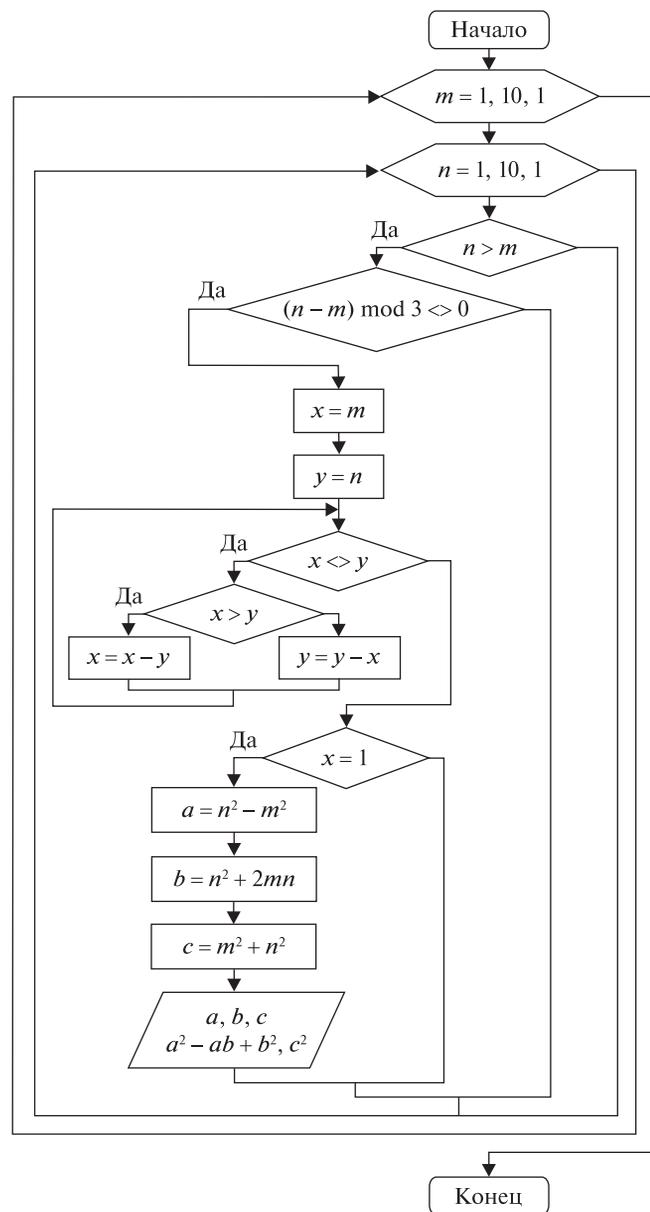
Алгоритм 4.

```

алг 4 (нат М, N, A, B, C, X, Y)
арг М, N
рез А, В, С
нач
нц для М от 1 до 10
нц для N от 1 до 10
если N>M
то если (N-M) mod 3 <> 0
то X:=M, Y:=N
нц пока X<>Y
если X>Y

```

Блок-схема 4.



```

то X:=X-Y
иначе Y:=Y-X
кц
если X=1
то A:=N2-M2, B:=N2+2MN, C:=M2+ N2
вывод А, В, С, A2-AB+B2, C2
все
все
кц
кц
кон

```

Программа 4.

```

program pr4;
uses crt;
var m, n, x, y, a, b, c: longint;
begin
for m:=1 to 10 do
for n:=1 to 10 do
begin
if n>m then
begin

```

```

if (n-m) mod 3 <>0 then
begin
  x:=m;
  y:=n;
  while x<>y do
  if x>y
  then x:=x-y
  else y:=y-x;
  if x=1 then
  begin
    a:=n*n-m*m;
    b:=n*n+2*m*n;
    c:=m*m+m*n+n*n;
    writeln('a=', a, ' b=', b, ' c=', c, '
'a*a-a*b+b*b, ' ', c*c);
  end;
end;
end;
end;
end.

```

Фигурные числа

Решим следующую задачу:

Даны K шаров одинакового радиуса. Построить пирамиду максимальной высоты. Сколько шаров уйдет на построение данной пирамиды? Сколько этажей будет содержать данная пирамида?

Прежде чем решить предложенную задачу, познакомимся с фигурными числами, знания о которых помогут решить нашу задачу.

Еще задолго до нашей эры ученые, комбинируя натуральные числа, составляли из них затейливые ряды, придавая элементам этих рядов то или иное геометрическое истолкование.

Так, например, в V—IV веках до н. э. возникли представления о рядах так называемых фигурных чисел.

Рассмотрим сначала такой ряд, в котором разность между каждым последующим и предыдущим членами равна одному и тому же натуральному числу (арифметическая прогрессия), например:

1, 2, 3, 4, 5, ... (разность $d = 1$),
 1, 3, 5, 7, 9, ... (разность $d = 2$),
 1, 4, 7, 10, 13, ... (разность $d = 3$),

или в общем виде:

1, $1 + d$, $1 + 2d$, $1 + 3d$, $1 + 4d$, ...

У каждого элемента ряда есть свое место. Чтобы получить n -й элемент ряда (назовем его a_n), надо к первому элементу ряда прибавить произведение разности ряда на число, которое на единицу меньше номера места, занимаемого этим элементом a_n :

$a_n = 1 + d(n - 1)$.

Элементы каждого из таких рядов называются **линейными фигурными числами, или фигурными числами первого порядка**. Из рядов с линейными фигурными числами образуем последовательные суммы этих чисел: первую сумму — из одного первого элемента ряда линейных фигурных чисел, вторую сумму — складывая первые два элемента того же ряда, третью сумму — складывая первые три элемента, и т. д., n -ю сумму — складывая первые n элементов.

Плоские фигурные числа

Так, первый ряд линейных фигурных чисел 1, 2, 3, 4, 5, ... производит следующий новый ряд, ряд сумм:

$S = 1$,
 $S = 1 + 2 = 3$,
 $S = 1 + 2 + 3 = 6$,
 $S = 1 + 2 + 3 + 4 = 10$, ...,

или

1, 3, 6, 10, 15, ...

Назвали эти числа *треугольными*.

Следующий ряд производит *квадратные* числа: 1, 4, 9, 16, 25, ...

Пятиугольные числа: 1, 5, 12, 22, 35, ...

Аналогично можно образовать *шестиугольные*, *семиугольные* и т. д. числа.

Все эти многоугольные числа называются **плоскими фигурными числами, или фигурными числами второго порядка**.

Геометрические «имена», которые получили эти числа, объясняются возможностью дать им наглядное истолкование. Построим равносторонний треугольник, квадрат, правильные пятиугольник, шестиугольник и т. д. со сторонами, равными 1. Затем, отправляясь в каждой фигуре от одной из вершин, удлиним все стороны в 2, 3, 4, ... раз, т. е., как говорят математики, построим многоугольники перспективно-подобные данным (рис. 3). Во всех вершинах получившихся фигур и на их сторонах на расстояниях, равных 1, поместим кружочки. Подсчет кружочков, расположенных в каждом треугольнике, приводит к ряду треугольных чисел: 1, 3, 6, 10, 15, ... Подсчет кружочков, расположенных в каждом квадрате, дает последовательность квадратных чисел: 1, 4, 9, 16, 25, ...

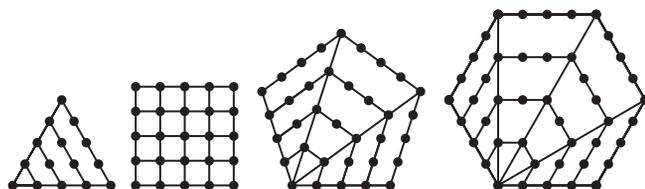


Рис. 3. Фигурные числа

Аналогичные подсчеты кружочков в каждом пятиугольнике, шестиугольнике и т. д. приводят соответственно к последовательностям чисел пятиугольных, шестиугольных и т. д.

Дадим сводку всех плоских фигурных чисел (табл. 2).

Отсюда при $d = 1$ получаем формулу для S_n первой строки таблицы, при $d = 2$ получаем формулу для S_n второй строки таблицы и т. д.

Между натуральными числами и плоскими фигурными числами и между самими плоскими фигурными числами существует много любопытных зависимостей. Проверим справедливость утверждений:

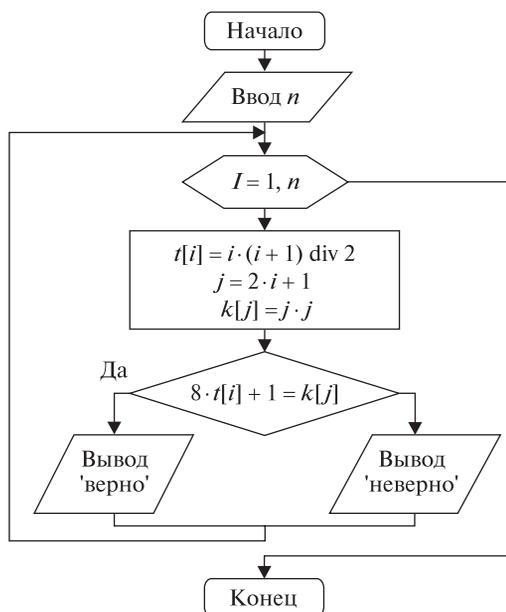
- Формула Диофанта устанавливает связь между треугольными и квадратными числами: $8T + 1 = K$.
- Никакое треугольное число не может оканчиваться цифрами 2, 4, 7 и 9.
- Всякое шестиугольное число есть треугольное с нечетным номером.

Программы 5—7 позволяют экспериментально убедиться в справедливости приведенных выше утверждений. Блок-схема и алгоритм проверки первого утверждения («Номер квадратного числа $2n + 1$, удовлетворяющий условию $8T + 1 = K$, соответствует номеру n треугольного числа») приведены ниже.

Таблица 2

d	Фигура	Числа					Общий член S_n
		S_1	S_2	S_3	S_4	S_5	
1	Треугольник	1	3	6	10	15	$n(n+1)/2$
2	Квадрат	1	4	9	16	25	n^2
3	Пятиугольник	1	5	12	22	35	$n(3n-1)/2$
4	Шестиугольник	1	6	15	28	45	$n(2n-1)$
...		.					
d		1	$2+d$	$3+3d$	$4+6d$	$5+10d$	$n(dn-(d-2))/2$

Блок-схема 5.



Алгоритм 5.

```

алг Фигурные числа;
арг нат n, i, j (n - порядковый номер фигурного
числа);
рез лит 'верно', 'неверно'
нач нат таб t[1..n], k[1..n]
ввод n;
нц для i от 1 до
t[i]:=i*(i+1) div 2;
j:=2*i+1;
k[j]:=j*j;
если 8*t[i]+1=k[j]
то вывод('верно')
иначе вывод('неверно')
все;
кц
кон
  
```

Программа 5.

```

program pr5;
uses crt;
var
n, i, j: word;
t, k: array[1..100] of word;
begin
readln(n);
for i:=1 to n do
begin
t[i]:=i*(i+1) div 2;
j:=2*i+1;
k[j]:=j*j;
  
```

```

if 8*t[i]+1=k[j]
then writeln('verno')
else writeln('neverno')
end;
end.
  
```

Программа 6.

```

program pr6;
uses crt;
var
n, i: word;
t: array[1..100] of word;
begin
readln(n);
for i:=1 to n do
begin
t[i]:=i*(i+1) div 2;
if (t[i] mod 10=2) or
(t[i] mod 10=4) or
(t[i] mod 10=7) or
(t[i] mod 10=9)
then writeln('neverno')
else writeln('verno')
end;
end.
  
```

Программа 7.

```

program pr7;
uses crt;
var
n, i, j: word;
s, t: array[1..100] of word;
begin
readln(n);
for i:=1 to n do
begin
s[i]:=i*(2*i-1);
j:=2*i-1;
t[j]:=j*(j+1) div 2;
if s[i]=t[j]
then writeln('verno')
else writeln('neverno')
end;
end.
  
```

Пьер Ферма обнаружил, например, что:

- всякое натуральное число есть треугольное или сумма двух или трех треугольных чисел;
- всякое натуральное число есть или квадрат, или сумма двух, трех или четырех квадратных чисел; всякое натуральное число есть или пятиугольное, или сумма двух, трех, четырех или пяти пятиугольных чисел;
- вообще всякое натуральное число может быть представлено в виде суммы не более чем k k -угольных чисел.

Таблица 3

d	Числа					Общий член V_n
	V_1	V_2	V_3	V_4	V_5	
1	1	4	10	20	35	$n(n+1)(n+2)/6$
2	1	5	14	30	55	$n(n+1)(2n+1)/6$
3	1	6	18	40	75	$n^2(n+1)/2$
4	1	7	22	50	95	$n(n+1)(4n-1)/6$
...						
d	1	$3+d$	$6+4d$	$10+10d$	$15+20d$	$n(n+1)(dn-(d-3))/6$

Для отдельных частных случаев эту теорему доказал петербургский математик Л. Эйлер, а общее доказательство дал в 1815 г. французский математик О.-Л. Коши.

Пространственные фигурные числа

Составляя последовательные суммы из плоских фигурных чисел $V_1 = S_1$, $V_2 = S_1 + S_2$, $V_3 = S_1 + S_2 + S_3$, $V_4 = S_1 + S_2 + S_3 + S_4$ и т. д., получим **пространственные фигурные числа, или фигурные числа третьего порядка**.

Так, ряд треугольных чисел 1, 3, 6, 10, 15, ... приводит следующий ряд чисел третьего порядка: 1, 4, 10, 20, 35, ...

Эти числа называют еще **пирамидальными**, так как для их геометрического представления выкладывают пирамиды из шаров одинакового диаметра.

Подложим под шар три шара. Получим пирамиду из четырех шаров. Подложим снизу еще шесть шаров, получим пирамиду из 10 шаров (рис. 4).

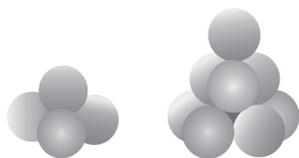


Рис. 4

Дадим сводку всех фигурных чисел третьего порядка (табл. 3).

Теперь вернемся к решению задачи, предложенной в начале раздела: «Даны K шаров одинакового радиуса. Построить пирамиду максимальной высоты. Сколько шаров уйдут на построение данной пирамиды? Сколько этажей будет содержать данная пирамида?»

Обозначения:

K — общее количество шаров,

n — количество этажей,

d — разность арифметической прогрессии, лежащей на основе плоских фигурных чисел из которых строится пирамида,

V — количество шаров, необходимых на построение пирамиды.

Из таблицы 2 видим, что максимальное число этажей будет при $d = 1$.

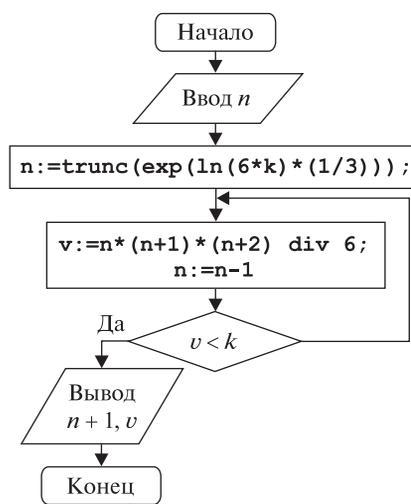
При составлении программы воспользуемся следующими соотношениями:

$$n(n+1)(n+2)/6 = V \leq K,$$

$$a^n = \exp(\ln(a) + n).$$

Ниже приведены блок-схема, алгоритм и программа решения задачи.

Блок-схема 6.



Алгоритм 6.

```

алг Фигурные числа pr8;
арг нат k (кол-во шаров);
рез нат n, v (n — кол-во этажей, v — кол-во шаров,
необходимых для пирамиды)
нач ввод(k);
n:=trunc(exp(ln(6*k)*(1/3)));
нц повторять
v:=n*(n+1)*(n+2) div 6;
n:=n-1;
кц до v<k;
вывод('n=', n+1, ' ', 'v=', v);
кон.
  
```

Программа 8.

```

program pr8;
uses crt;
var k, n, v: integer;
begin
  readln(k);
  n:=trunc(exp(ln(6*k)*(1/3)));
  repeat
    v:=n*(n+1)*(n+2) div 6;
    n:=n-1;
  until v<k;
  writeln('n=', n+1, ' ', 'v=', v);
end.
  
```

Литература

1. Сулейманов Р. Р. Компьютерное моделирование математических задач: методическое пособие. М.: БИНОМ. Лаборатория знаний, 2013.
2. Сулейманов Р. Р. Компьютерное моделирование математических задач: учебное пособие. М.: БИНОМ. Лаборатория знаний, 2012.

С. М. Окулов, О. А. Пестов,

Вятский государственный гуманитарный университет, г. Киров

ДВОИЧНОЕ ДЕРЕВО ПОИСКА

Аннотация

Рассмотрена классическая структура данных информатики — двоичное дерево поиска. Материал предназначен для теоретического занятия в курсе углубленного изучения информатики.

Ключевые слова: структуры данных, дерево поиска, обход дерева.

Контактная информация

Окулов Станислав Михайлович, доктор пед. наук, канд. тех. наук, профессор кафедры прикладной математики и информатики Вятского государственного гуманитарного университета (ВятГУ), г. Киров; *адрес:* 610002, г. Киров, ул. Красноармейская, д. 26; *телефон:* (8332) 67-53-01; *e-mail:* okulov@vshu.kirov.ru

S. M. Okulov, O. A. Pestov,
Vyatka State University of Humanities,
Kirov

BINARY SEARCH TREE

Abstract

The authors describe a classical data structure called binary search tree. It can be studied within a theoretical practice during the course of advanced informatics.

Keywords: data structures, search tree, tree traversal.

Двоичное дерево поиска можно назвать классической структурой данных в информатике. Дело в том, что организация данных в таком виде позволяет решать задачу их поиска весьма эффективно. Умение использовать эту структуру, применять при решении задач — необходимый компонент подготовки школьника при углубленном изучении предмета.

Традиционно эта тема излагается с использованием ссылочного типа данных (указателей) так, как это сделано в работе [2]. В данном материале дерево описывается только с использованием массивов, что, безусловно, позволяет с меньшими затратами на отладку использовать его при решении задач.

Не давая формального определения двоичного дерева поиска (ДДП), поясним эту структуру на базе простых примеров.

На рисунке 1 изображены три различных ДДП. Элемент структуры состоит из ключа, информационной части и не более двух указателей (ссылок) на другие элементы. Один элемент выделен, на него нет ссылок, и он называется *корнем дерева*. Информационная часть, она может быть любой, на рисунке 1 не приводится. Обозначены лишь ключи, являющиеся в данном случае целыми числами. *Ссылки элемента*, их называют левой и правой, указывают на элементы, которые в свою очередь являются корнями соответствующих поддеревьев.

Итак, дерево двоичное, потому что элемент имеет не более двух связей. Поиск-дерево обладает тем свойством, что для любого его элемента ключи в левом поддереве меньше, а ключи в правом — больше, чем значение ключа данного элемента.

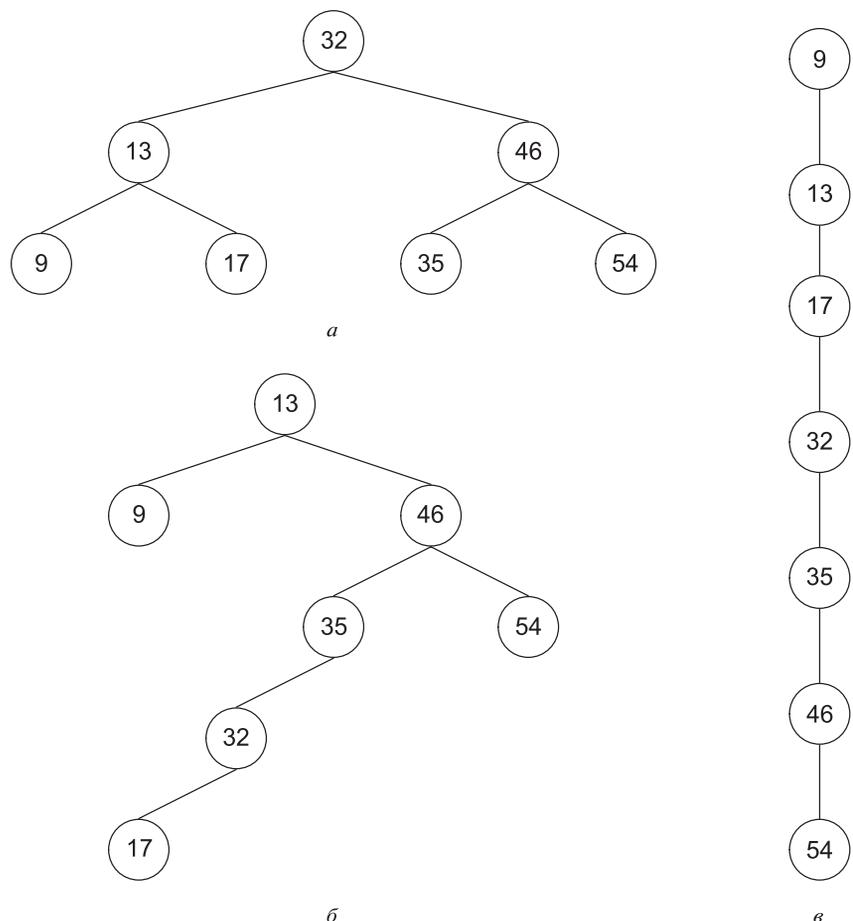


Рис. 1. Три различных двоичных дерева поиска, построенных на одних и тех же данных

Обратимся к рисунку 1. В первом случае (а) данные вводились в такой очередности: 32, 13, 46, 9, 17, 35, 54; во втором (б) — 13, 9, 46, 35, 32, 17, 54; в третьем (в) — 9, 13, 17, 32, 35, 46, 54. Одни и те же данные — и три различных дерева. Первое дерево называют *сбалансированным*, количество элементов в правом и левом поддеревьях для каждого элемента, являющегося корнем, совпадают. Во втором случае этой идеальности уже нет, а в третьем получился вообще «вырожденный» случай с точки зрения понятия «дерево» — оно превратилось в линейную структуру.

В чем особенность сбалансированного дерева? Высота дерева, а это количество отрезков на пути от корня до самых нижних элементов, определяется как $\log_2 n$, где n — количество элементов в дереве. И достаточно очевидно, что поиск элемента дерева с заданным значением ключа будет пропорционален именно этому значению. В последнем случае, на рисунке 1, в, поиск уже осуществляется за время $O(n)$ и ничем не отличается от поиска в линейном массиве при условии, что он всегда начинается с первого элемента и осуществляется последовательно путем просмотра, следующего за ним, ибо до конкретного элемента мы можем «добраться», только пройдя все предыдущие.

Таким образом, для работы с ДДП необходимо не только уметь вставлять, удалять и просматривать элементы, но и, при возможности, сохранять дерево в идеальном или близком к нему состоянии.

ДДП в памяти компьютера описывается тремя одномерными массивами: l, r и x . Ячейки массивов с номерами i определяют один элемент дерева. И так, в $x[i]$ хранится значение ключа, в $l[i]$ указывается адрес левого, а в $r[i]$ — адрес правого элемента. В переменной $root$ — адрес корня дерева, в переменной n — количество элементов, когда-либо добавленных в дерево. Если не было операций удаления, то это просто количество элементов в дереве.

```
const MAXN= ;
var
  l, r, x: array[1..MAXN] of integer;
  root, n: integer;
```

Для дерева на рисунке 1, а его описание в памяти показано в таблице 1.

Логика поиска элемента дерева с заданным значением ключа достаточно прозрачна. Приведем ее для полноты описания.

Таблица 1

i	1	2	3	4	5	6	7
x	32	13	46	9	17	35	54
l	2	4	6	0	0	0	0
r	3	5	7	0	0	0	0

```
function find(key: integer): integer;
  var v: integer;
begin
  v:=1;
  while (v<>0) and (x[v]<>key) do
    if key<x[v] then v:=l[v] else v:=r[v];
  find:=v;
end;
```

Определим процедуру выделения памяти в массивах для хранения данных об элементе дерева.

```
procedure create(key: integer);
begin
  n:=n+1;
  l[n]:=0;
  r[n]:=0;
  x[n]:=key;
end;
```

И рассмотрим логику вставки элемента в дерево, предполагая, что в дереве нет элемента с заданным значением ключа.

```
procedure insert(key: integer);
  var v: integer;
begin
  create(key);
  if root=0 then root:=n; {Элемент является корнем, он первый}
  v:=root;
  while v<>n do {Поиск места для вставки элемента}
    if key<x[v] then
      begin {Идем в левое поддерево}
        if l[v]=0 then l[v]:=n; {Место найдено}
        v:=l[v] {Переход к следующему элементу}
      end
    else
      begin {Идем в правое поддерево}
        if r[v]=0 then r[v]:=n; {Место найдено}
        v:=r[v]; {Переход к следующему элементу}
      end;
  end;
```

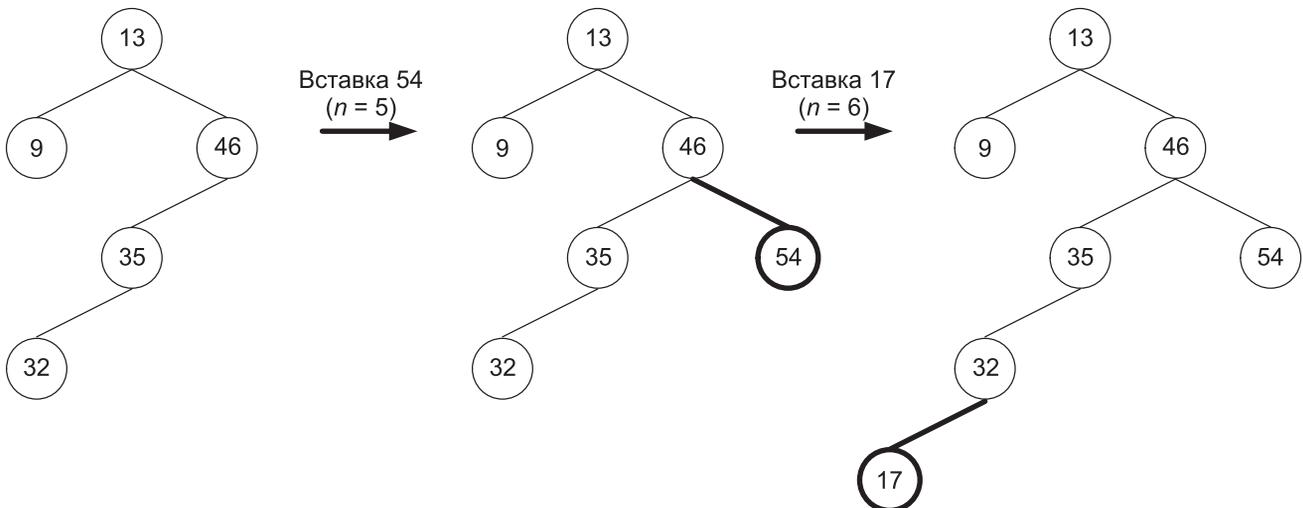


Рис. 2. Вставка элементов в дерево

Пусть есть дерево, полученное в результате при последовательном вводе элементов с ключами 13, 9, 46, 35, 32 (рис. 2). Необходимо вставить элемент с ключом 54.

Элемент создается с помощью процедуры *create*, а затем по логике *insert* осуществляется поиск его места в дереве. Начинаем с корня и идем в правое поддерево ($54 > 13$). Так как 54 больше и 46, то следующим значением v будет правая ссылка этого элемента, а она равна нулю. Место найдено. Новым значением правой ссылки у элемента с ключом 46 станет адрес нового элемента. Аналогично при вставке элемента с ключом 17. Необходимо только два раза перейти по левым ссылкам от элемента с ключом 46.

Можно ли упростить логику? Оказывается, да, путем введения фиктивного элемента с номером 1: $x[1]$ равно значению ключа, оно заведомо меньше возможных значений, например, -1000 ; $l[1]$ равно нулю; $r[1]$ указывает на корень дерева (рис. 3). Для создания достаточно вызвать *create*(-1000).

В этом случае можно отказаться от переменной *root* как указателя на корень дерева и, соответственно, в логике вставки — от анализа, в корне мы или нет.

```

procedure insert(key: integer);
  var v, p: integer;
begin
  p:=1;
  v:=r[p];
  while v<>0 do
    begin
      p:=v;
      if key<x[v] then v:=l[v] else v:=r[v];
    end;
  create(key);
  if key<x[p] then l[p]:=n else r[p]:=n;
end;

```

Дерево — рекурсивная структура. Вывод элементов дерева — простая рекурсивная процедура.

```

procedure print(v: integer);
begin
  if v=0 then exit;
  print(l[v]);
  write(x[v], ' ');
  print(r[v]);
end;

```

Для дерева на рисунке 1, а последовательность обхода элементов показана на рисунке 4.

Ключи будут выведены в следующей очередности: 9, 13, 17, 32, 35, 46, 54. Они выводятся в одной строке, что, конечно, не наглядно, ибо структура дерева не просматривается. Незначительная модификация процедуры дает тот же результат (рис. 5), но структура очевидна.

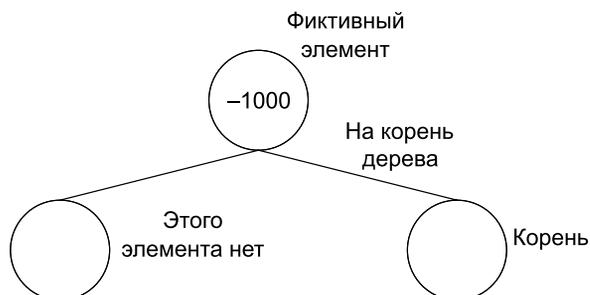


Рис. 3. Фиктивный элемент

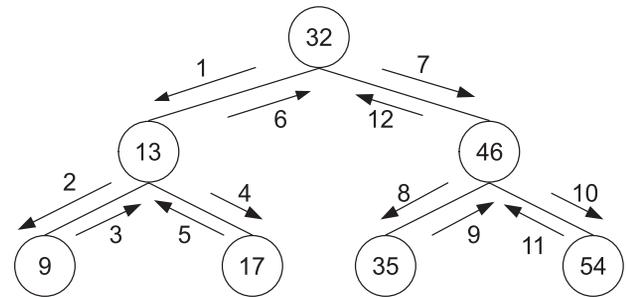


Рис. 4. Обход дерева

```

          9
        13 17
       32  35
      46  54

```

Рис. 5. Вывод элементов дерева

```

procedure print(v, h: integer);
  var i: integer;
begin
  if v=0 then exit;
  print(l[v], h+6);
  for i:=1 to h do write(' ');
  writeln(x[v]);
  print(r[v], h+6);
end;

```

Прежде чем перейти к рассмотрению методов удаления элементов из дерева, покажем, как можно организовать проверочную работу уже с приведенной логикой. Естественно, это не единственный вариант, но один из самых простых. Предположим, что необходимо создать ДДП из 20 элементов с ключами из интервала от 1 до 100, а затем удалить все элементы, имеющие ключи в интервале от 40 до 60. Следующая логика решает эту задачу.

```

var i: integer;
begin
  randomize;
  create(-1000);
  for i:=1 to 20 do insert(random(100));
  print(r[1], 0);
  for i:=1 to 20 do delete(40+i);
  print(r[1], 0);
end.

```

Перейдем к рассмотрению методов удаления элементов ДДП. При удалении необходимо не только найти элемент, но и сохранить структуру дерева — оно должно оставаться деревом поиска.

Пусть мы нашли удаляемый элемент. На него указывает значение переменной v . Для сохранения структуры дерева необходимо поставить на это место или самый большой элемент из левого поддерева (рис. 6, а), или самый наименьший — из правого поддерева (рис. 6, б). Удаляемые элементы на рисунке 6 отмечены стрелками. Выберем первый вариант.

Процедура удаления в этом случае выглядит так:

```

procedure delete(key: integer);
  var v, u, p, z: integer;
begin
  p:=1;
  v:=r[p];

```

```

while (v<>0) and (x[v]<>key) do
  begin
    p:=v;
    if key<x[v] then v:=l[v] else v:=r[v];
  end;
if v=0 then exit;
{Элемент найден значение переменной v. В переменной p хранится адрес предыдущего элемента дерева. В левом поддереве находим максимальный элемент и ставим его на место v}
if l[v]=0 then u:=r[v] {Левого поддерева нет. Запоминаем адрес правого поддерева (рис. 7,а)}
else if r[l[v]]=0 then
  begin {Правого поддерева у левого сына удаляемого элемента нет (рис. 7,б)}
    u:=l[v];
    r[u]:=r[v];
  end
else
  begin {Поиск наибольшего элемента}
    u:=r[l[v]];
    z:=l[v];
    while r[u]<>0 do
      begin {Идем по правым поддеревьям}
        z:=u;
        u:=r[u];
      end;
    {Элемент найден (рис. 7,в)}
    r[z]:=l[u]; {1}
    l[u]:=l[v]; {2}
    r[u]:=r[v]; {3}
  end;
if l[p]=v then l[p]:=u else r[p]:=u; {4}
end;

```

Несмотря на кажущуюся простоту логики, как правило, она воспринимается достаточно трудно. Поясним ее на рисунке 7. Значением переменной v является адрес удаляемого элемента, значением p — адрес предшествующего элемента, а значением переменной u — адрес того элемента, который «ставится» в дереве на место v . Случай на рисунке 7, а самый простой. У удаляемого элемента нет левого поддерева. Необходимо запомнить адрес правого поддерева в элементе p , и все. Вариант логики на рисунке 7, б отражает ситуацию, когда у левого сына удаляемого элемента нет правого поддерева, — искать максимальный элемент негде, им является сам левый сын. Адрес левого сына становится адресом в p , а правое поддерево v становится правым поддеревом u . Самый общий случай, а именно: есть правое поддерево у левого сына v , показан на рисунке 7, в. Конструкция while нам обеспечивает нахождение «самого правого» элемента в поддереве (наибольшего). Адрес предшествующего ему элемента фиксируется в z . Остается выполнить действия, отмеченные, как и в предыдущих случаях, на рисунке 7, в пунктиром. Каждая пунктирная линия отмечена цифрой в фигурных скобках. Аналогично в логике: оператор, выполняющий это действие, отмечен такой же меткой. Во-первых, адрес левого поддерева у найденного элемента (u) становится адресом правого поддерева у предшествующего элемента (z). Во-вторых, левым поддеревом найденного элемента (u) становится левое поддерево удаляемого элемента (v), и, в-третьих, правым поддеревом u становится правое поддерево v . После этого, как и в предыдущих случаях,

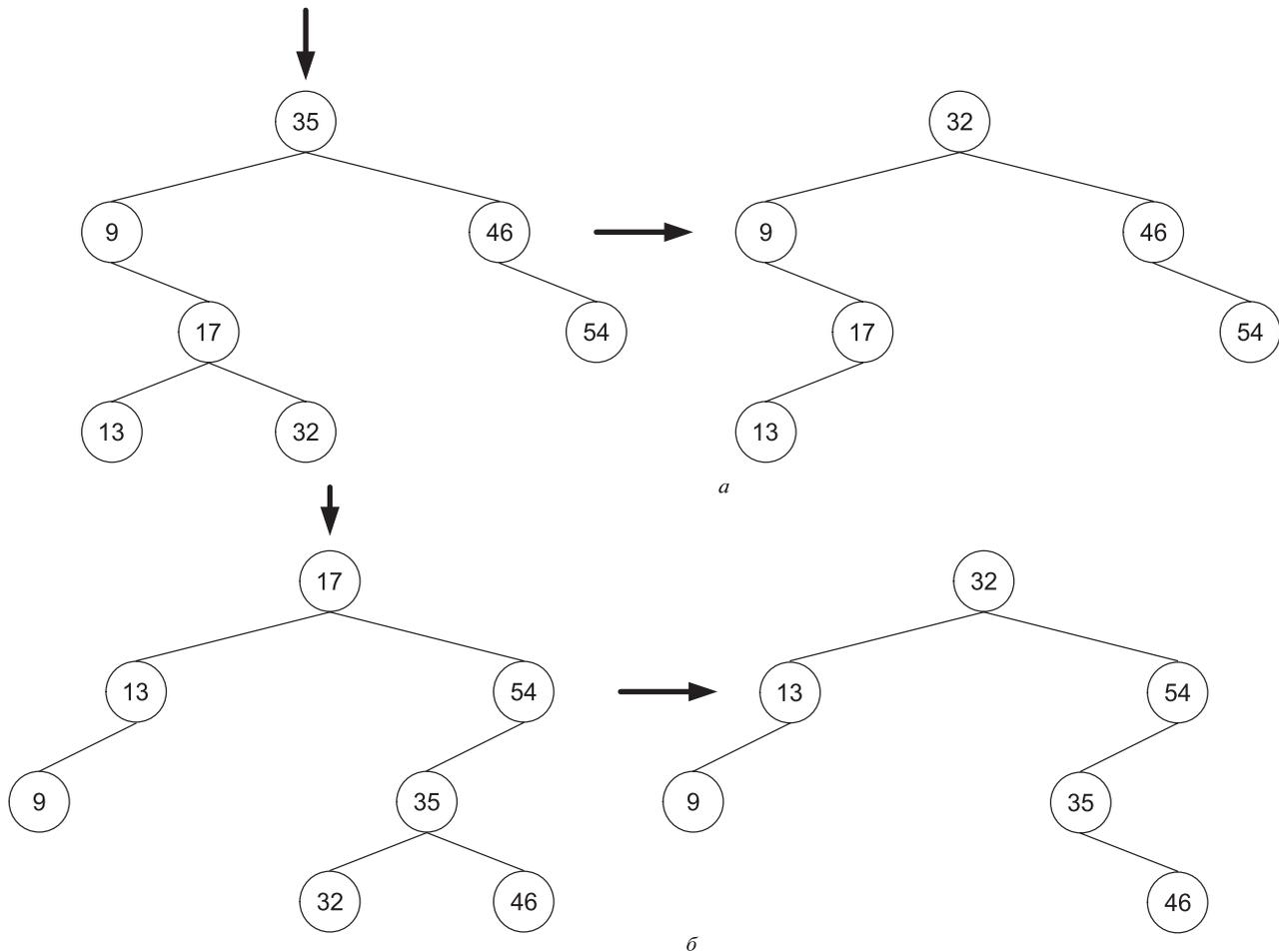


Рис. 6. Принцип удаления элемента из ДДП

корректируется адрес u элемента, предшествующего v , а это элемент p .

Примечание. Следует обратить внимание на то, что элемент удаляется из дерева, но в памяти, а именно в массивах x, l, r он остается как некий «мусор».

В некоторых случаях, особенно когда позволяет память, можно обойтись без явного удаления элемента из дерева. До этого мы в своем обсуждении предполагали, что ключи не повторяются, каждый из них уникален. Если снять это ограничение и допустить существование повторений, то тогда можно ввести четвертый массив (k) счетчиков числа повторений элементов с заданным значением ключа. В этом случае для удаления элемента из дерева все остальное (вставка, просмотр) остается неизменным по своей сути, достаточно уменьшить соответствующее значение счетчика.

Удаление здесь очевидно.

```

procedure delete(key: integer);
var v: integer;
begin
v:=find(key);
if (v<>0) and (k[v]>0) then
k[v]:=k[v]-1;
end;
    
```

Задача [1].

Напишите программу, реализующую структуру данных, позволяющую добавлять и удалять элементы,

а также находить k -й максимум. Признаком первого типа запросов является 1, второго — -1, третьего — 0.

Примеры запросов:

'+1 4' — добавить данные с ключом 4;

'-1 3' — удалить данные с ключом 3;

'0 7' — найти данные, имеющие 7-е максимальное значение ключа.

Гарантируется, что в процессе работы в структуре не требуется хранить элементы с равными ключами или удалять несуществующие элементы, а также то, что при запросе k -го максимума он существует. Информация о том, какие данные будут добавляться, известна заранее. Необходимо обработать 100000 запросов за одну секунду.

Решение.

Если построение и модификацию ДДП применять в оперативном режиме — при поступлении каждого запроса, то структура дерева непредсказуема. Оно может «выродиться» в простой линейный список. В этом случае время обработки запроса может оказаться $O(n)$, а если дерево будет сбалансированным, то потребуется всего лишь $O(\log_2 n)$, где n — суммарное количество различных элементов, которые будут добавляться.

Ключевым моментом для генерации идеи решения является последняя фраза в условии задачи. На предварительной стадии надо обработать запросы первого типа и создать по ним сбалансированное ДДП, отметив

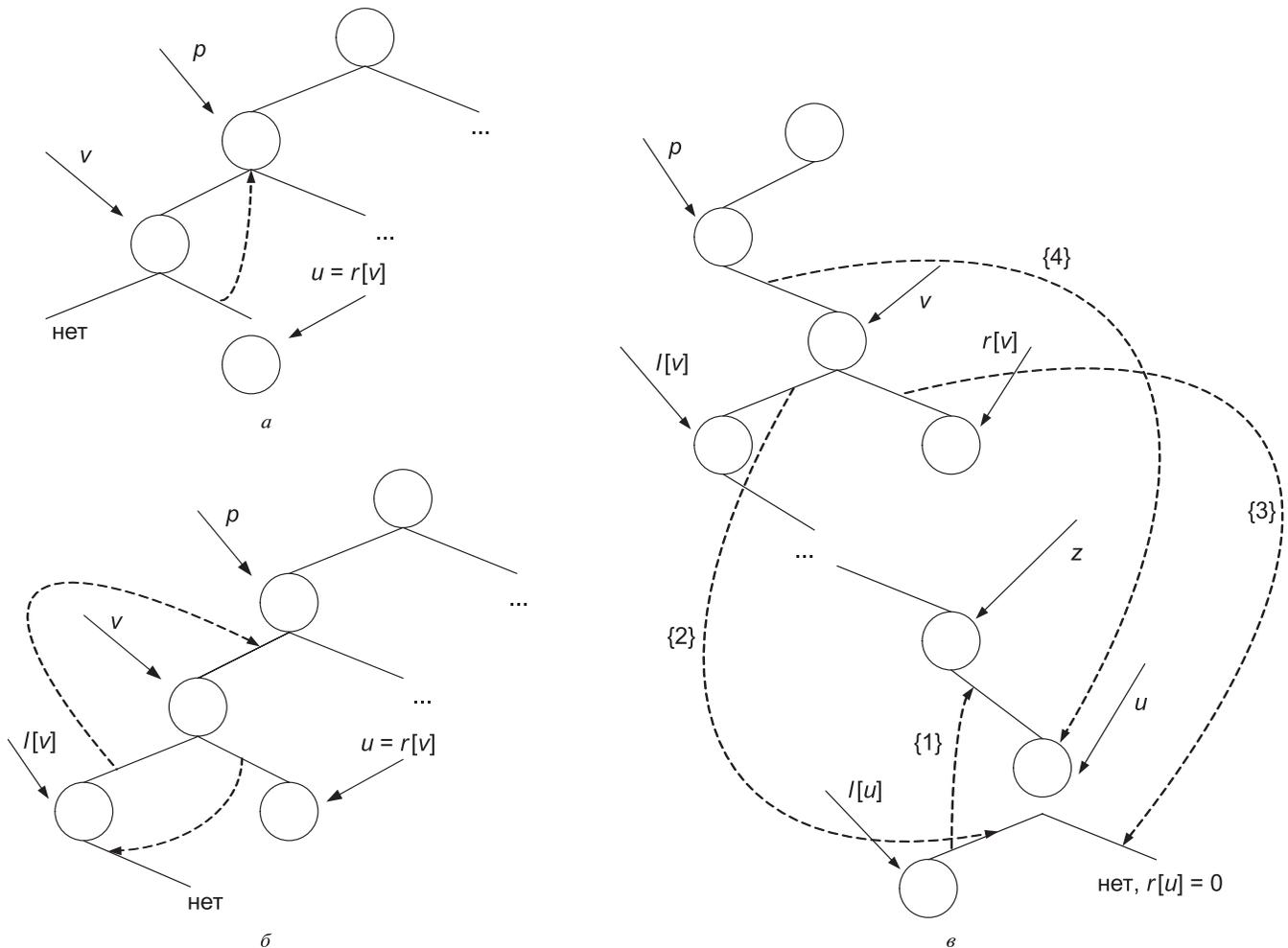


Рис. 7. Логика удаления элемента из ДДП

все его элементы как отсутствующие. С этой целью для каждого элемента определим признак k ($k[v]=0$ означает отсутствие). Кроме того, запросы третьего типа требуют знать для каждого элемента (v), как корня соответствующего поддерева, количество элементов в данном поддереве ($sk[v]$).

Описание данных:

```
const MAXM=100000;
var
  l, r, x, k, sk: array[0..MAXM+1] of integer;
  n: integer;
```

Пусть ДДП построено, тогда операции вставки, удаления осуществляются простым добавлением 1 или -1 к полю k найденного элемента. А так как дерево сбалансированно, то время выполнения операции пропорционально $O(\log_2 n)$.

```
procedure change(key, delta: integer);
var v: integer;
begin
  v:=1;
  while (x[v]<>key) do
  begin
    if key<x[v] then v:=l[v] else v:=r[v];
    sk[v]:=sk[v] + delta;
  end;
  k[v]:=k[v]+delta;
end;
```

Поиск k -го максимального элемента при построенном сбалансированном дереве выполняется с помощью достаточно простой функции и за время $O(\log_2 n)$.

```
function kthmax(z: integer): integer;
var v: integer;
begin
  v:=r[1];
  while (z<=sk[r[v]]) or (z>sk[r[v]]+k[v]) do
  {Если z больше чем суммарное количество
  вершин в правом поддереве и текущей вершине,
  то мы должны идти влево. А если меньше
  чем количество вершин в правом поддереве,
  то должны идти вправо, иначе - выводить
  текущую вершину. При этом мы все время
  "смотрим" на правое поддерево, так как
  ищем максимум}
  if z<=sk[r[v]] then v:=r[v]
  else
  begin
    z:=z-sk[r[v]]-k[v];
    v:=l[v];
  end;
  kthmax:=x[v];
end;
```

Если отложить на время логику построения сбалансированного ДДП, то решение задачи выглядит следующим образом.

```
procedure solve;
var m, t, i: integer;
begin
  x[1]:=-1000000001; {фиктивный элемент}
  t:=1;
  read(m); {Количество запросов}
  for i:=1 to m do
  begin
    read(q[i], a[i]); {Массивы q и a должны быть
    объявлены}
    if (q[i]>0) then
    begin {Если запрос на создание элемента, то
    запоминаем значение ключа в массиве x}
      t:=t+1;
      x[t]:=a[i];
    end;
  end;
  {Сортируем массив добавляемых элементов одним из
  методов быстрой сортировки за время  $O(t \cdot \log_2 t)$ }
  sort(2, t);
  {Удаляем из x повторяющиеся элементы}
  n:=1;
  for i:=2 to t do
  if x[i]<>x[n] then
  begin
    n:=n+1;
    x[n]:=x[i];
  end;
  {Строим сбалансированное ДДП как правое поддерево
  фиктивного корня}
  r[1]:=make(2, n);
  {Отвечаем на запросы за общее время  $O(m \cdot \log_2 n)$ }
  for i:=1 to m do
  if q[i]=0 then writeln(kthmax(a[i]))
  else change(a[i], q[i]);
end;
```

Осталось построить сбалансированное двоичное дерево поиска по упорядоченному массиву чисел без повторяющихся элементов. Это делается с помощью простой рекурсивной логики, которая достояна того, чтобы быть заключением данного материала.

```
function make(i, j: integer): integer;
var m: integer;
begin
  if i>j then
  begin
    make:=0;
    exit;
  end;
  m:=(i+j) div 2;
  l[m]:=make(i, m-1);
  r[m]:=make(m+1, j);
  make:=m;
end;
```

Литературные и интернет-источники

1. Летняя компьютерная школа, г. Кострома. <http://video.lksh.ru/>
2. Окулов С. М. Абстрактные типы данных. М.: БИНОМ. Лаборатория знаний, 2009.

Г. О. Патрушев,

Институт математики, физики и информатики Красноярского государственного педагогического университета

АЛГОРИТМ ОРГАНИЗАЦИИ КРУГОВОГО ТУРНИРА*

Аннотация

В статье рассматриваются алгоритмические подходы к организации кругового турнира (по принципу «каждый с каждым»). Предлагаются как рекурсивные, так и нерекурсивные версии алгоритма для различного числа команд. Исследуются вопросы об оптимальной продолжительности такого турнира.

Ключевые слова: алгоритм кругового турнира, оптимальная продолжительность кругового турнира, рекурсивная версия кругового турнира.

Контактная информация

Патрушев Глеб Олегович, канд. физ.-мат. наук, доцент кафедры физики Института математики, физики и информатики Красноярского государственного педагогического университета; адрес: 660049, г. Красноярск, ул. Перенсона, д. 7; телефон: (391) 263-97-24; e-mail: gor63@mail.ru

G. O. Patrushev,

Institute of Mathematics, Physics and Informatics of Krasnoyarsk State Pedagogical University

ALGORITHM OF THE ORGANIZATION OF CIRCULAR TOURNAMENT

Abstract

In the article algorithmic approaches to the organization of circular tournament (by the principle "everyone with everyone") are considered. Both recursive and nonrecursive versions of algorithm for various number of teams are offered. Questions of the optimum duration of such tournament are investigated.

Keywords: algorithm of circular tournament, optimum duration of circular tournament, recursive version of circular tournament.

Началась эта история достаточно буднично. Ко мне подошел преподаватель физкультуры и попросил посодействовать в проведении кругового турнира по мини-футболу. Вся сложность заключалась в составлении расписания для этого турнира. Попытка алгоритмизации составления расписания натолкнулась на определенные трудности, процесс преодоления которых, думаю, весьма поучителен.

Составление матчевой сетки, в которой каждая команда должна сыграть со всеми остальными, является не такой легкой задачей, как могло бы показаться на первый взгляд. Процесс осложняется дополнительными требованиями, а именно:

- каждая команда должна играть не более одного матча в день;
- проведение всего турнира должно занять как можно меньше времени.

Ясно, что при нечетном числе команд каждый день в лучшем случае одна команда не задействована в матчах (ей нет пары). В худшем случае количество неиграющих команд может, видимо, оказаться и большим — если они уже встречались в матчах ранее.

Можно попытаться оценить минимальное количество дней, необходимых для проведения такого турнира.

Пусть в турнире принимают участие N команд. Тогда полное количество сыгранных матчей есть число сочетаний из N по 2 (в каждом матче принимают участие 2 команды):

$$C_N^2 = \frac{N!}{2!(N-2)!} = \frac{(N-1)N}{2}.$$

Если число команд четное, то в наилучшем случае каждый день задействованы все команды ($\frac{N}{2}$ пар). Таким образом, турнир никак не может занять меньше $N-1$ дня.

Расписание турнира удобно строить в виде таблицы, столбцы которой соответствуют дням турнира, строки — номерам команд, участвующих в турнире. Элементом на пересечении строки i и столбца j является номер команды, с которой команда i должна провести матч в j -й день турнира. Нолик в соответствующих клетках говорит о том, что в этот день команда отдыхает.

На рисунке 1 приведено возможное распределение 5 команд по 5 дням турнира. Ясно, что этот вариант соответствует целой группе расписаний с точностью до перестановки строк и столбцов.

		Дни турнира				
		1	2	3	4	5
Номера команд	1	4	5	0	3	2
	2	5	3	4	0	1
	3	0	2	5	1	4
	4	1	0	2	5	3
	5	2	1	3	4	0

Рис. 1. Вариант расписания для 5 команд

Приведенное расписание было составлено вручную. Это было сделано для того, чтобы подметить какие-либо закономерности в размещении команд, понять принцип составления такого расписания и впоследствии автоматизировать его. Оказалось, что схема действий человека очень подходит под алгоритм *backtracking*'а, иначе называемый «метод отбора-отказа» или «метод проб и ошибок». Он известен широкому кругу программистов по знаменитой «задаче о ферзях» [2, 5].

* Материалы к статье можно скачать на сайте ИНФО: <http://www.infojournal.ru/infojournal/school/archive/8-2014>

Следуя этой идеологии, вначале расставляем соперников для первой команды, затем, с учетом уже занятых команд, — для второй. Если на каком-то шаге получается слишком большое число неиграющих команд в один из дней (в идеале больше одной), возвращаемся на шаг назад, пробуем другой вариант расстановки и снова идем вперед. Если на каком-то шаге перебор всех возможных вариантов расстановки не приводит к успеху, значит, необходимо отступить еще на шаг назад, и т. д.

Для небольшого числа команд такой метод вполне приемлем, но после многочисленных попыток составить расписание всего для 9 команд автор капитулировал. Слишком большое количество информации, оказывается, необходимо запоминать в стеке возвратов на каждом шаге, а именно — очередность расстановки команд на матчи, с тем чтобы при откате на шаг назад отказаться именно от последнего варианта. Соответственно, следует ожидать, что и компьютерный вариант алгоритма будет излишне громоздок.

Итак, лобовой способ решения задачи потерпел фиаско. Настала пора посмотреть, а может быть, мы ломимся в открытую дверь и для подобных задач о расстановке давно придуманы эффективные алгоритмы?

Поиск в литературе, увы, не привел к изобилию подсказок. В основном фигурировали уже готовые таблицы турнира для 4, 10 команд и пр. Только в книге А. Ахо, Д. Хопкрофта и Д. Ульмана [1] обнаружилось рекомендации и подходы к этой задаче.

Правда, авторы рассматривают частный случай, а именно, когда число команд составляет степень двойки: 2, 4, 8, 16 и т. д. Для составления матчевой таблицы применяется метод декомпозиции, который получил широкое применение не только при разработке алгоритмов, но и в проектировании электронных схем, построении математических доказательств и в других сферах.

Метод декомпозиции позволяет составить расписание для половины команд. Это расписание составляется на основе рекурсивного применения данного алгоритма для половины этой половины команд и т. д. Когда количество команд будет сокращено до двух, возникает «базовая ситуация», в которой мы просто устанавливаем порядок проведения встреч между ними.

Допустим, в турнире участвуют 8 команд. Расписание для команд 1—4 заполняет верхний левый угол (4 строки × 3 столбца) составляемого расписания. Нижний левый угол (4 строки × 3 столбца) этого расписания должен свести между собой команды с более высокими номерами (5—8). Эта часть расписания получается путем прибавления числа 4 к каждому элементу в верхнем левом углу.

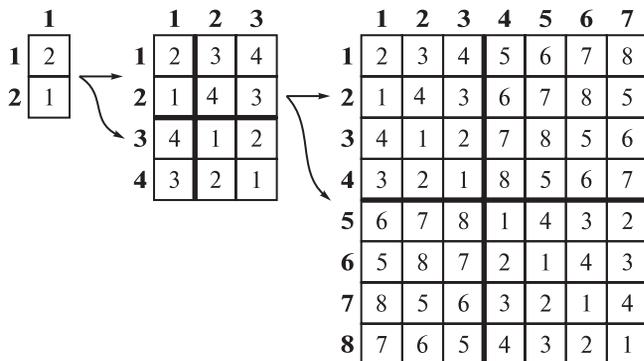


Рис. 2. Конструирование расписания методом декомпозиции для 2^k команд

Итак, нам удалось упростить задачу. Теперь остается свести между собой команды с низкими и более высокими номерами. Сделать это нетрудно: надо на 4-й день свести в пары команды, имеющие номера 1—4, с командами 5—8 соответственно, а в последующие дни просто циклически переставлять номера 5—8. Этот процесс показан на рисунке 2.

Составить программу согласно приведенным рекомендациям оказывается нетрудно. Ее текст с минимально необходимыми комментариями приведен ниже.

```
{рекурсивная версия}
const N=16; {число команд – степень двойки}
var
  {матчевая таблица}
  A: array [1..N-1,1..N] of byte;

procedure PrintTable;
{печать матчевой таблицы}
var i, j: integer;
begin
  for j:=1 to N do
    begin
      for i:=1 to N-1 do write(A[i,j]:3);
      writeln;
    end;
end; {PrintTable}

procedure TableForm(K: integer);
{формирование матчевой таблицы для K команд}
var i, j: integer;
begin
  if K<>2
  then TableForm(K div 2)
  else {определяем очередность явно}
  begin
    A[1,1]:=2;
    A[1,2]:=1;
  end;
  {левый нижний угол таблицы}
  for j:=1 to K div 2 do
    for i:=1 to (K-1) div 2 do
      A[i,j+K div 2]:=A[i,j]+K div 2;
  {правый верхний угол таблицы}
  for j:=1 to K div 2 do
    A[K div 2,j]:=j+K div 2;
  for i:=K div 2 +1 to K-1 do
    begin
      for j:=1 to K div 2 -1 do
        A[i,j]:=A[i-1,j+1];
        A[i,K div 2]:=A[i-1,1]
      end;
  {правый нижний угол таблицы}
  for j:=K div 2+1 to K do
    A[K div 2, j]:=j-K div 2;
  for i:=K div 2 +1 to K-1 do
    begin
      for j:=K div 2 +1 to K do
        A[i,j]:=A[i-1,j+1];
        A[i,K]:=A[i-1,K div 2+1]
      end;
  end; {TableForm}

BEGIN
  TableForm(N);
  PrintTable;
END.
```

Поскольку в программе есть лишь единственный рекурсивный вызов в начале условной конструкции, то разумной будет замена рекурсии на итерацию, как и считает Н. Вирт [3]. Соответствующим образом измененная программа имеет следующий вид:

```

{рекурсия заменена на итерацию}
const N=16; {число команд – степень двойки}
var
  A: array [1..N-1,1..N] of byte;
  K: integer;

procedure PrintTable;
{печать матчевой таблицы}
var i, j: integer;
begin
  for j:=1 to N do
    begin
      for i:=1 to N-1 do write(A[i,j]:3);
      writeln;
    end;
end; {PrintTable}

procedure TableForm;
var i, j: integer;
begin
  {левый нижний угол}
  for j:=1 to K div 2 do
    for i:=1 to (K-1) div 2 do
      A[i,j+K div 2]:=A[i,j]+K div 2;
  {правый верхний угол}
  for j:=1 to K div 2 do
    A[K div 2,j]:=j+K div 2;
  for i:=K div 2 + 1 to K-1 do
    begin
      for j:=1 to K div 2 - 1 do
        A[i,j]:=A[i-1,j+1];
        A[i,K div 2]:=A[i-1,1]
      end;
  {правый нижний угол}
  for j:=K div 2+1 to K do
    A[K div 2, j]:=j-K div 2;
  for i:=K div 2 + 1 to K-1 do
    begin
      for j:=K div 2 + 1 to K do
        A[i,j]:=A[i-1,j+1];
        A[i,K]:=A[i-1,K div 2+1]
      end;
end; {TableForm}

BEGIN
{определяем очередность явно для 2 команд}
A[1,1]:=2; A[1,2]:=1; K:=1;
while K<>N do
  begin
    K:=2*K;
    TableForm;
  end;
PrintTable;
END.

```

В [2, 5] предлагается утверждение, что, когда число команд является степенью двойки $N = 2^k$, турнир всегда можно провести в $N - 1$ дней. Неясными остаются два момента:

- Охватывает ли $N = 2^k$ все случаи проведения турнира в минимально допустимые $N - 1$ дней?
- Как построить расписание, если число команд отличается от $N = 2^k$, и какова в этом случае будет продолжительность турнира?

Ответ на первый вопрос звучит отрицательно. Метод проб и ошибок дает возможность провести турнир для $N = 6$ команд (а это не степень двойки!) за 5 дней. Соответствующая схема может выглядеть так, как представлено на рисунке 3.

Приведенные в [1] далее рекомендации по второму вопросу звучат следующим образом. При нечетном количестве участников каждый день один из них должен полностью освобождаться от проведения матчей,

	1	2	3	4	5
1	4	5	6	3	2
2	5	3	4	6	1
3	6	2	5	1	4
4	1	6	2	5	3
5	2	1	3	4	6
6	3	4	1	2	5

Рис. 3. Расписание турнира для 6 команд

а проведение турнира в целом займет N , а не $N - 1$ дней. Если, однако, сформировать две группы с нечетным количеством участников, тогда команды, освобожденные от проведения матчей в каждой такой группе, могут провести матч друг с другом.

Следование этому совету означает отход от ясной и простой логики половинного деления и вязывание в дебри проверки условий типа: четной или нечетной является очередная «половинка», на которые мы разбили список команд. Кроме того, остается неясным, почему верхняя граница длительности турнира при нечетном числе участников равна N . Корректного доказательства этого не приведено.

В замечательной книге венгерского математика Д. Пойа [7], которую, я считаю, следует прочесть всякому начинающему программисту, дается совет, как поступать с неподдающейся задачей. Если вольно пересказать Д. Пойа, то ее (задачу) следует видоизменить, поставить «с ног на голову», покрутить и т. д.

Именно на этом пути (покрутить задачу) неожиданно нашлось решение (неожиданно ли?), анализ которого позволил дать ответ на все поставленные выше вопросы.

Разобьем N команд на пары произвольным образом. Если число команд нечетное, дополним список команд фиктивной командой с нулевым номером. Команда, которой выпадет играть с ней, в этот день отдыхает. Полученные пары определяют матчи первого дня турнира. После чего, для определения соперников в следующий день, организуем «прокрутку» команд по схеме, приведенной на рисунке 4. Первую команду оставим на своем месте, а остальные сдвинем на одну позицию в направлении, показанном стрелками (вращать можно как в направлении движения часовой стрелки, так и против).

Получили расписание второго дня турнира. Для определения матчей третьего дня еще раз сдвинем команды на одну позицию, и так до тех пор, пока не сделаем полный круг. При таком способе ротации каждая команда сыграет со всеми своими соперниками по одному матчу без повторов.

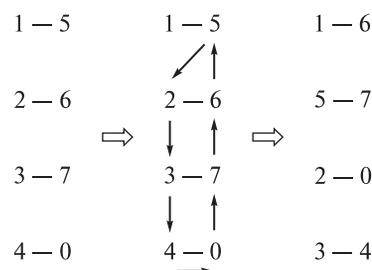


Рис. 4. Схема ротации команд при организации турнира

При четном числе команд, принимающих участие в турнире, сдвиг будет ровно на один меньше, чем команд (первая команда не принимает участие в ротации). Таким образом, становится ясно, что при любом четном количестве команд турнир можно уложить в $N-1$ день, прежде чем круг ротации команд замкнется. При нечетном количестве команд с учетом дополнительной фиктивной команды ротация совершается за N сдвигов. Получаем, что и в этом случае N дней достаточно для проведения турнира.

Проведенное рассуждение, разумеется, не является общепринятым косвенным доказательством того факта, что круговой турнир с произвольным числом команд N можно провести в N дней. Скорее, оно относится к так называемым конструктивным доказательствам [4]. Напомним, что конструктивным называется доказательство, которое факт существования объекта с заданными свойствами (в нашем случае N -дневное расписание турнира) устанавливает, опираясь на конкретную процедуру построения такого объекта.

Программная реализация алгоритма ротации не сложна:

```
const N=11; {число команд}
var
  List: array [1..N+1] of word; {список команд}
  A: array [0..N,1..N+1] of word; {таблица турнира}
  NN: word; {дополненное до четного число команд}
  i: integer;

procedure Init;
{инициализация списка команд}
var i: integer;
begin
  for i:=1 to NN do List[i]:=i;
  if Odd(N) then List[NN]:=0; {фиктивная команда}
end; {Init}

procedure DayToTable(k: Integer);
{матчи k-го дня - в таблицу}
var i: integer;
begin
  for i:=1 to (NN div 2) do
  begin
    A[List[i+NN div 2],k]:=List[i];
    A[List[i],k]:=List[i+NN div 2];
  end;
end; {DayToTable}

procedure Rotate;
{ротация команд}
var
  Z: word;
  i: integer;
begin
  Z:=List[NN div 2];
  for i:=(NN div 2) downto 3 do
    List[i]:=List[i-1];
  List[2]:=List[NN div 2 +1];
```

```
  for i:=(NN div 2+1) to NN-1 do
    List[i]:=List[i+1];
  List[NN]:=Z;
end; {Rotate}

procedure PrintTable;
{печать таблицы}
{горизонтальная ось - дни турнира,
вертикаль - команды по порядку,
начиная с первой}
var i, j: integer;
begin
  for i:=1 to N do {цикл по командам}
  begin
    for j:=1 to NN-1 do {цикл по дням}
      write(A[i,j]:4);
      writeln;
    end;
  end; {PrintTable}

BEGIN
{учет фиктивной команды}
if Odd(N) then NN:=N+1 else NN:=N;
Init;
for i:=1 to NN-1 do
begin
  DayToTable(i);
  Rotate;
end;
PrintTable;
END.
```

В заключение хочется отметить еще одно замечательное свойство последнего способа составления расписания турнира. Этот алгоритм не только прост и изящен, но и, по терминологии А. В. Кушниренко [6], является однопроходным, т. е. выполняет операцию со структурой, состоящей из N элементов, за N действий. Такие алгоритмы имеют наилучшую асимптотическую временную сложность $O(N)$.

Автор выражает благодарность А. В. Позднякову и В. О. Саяфарову за обсуждение результатов настоящей работы.

Литература

1. Ахо А., Хопкрофт Д., Ульман Д. Структуры данных и алгоритмы. М.: Вильямс, 2000.
2. Брудно А. Л., Каплан Л. И. Московские олимпиады по программированию. М.: Наука, 1990.
3. Вирт Н. Алгоритмы и структуры данных. М.: Мир, 1989.
4. Грэхем Р., Кнут Д., Паташник О. Конкретная математика. Основание информатики. М.: Мир, 1998.
5. Зубов В. С. Справочник программиста. М.: Филинь, 1999.
6. Кушниренко А. В., Лебедев Г. В. 12 лекций о том, для чего нужен школьный курс информатики и как его преподавать. М.: Лаборатория базовых знаний, 2000.
7. Пойа Д. Математика и правдоподобные рассуждения. М.: Наука, 1975.



В. Ф. Очков, Е. П. Богомолова,

Национальный исследовательский университет МЭИ, Москва

РИМСКИЕ — АРАБСКИЕ

Аннотация

В статье рассмотрена программа, позволяющая конвертировать римские и арабские числа и раскладывать суммы денег на банкноты и монеты.

Ключевые слова: римские числа, арабские числа, Mathcad.

Контактная информация

Очков Валерий Федорович, доктор тех. наук, профессор, Национальный исследовательский университет МЭИ, Москва; адрес: 111250, г. Москва, Красноказарменная ул., д. 14; телефон: (495) 362-71-71; e-mail: ochkov@twt.mpei.ac.ru

V. F. Ochkov, E. P. Bogomolova, National Research University MPEI, Moscow

ROMAN — ARABIC

Abstract

A program for the converting Roman and Arabic numerals, the amount of money decompose on notes and coin is considered in the article.

Keywords: Roman and Arabic numerals, Mathcad.

Одна из важных тем школьного курса информатики — это *позиционные системы счисления* (двоичная, восьмеричная, десятичная, шестнадцатеричная и др.), где значение зависит не только от *символа* (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, G, H и др.), но и от той *позиции*, где он расположен. Школьники должны уметь как минимум перевести десятичное число в двоичное и наоборот. Пакет Mathcad 15 имеет встроенные инструменты для работы с такими числами. На рисунке 1 можно видеть число, составленное из трех чисел в разной форме представления (2.73o — восьмеричное число, 9h — шестнадцатеричное число и 100b — двоичное число). Оно выведено «на печать» в четырех встроенных в Mathcad форматах. Если число не заканчивается суффиксом b (binary — двоичное), o (octal — восьмеричное) или h (hexadecimal — шестнадцатеричное), то это десятичное число.

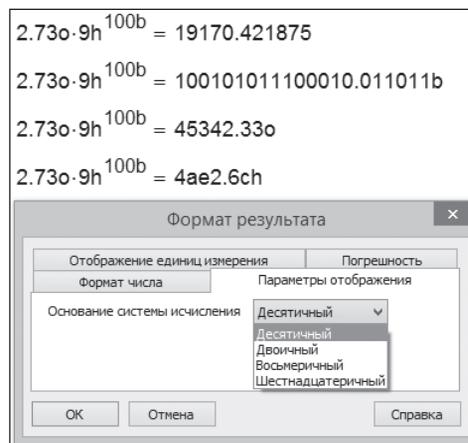


Рис. 1. Системы исчисления в среде Mathcad 15

В среде Mathcad несложно создать пользовательскую функцию с тремя аргументами для перевода записи чисел из одной системы счисления в другую. На рисунке 2 показан такой расчет (ее автор — датчанин С. Гроде), открытый в Интернете и позволяющий представлять целые числа в виде чисел, записанных в различных системах счисления: от двоичной до шестнадцатеричной. Так, троичное число 020202 эквивалентно семеричному числу 350, что видно из рисунка 2.

С числами в двоичном, восьмеричном и прочих представлениях в обыденной жизни мы почти не имеем дело*. Решая задачу на компьютере или просто с помощью калькулятора, мы вводим привычные десятичные числа. Компьютер переводит их в двоичный формат, проводит с ними нужные действия, получает двоичный ответ, который переводится в десятичный формат и выводится на экран дисплея.

Более привычным для нас является *римское (непозиционное)* представление чисел — *римские числа*. С ними, как это ни покажется странным, мы имеем дело очень часто — практически каждый день. И не только тогда, когда смотрим на часы с римскими числами на циферблате, но и тогда, когда...

* В последней версии Mathcad, Mathcad Prime, от чисел в недесятичном формате отказались.

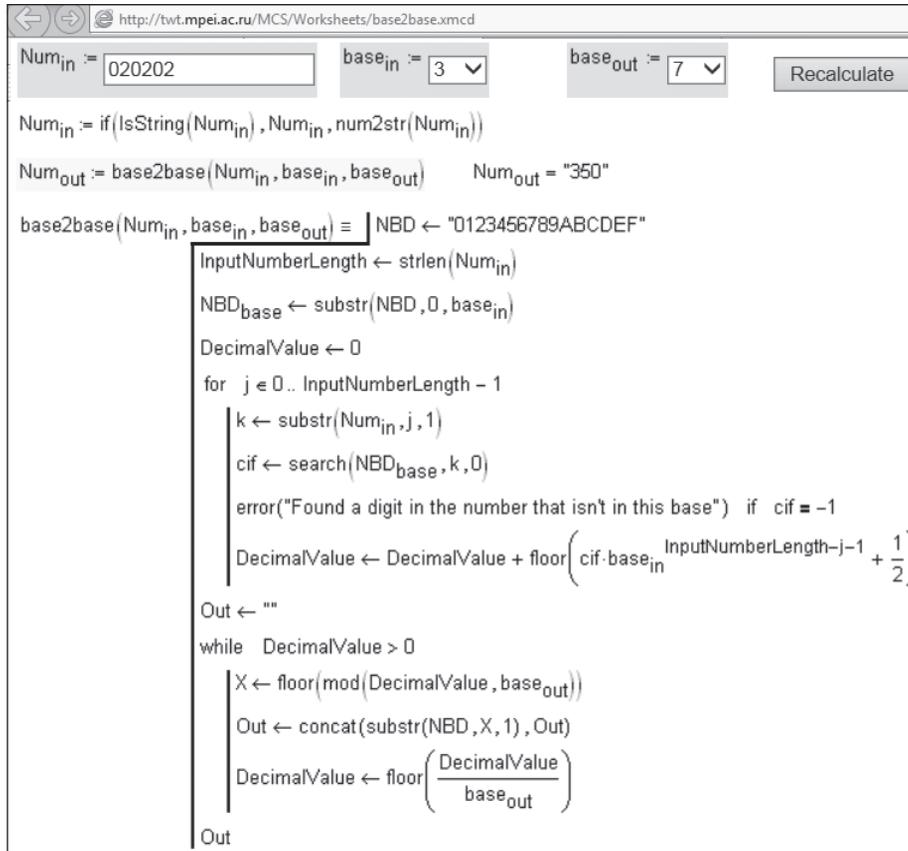


Рис. 2. Страница сайта для перевода записи чисел из одной системы счисления в другую

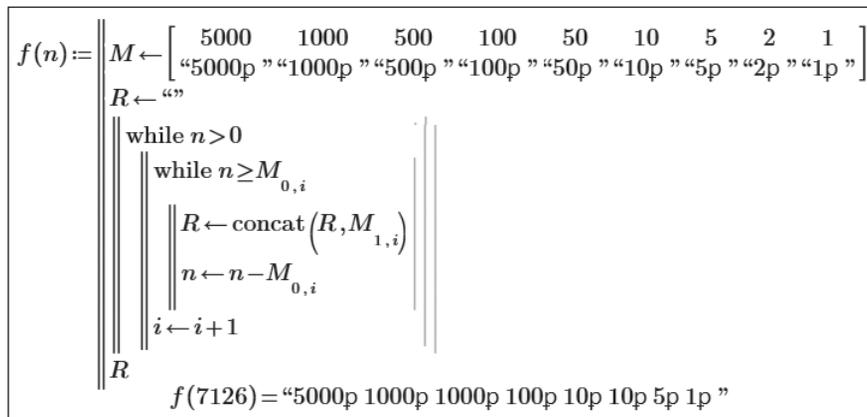


Рис. 3. Программа-функция для разложения суммы денег на банкноты и монеты

снимаем деньги в банкомате. В банкомат мы вводим арабское число (7 126 руб., например), а банкомат выдает нам... римское число (рис. 3). «Римский» счет денег мы ведем и тогда, когда нас просят записать сумму денег не числами, а словами: «Семь тысяч сто двадцать шесть рублей».

На рисунке 3 показана программа-функция, написанная для Mathcad Prime, которая переводит сумму денег в набор банкнот и монет. Если в этой программе матрицу банкнот и монет заменить на матрицу римских чисел (рис. 4), то она будет переводить арабские числа в римские.

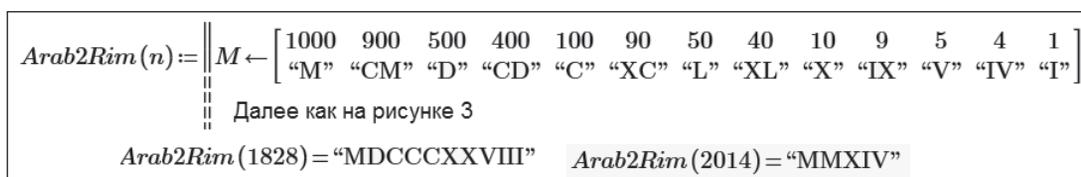


Рис. 4. Перевод арабского числа в римское

Алгоритм перевода арабского числа в римское довольно прост: из арабского числа n изымаются «римские банкноты и монеты» (вторая строка матрицы M) до тех пор, пока «арабские» деньги не кончатся ($\text{while } n > 0$). При этом проверяется, есть ли для этой операции соответствующая i -я банкнота или монета ($\text{while } n \geq M_{0,i}$). Встроенная в Mathcad функция *concat* собирает воедино цепочку символов — номинал банкнот и монет.

Сложнее написать обратную функцию — функцию перевода римского числа в арабское или функцию для подсчета суммы, заданной банкнотами и монетами. Такую операцию мы проводим в уме, когда видим старинное здание с годом постройки в виде римского числа на фронтоне или памятник с «римскими» датами рождения и смерти. Задачу можно упростить, если воспользоваться прямой (исходной) функцией перевода арабского числа в римское (рис. 3, 4): перебираются все арабские числа от единицы, далее они переводятся в «римский» вид, который сравнивается с исходным римским числом. Если произошло совпадение, то найденное арабское число выводится «на печать» (рис. 5).

$$\text{Rim2Arab}(R) := \left\| \begin{array}{l} n \leftarrow 1 \\ \text{while } \text{Arab2Rim}(n) \neq R \\ \quad \left\| \begin{array}{l} n \leftarrow n + 1 \\ n \end{array} \right. \end{array} \right\|$$

$\text{Rim2Arab}(\text{“MDCCLXXXVIII”}) = 1828$
 $\text{Rim2Arab}(\text{“MMXIV”}) = 2014$

Рис. 5. Перевод римского числа в арабское

Один из авторов видел у себя на даче одно необычное использование римских чисел. На соседнем участке

строилась баня: собирался сруб из бревен, которые были помечены зарубками — парами римских цифр: I I, II IV, III IX и т. д. Первая цифра означала номер стены бани от первой до четвертой, а вторая — номер бревна в этой стене от самого нижнего до самого верхнего. Этот сруб был изготовлен где-то далеко и перевезен на новое место. Перед его разборкой плотники пометили бревна римскими числами потому, что арабские числа топором «написать» довольно сложно.

Римские числа можно рассматривать как некие зашифрованные арабские числа. В связи с этим можно предложить школьникам на уроках информатики такую игру: создать матрицу, подобную той, которая показана на рисунке 4, и по ней шифровать арабские числа, которые другие школьники будут пытаться расшифровывать [1]. Упомянутый нами банкомат не только переводит арабское число затребованной суммы денег в римское число банкнот и монет (см. рис. 3), но и шифрует все введенные клиентами банка числа (PIN-код банковской карточки, например) перед пересылкой их в банк, выдавший карточку. Изучение приемов шифровки — это не только очень увлекательное, но и очень полезное занятие, помогающее стать востребованным специалистом в области безопасности информационных технологий.

И последнее. Было бы лучше, если бы у человека в процессе эволюции оказалось бы не десять, а **восемь** пальцев — по четыре на каждой из **двух** рук. Как у персонажей некоторых мультфильмов. В этом случае наша основная система счисления была бы не десятичной, а восьмеричной, что упростило бы наше общение с компьютером, базирующемся на **байте** (восьмерка) и **бите** (двойка).

Литература

1. *Очков В. Ф.* Mathcad и криптография // Информатика в школе. 2013. № 10.

НОВОСТИ

Число образовательных учреждений для одаренных детей в России увеличат

Минобрнауки РФ планирует увеличить число школ для одаренных детей и упорядочить их деятельность. Как поясняет пресс-служба министерства, осенью этого года вступит в силу приказ ведомства, устанавливающий порядок создания и работы таких организаций.

«В России и сейчас существуют школы для одаренных ребят, однако в основе их деятельности пока единственный документ — закон “Об образовании в РФ”. Теперь же работа и организация подобных образовательных организаций будут упорядочены специальным приказом министерства», — цитирует ИТАР-ТАСС пресс-службу Минобрнауки.

Приказ коснется нетиповых школ, как правило, с углубленным изучением отдельных предметов — региональных лицеев, гимназий на базе ведущих вузов и других организаций. «Главная цель, которую преследует министерство, — индивидуальная поддержка

и дальнейшее развитие способностей одаренных детей, — подчеркнули в ведомстве. — Планируется, что документ будет способствовать развитию сети учебных заведений для таких ребят, и число таких школ значительно увеличится».

Приказ ведомства предусматривает создание особых условий обучения для одаренных детей. Так, соотношение численности научно-педагогических сотрудников к численности учащихся должно составлять один к трем, т. е. на одного педагога должно приходиться не более трех детей. Кроме того, подобные инновационные школы смогут сами определять систему оценок при проведении контроля успеваемости и промежуточной аттестации.

«Тем не менее в девятом классе одаренным детям, как и всем другим, нужно будет проходить государственную итоговую аттестацию, а в одиннадцатом классе — сдать ЕГЭ», — заключили в министерстве.

(По материалам «Учительской газеты»)

В. Д. Кильдишов,

Западно-Подмосковный институт туризма (филиал Российской международной академии туризма)

СОЗДАНИЕ ЧАСОВ С ИСПОЛЬЗОВАНИЕМ ДИАГРАММЫ MS EXCEL

Аннотация

В статье представлена методика создания модели часов реального времени с использованием диаграммы MS Excel.

Ключевые слова: модель, часы, реальное время, диаграмма, MS Excel.

Контактная информация

Кильдишов Вячеслав Дмитриевич,
канд. тех. наук, доцент, профессор Российской международной академии туризма, профессор кафедры управления персоналом Западно-Подмосковного института туризма; адрес: 143050, Московская область, Одинцовский р-н, р.п. Большие Вяземы, ул. Институт, вл. 5, стр. 1; телефон: (495) 598-26-98; e-mail: kildishov47@rambler.ru

V. D. Kildishov,

Western-Moscow Institute of Tourism (branch of Russian International Academy for Tourism)

CREATION OF CLOCK WITH THE USE OF A DIAGRAM MS EXCEL

Abstract

The article describes the procedure of creating the model of the clock with real time with the use of a MS Excel diagram.

Keywords: model, clock, real time, diagram, MS Excel.

Особый интерес всегда вызывают динамические объекты, к которым можно отнести часы. Рассмотрим методику создания модели часов с использованием MS Excel. Часы должны показывать реальное время.

Создание модели можно разбить на следующие этапы:

- формирование ячеек с координатами циферблата и стрелок;
- построение изображения часов с использованием диаграммы;
- форматирование элементов изображения часов;
- применение процедур для получения реального времени и управления часами.

Для удобства работы объединим первые два этапа и последовательно будем создавать элементы часов.

Начнем с циферблата часов. Обычно на циферблате по окружности располагаются 60 меток, которые соответствуют числу секунд в минуте. Вычислим в ячейках угловые шаги делений для секунд, минут и часов (A3:C3) с учетом движения стрелок.

	A	B	C	D	E	F	G
1	Шаг в радианах				Радиус		
2	Секунд	Минут	Часа		Циферблата		Метки
3	0,10472	0,00873	0,5236		10		9
	=2*ПИ()/60	=2*ПИ()/(60*6*2)	=2*ПИ()/(12)				

Для организации перемещения секундной стрелки разделим 360° на 60 (A3). Зададим в условных единицах радиус окружности меток (E3). Создадим массив A6:D66, в котором вычислим координаты точек для всех секундных меток в прямоугольной системе координат по формулам:

$$x_i = R \cdot \cos \varphi_i,$$

$$y_i = R \cdot \sin \varphi_i,$$

где R — радиус окружности меток, $\varphi_i = 2 \cdot \pi / 60$ — угол i -й секундной метки, который вычисляем в ячейках B6:B66 по значениям i , сформированным в столбце A.

В этом случае центр циферблата находится в начале координат. Для замыкания окружности в последней ячейке массива продублируем координаты первой точки. Выделим с запасом ячейки массива и обратимся к диаграмме *Точечная с гладкими кривыми и маркерами*. Можно убрать соединительные линии между метками, но лучше их оставить. Однако необходимо зафиксировать минимальные и максимальные значения осей, добавить основные линии сетки, а также «протяжкой» обеспечить изображение циферблата в виде окружности.

	A	B	C	D
5	Секунды	Угол, рад	X	Y
6	0	0	0,00	10,00
7	1	0,10472	1,05	9,95
8	2	0,20944	2,08	9,78
9	3	0,31416	3,09	9,51
10	=A6*\$A\$3	41888	4,07	9,14
11	5	0,5236	5,00	8,66
12	=E\$3*SIN(B6)		5,88	8,09
13	7	0,73304	6,69	7,43
14	8	0,83776	=E\$3*COS(B6)	6,69
15	9	0,94248	8,09	5,88
16	10	1,0472	8,66	5,00

В ячейках C68:D75 вычислим координаты стрелок. Строка 67 пустая. Это позволит разделить изображения элементов часов. Начало координат отрезков стрелок привязываем к началу координат. Координаты

окончаний стрелок рассчитываем с учетом текущих значений часов, минут, секунд и цен соответствующих угловых делений. Формулы вычислений углов стрелок приведены ниже:

$$\begin{aligned}\alpha_s &= \Delta\alpha_s \cdot s, \\ \alpha_m &= \Delta\alpha_m \cdot m \cdot 12, \\ \alpha_h &= \Delta\alpha_h \cdot h + \Delta\alpha_m \cdot m,\end{aligned}$$

где $\alpha_s, \alpha_m, \alpha_h$ — углы секундной, минутной и часовой стрелок, отсчитываемые от оси ординат по часовой стрелке; $\Delta\alpha_s, \Delta\alpha_m, \Delta\alpha_h$ — шаги изменения углов секундной, минутной и часовой стрелок соответственно; s, m, h — текущие значения секунд, минут, часов реального времени.

	A	B	C	D	
62	56	5,86431	-4,07	9,14	
63	57	5,96903	-3,09	9,51	
64		$=(\$E\$3-1)*\text{SIN}(\$I\$3*\$A\$3)$	0,08	9,78	
65		$=(\$E\$3-1)*\text{COS}(\$I\$3*\$A\$3)$	5	9,95	
66			0,00		
67			0,00		
68		Секунды	0,00	0,00	
69			-9,00	0,00	
70			$=(\$E\$3-2)*\text{SIN}(\$B\$3*\$J\$3*12)$	0,00	
71			0,00		
72	Стрелка	Минуты	6,93	-4,00	
73			0,00		
74			0,00	0,00	
75		Часы	1,04	5,91	
76			$=(\$E\$3-4)*\text{COS}(\$K\$3*\$C\$3+\$B\$3*\$J\$3)$		
77			0,00	8,00	
78			$=(\$G\$3-1)*\text{SIN}(\$B\$36)$	0,00	
79			0,00	9,00	
80	Метки для часовой стрелки		8,00	0,00	
81		3	9,00	0,00	
82					
83		6	0,00	-8,00	
84				0,00	-9,00
85			$=(\$G\$3)*\text{COS}(\$B\$36)$		
86			-8,00	0,00	
87			-9,00	0,00	

При таком задании углов секундная стрелка пробегает по всем меткам циферблата. Минутная стрелка — так же, но только с интервалами, равными 60 секундам. Часовая — движется в угловом секторе часа с интервалами, равными одной минуте. Текущие значения времени выбираются из ячеек I3:K3, эти значения целесообразно заносить с клавиатуры в начале создания модели для ее отладки. Для изображения стрелок формируем пары ячеек с пропуском одной строки. Первая пара координат соответствует началу стрелки. В них координаты равны нулю. Во второй паре вычисляем прямоугольные координаты концов стрелок по формулам, приведенным выше. В качестве угловых координат используем $\alpha_s, \alpha_m, \alpha_h$. Радиус секундной стрелки меньше радиуса окружности циферблата на 1, радиус минутной стрелки — на 2, радиус часовой стрелки — на 4.

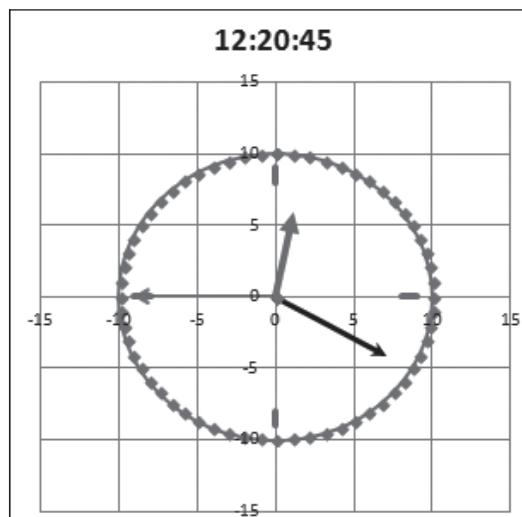
	H	I	J	K
1		Время		
2		Секунды	Минуты	Часы
3		45	20	12
4				
5	12:20:45	Запуск часов		
6		Останов		
7		Обнуление		
8				
9				
10				

После появления отрезков стрелок нужно сформировать изображения стрелок. Активизируем точки концов стрелок, в контекстном меню выбираем **Формат точки данных**, в окне выбираем пункт **Тип линии** и в списке **Тип начала** выделяем нужное изображение стрелки.

Сформируем на циферблате четыре метки для часовой стрелки в массиве C77:D87. Создадим укороченные отрезки меток для 0, 3, 6 и 9 часов. Длина отрезков равна одной условной единице. Для удобства задаем максимальный радиус часовых меток, который на единицу меньше радиуса циферблата (G3). Расчет координат отрезков производим по тем же формулам. Используем значения углов для соответствующих значений времени, а значение радиуса метки часа уменьшаем на единицу расчета для начального значения координат отрезка.

При необходимости нужно скорректировать диапазон данных координат в диаграмме с учетом того, что массив увеличился: C6:D87. Для реалистичности изображения меток произведем форматирование маркеров и линий отрезков. Щелкаем дважды на нужном маркере отрезка, через контекстное меню **Формат точки данных** в диалоговом окне выбираем **Параметры маркера** и устанавливаем **Нет**. Можно изменить цвет и толщину линий меток часов.

Обеспечим формирование текущего значения реального времени для моделирования движения стрелок. В первую очередь определимся с ячейками, в которых будут записываться значения текущего времени. В ячейке H5 индицируется текущее время. В ячейках I3:K3 — значения секунд, минут и часов, которые используются для вычисления положения соответствующих стрелок. Для индикации числового значения времени в диаграмме в окне **Название диаграммы** устанавливаем курсор, в строке формул вносим знак равенства и щелкаем в ячейке с текущим временем (H5). В названии будем видеть время, которое можно сравнить с положением стрелок часов.



Теперь надо «подключиться» к реальному времени. Создаем через **Разработчик** и **Элементы управления** три кнопки и через их свойства формируем названия. Далее нужно вставить соответствующие программы, которые выполняются по щелчку. Дважды щелкаем на каждой кнопке в режиме Конструктора и получаем заготовки для программ. В них заносим следующие тексты программ кнопок:

```

Sub CommandButton1_Click() 'Запуск часов по щелчку
Call Bklshetchika 'Вызов процедуры включения
    счетчика
End Sub

Sub CommandButton2_Click() 'Останов часов
    по щелчку
Call Stopchet 'Вызов процедуры остановки счетчика
End Sub

CommandButton3_Click() 'Обнуление часов по щелчку
Cells(3, 9)=0 'Обнуление ячейки счетчика Секунды
Cells(3, 10)=0 'Обнуление ячейки счетчика Минуты
Cells(3, 11)=0 'Обнуление ячейки счетчика Часы
Cells(5, 8)=Null 'Обнуление ячейки счетчика
    времени ч.м.с.
End Sub.

```

Остается создать процедуры, которые обеспечивают получение реального времени каждую секунду. Заходим в Visual Basic и через Insert активируем Module. В его окне объявляем переменную времени и создаем программы следующих процедур:

```

Dim Zadaniecledtime As Date 'Объявление переменной
    в формате дата
Sub Zapistime() 'Процедура записи текущего значения
    времени
Cells(5, 8)=Now 'Запись текущего значения времени
Cells(3, 11)=Hour(Now) 'Запись часа

```

```

Cells(3, 10)=Minute(Now) 'Запись минуты
Cells(3, 9)=Second(Now) 'Запись секунды
Call Bklshetchika
End Sub

```

```

Sub Bklshetchika() 'Запуск счетчика с интервалом
    1 секунда
Zadaniecledtime=Now+TimeValue("00:00:01") 'За-
    пись значения времени для запуска счетчика с уче-
    том интервала
Application.OnTime Zadaniecledtime, "Zapistime"
'Вызов процедуры записи счетчика времени в задан-
    ное время
End Sub

```

```

Sub Stopchet() 'Процедура остановки счетчика
Application.OnTime
    EarliestTime:=Zadaniecledtime,
    Procedure:="Zapistime", Schedule:=False 'Форми-
    рование запрета вызова процедуры записи значения
    времени счетчика
End Sub.

```

Сохраняем созданные процедуры. Модель часов создана. В Интернете можно найти подробные описания используемых методов и функций в модели.

При открытии файла нужно разрешить функционирование макросов и элементов ActiveX. Затем включить часы, а в конце остановить. При необходимости можно обнулить часы.

НОВОСТИ

В школах появятся 3D-табло, подсказывающие ученикам безопасный путь домой

В столичных школах планируют установить 3D-табло с планом безопасной дороги домой для учеников: на схеме будут указаны маршруты, следуя которым дети обойдут помойки, точки скопления бездомных и гастарбайтеров, места, где наиболее часто происходят преступления, а также опасные участки, с точки зрения автоаварий, пишут «Известия».

Пилотный проект уже проходит в школе № 875 района Тропарево-Никулино — там стенд уже поставлен. Для того чтобы разработать «безопасный путь следования детей», сотрудники школы, родители и участковые в течение трех месяцев обследовали все близлежащие территории. Как рассказал руководитель межведомственного

ресурсного Центра мониторинга и экспертизы безопасности образовательной среды МГППУ, разработавшего проект, Иван Бордик, «картографическая основа» для таких табло направлена во все школы Москвы.

Созданием маршрутов для каждой школы займется департамент образования, а средства на стенды учреждения выделят самостоятельно. Эффект 3D будет достигаться с помощью использования голографии, само табло может быть изготовлено из металла или пластика, в зависимости от бюджета. Также школы по желанию смогут установить электронные табло с безопасным маршрутом.

(По материалам портала NEWSmsk.com)

Общественный транспорт в Москве обеспечили электронной библиотекой

В московском общественном транспорте появились наклейки с QR-кодами, с помощью которых теперь можно скачать электронную книгу в смартфон или планшет. Совместный проект по размещению электронных книг в столичных автобусах, троллейбусах и трамваях презентовали Мосгортранс, Московский городской библиотечный центр и книжный онлайн-сервис «ЛитРес».

В салонах наземного транспорта наклеили QR-коды, каждый из которых соответствует книге. Отсканировав код в специальной программе телефона или планшета, вы попадете на сайт cityreader.ru с возможностями прочитать книгу онлайн, скачать ее или найти ближайшую библиотеку, в которой можно взять бумажный экземпляр.

Пока для пассажиров доступно 50 популярных книг. Среди них произведения Виктора Пелевина, Владими-

ра Сорокина, Людмилы Улицкой, Сергея Лукьяненко, Михаила Лермонтова, Льва Толстого и другой русской и зарубежной литературы. Большинство из них в бесплатном доступе, но часть произведений представлена фрагментом из нескольких первых глав, полную же версию можно купить в интернет-магазине. Пока коды наклеят в 676 единицах подвижного состава. В проекте задействованы по одному автобусному, троллейбусному парку и трамвайному депо, которые располагаются на юго-западе Москвы. В пилотном режиме проект проработает в течение месяца. В следующем году количество наклеек планируется увеличить, а в конечном итоге оснастить ими весь подвижной состав Мосгортранса.

(По материалам «Учительской газеты»)

КОНКУРСЫ

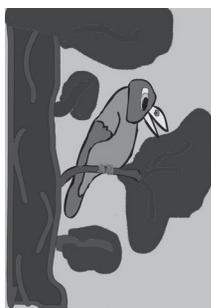
В оформлении обложки данного номера журнала «Информатика в школе» использованы работы следующих авторов — победителей конкурса цифровых изображений и фотографий:



Кристина Петрова,
 учащаяся творческого объединения
 «Информационные технологии» Станции юных техников,
 ученица средней общеобразовательной школы № 49,
 г. Чебоксары, Чувашская Республика



Екатерина Каменева,
 ученица гимназии № 39, г. Петропавловск-Камчатский



Анастасия Степанова,
 учащаяся творческого объединения
 «Информационные технологии»
 Станции юных техников,
 ученица средней общеобразовательной
 школы № 49, г. Чебоксары,
 Чувашская Республика



Ульяна Порозова,
 ученица средней общеобразовательной школы № 1,
 г. Кунгур, Пермский край



Александра Константинова,
 учащаяся творческого объединения
 «Информационные технологии»
 Станции юных техников,
 ученица средней общеобразовательной
 школы № 49, г. Чебоксары,
 Чувашская Республика



Дмитрий Лычак,
 ученик гимназии № 39, г. Петропавловск-Камчатский



Никита Алешников,
 ученик средней общеобразовательной школы № 8,
 г. Ноябрьск, Ямало-Ненецкий автономный округ

Уважаемые авторы и читатели!

Издательство «Образование и Информатика» объявляет о проведении конкурса цифровых изображений и фотографий, которые будут размещены на обложке журнала «Информатика в школе» в первом полугодии 2015 года.

Подробности — на сайте издательства «Образование и Информатика»: <http://www.infojournal.ru/>

ПРАВИЛА ДЛЯ АВТОРОВ

Общие положения

Все присланные статьи рецензируются. Публикация статей возможна только при наличии положительного отзыва рецензентов.

Поскольку рецензирование и предпечатная подготовка материалов занимают не менее трех месяцев, статьи следует присылать в редакцию заблаговременно.

Редакция не берет платы за публикацию рукописей аспирантов.

Требования к файлам рукописи

1. Текст статьи должен быть представлен в формате текстового редактора Microsoft Word (*.doc, *.rtf):

- формат листа — А4;
- все поля по 2 см;
- шрифт — Times New Roman, кегль — 12 пт, расстояние между строками 1,5 (полтора) интервала.
- графические материалы вставлены в текст.

2. Файл со статьей должен содержать следующие данные для публикации, **необходимо строго придерживаться указанной ниже последовательности:**

- **И. О. Фамилия** автора(ов) на русском языке.
- **Место работы** автора(ов) на русском языке. Необходимо указать место работы каждого автора. Если из названия организации не следует принадлежность к населенному пункту, через запятую указать название населенного пункта.
- **Название статьи** на русском языке.
- **Аннотация** на русском языке.
- **Ключевые слова** на русском языке (через запятую).
- **Подробная информация об авторах:** для каждого из авторов фамилия, имя, отчество (полностью), ученая степень, ученое звание, должность и место работы, адрес места работы (с индексом), рабочий телефон (с кодом города), адрес электронной почты (e-mail).
- **И. О. Фамилия** автора(ов) на английском языке.
- **Место работы** автора(ов) на английском языке.
- **Название статьи** на английском языке.
- **Аннотация** на английском языке.
- **Ключевые слова** на английском языке (через запятую).
- **Текст статьи** в указанном выше формате.
- **Список литературных и интернет-источников**, упорядоченный в алфавитном порядке.

Образец статьи можно скачать на сайте ИНФО: <http://infojournal.ru/authors/rules/>

3. К статье необходимо приложить сопроводительное письмо, содержащее подробные сведения об авторе: фамилия, имя, отчество (полностью), домашний почтовый адрес (с индексом), номера контактных телефонов (мобильного и домашнего), адрес электронной почты (e-mail). Данные сведения необходимы для оперативной связи с автором статьи и пересылки авторского экземпляра журнала и НЕ ПОДЛЕЖАТ ПУБЛИКАЦИИ. Если авторов несколько, необходимо представить указанные сведения обо всех авторах.

4. При необходимости статья может сопровождаться дополнительным материалом в электронном виде (презентации, листинги программ, книги Excel, примеры выполнения работ и др.), который будет размещен на сайте журнала.

5. Иллюстрации следует представлять в виде отдельных графических файлов (даже при их наличии в документе Word) в формате TIFF или JPG, разрешение — 300 пикселей на дюйм.

Пересылка материалов по электронной почте

1. Пересылать статьи, а также иллюстрации и дополнительные материалы к ним нужно по адресу **readinfo@infojournal.ru** в виде прикрепленных к письму файлов. Если файлы пересылаются в архивах, они должны быть упакованы архиваторами WinZIP или WinRAR. **Самораспаковывающиеся архивы не допускаются!**

2. **В теме письма** необходимо написать:

- «Статья в ИНФО. Ф.И.О. автора(ов)» — для публикации в журнале «Информатика и образование»;
- «Статья в ИвШ. Ф.И.О. автора(ов)» — для публикации в журнале «Информатика в школе»;
- «Статья. Ф.И.О. автора(ов)» — для публикации в любом из журналов («Информатика и образование», «Информатика в школе»).

3. **В теле письма** обязательно должна присутствовать следующая информация:

- Ф.И.О. автора(ов).
- Название статьи.
- Текст сопроводительного письма со сведениями об авторе(ах).

Редакция оставляет за собой право не рассматривать к публикации статьи, прикрепленные к «пустым» письмам (не содержащим сопроводительную текстовую информацию).

4. При повторной отправке материалов, а также дополнений или исправлений необходимо обязательно сообщить об этом в сопроводительном тексте электронного письма с указанием Ф.И.О. автора, названия статьи и даты отправки предыдущего письма.

Журнал «Информатика в школе»

Индексы подписки (агентство «Роспечать»)
на 1-е полугодие 2015 года

- 81407 — для индивидуальных подписчиков
- 81408 — для организаций

Периодичность выхода: 5 номеров в полугодие (в январе не выходит)

Редакционная стоимость:
индивидуальная подписка — 150 руб.
подписка для организаций — 300 руб.



Федеральное государственное унитарное предприятие "Почта России" Ф СП - 1
Бланк заказа периодических изданий

АБОНЕМЕНТ На ~~газету~~ журнал
(индекс издания)

Информатика в школе
(наименование издания)

Количество комплектов

На 20**15** год по месяцам

1	2	3	4	5	6	7	8	9	10	11	12

Куда
(почтовый индекс) (адрес)

Кому

Линия отреза

ДОСТАВОЧНАЯ
КАРТОЧКА (индекс издания)

ПВ место литер

На ~~газету~~ журнал
(наименование издания)

Стоимость	подписки	руб.	Количество комплектов
	каталожная	руб.	
	переадресовки	руб.	

На 20**15** год по месяцам

1	2	3	4	5	6	7	8	9	10	11	12

<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Город
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	село
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	почтовый индекс
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	область
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Район
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	код улицы
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	улица
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	дом
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	корпус
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	квартира
					Фамилия И.О.

21 - 24 октября, 2014

Москва, ВДНХ, павильон 57



16-й Всероссийский форум «ОБРАЗОВАТЕЛЬНАЯ СРЕДА»

ОРГАНИЗАТОР:

ОАО «Выставка Достижений
Народного Хозяйства»

ПОДДЕРЖКА:

Комитет по науке,
образованию, культуре и
информационной политике
Совета Федерации
Федерального Собрания
Российской Федерации

Комитет по образованию
Государственной Думы
Российской Федерации
Торгово-промышленная
палата Российской Федерации

Совет ректоров вузов
Москвы и Московской области

СНГ: ОБРАЗОВАНИЕ

РЕГИОНАЛЬНЫЕ СИСТЕМЫ ОБРАЗОВАНИЯ

СОВРЕМЕННЫЕ ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОСНАЩЕНИЕ ОБРАЗОВАТЕЛЬНЫХ УЧРЕЖДЕНИЙ

УЧЕБНАЯ И РАЗВИВАЮЩАЯ ЛИТЕРАТУРА

Контакты:

129223, Россия, Москва, проспект Мира, домовладение 119,
ОАО «ВДНХ»

Тел.: +7 (495) 981-81-06, E-mail: edu@Vvcentre.ru

WWW.EDU-EXPO.RU