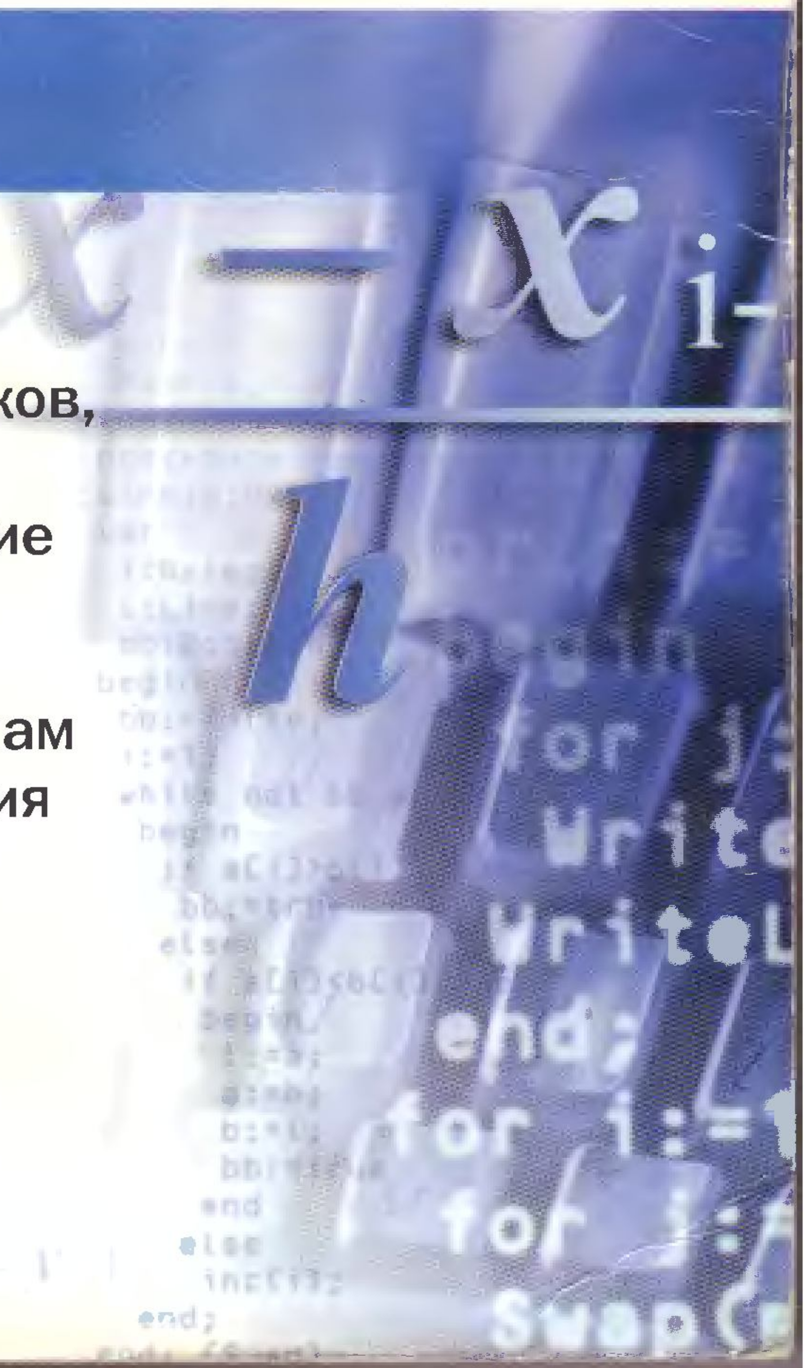
А. Г. Юркин

ПИТЕР®

ЗАДАЧНИК ПО ПРОГРАММИРОВАНИЮ



- для студентов и старшеклассников, изучающих программирование
- задачи по всем основным разделам программирования
- дифференциация по тематическим разделам и сложности

А. Юркин

Задачник по программированию

Главный редактор — *Е. Строганова*
Заведующий редакцией — *И. Корнеев*
Руководитель проекта — *А. Васильев*
Литературный редактор — *Ю. Королева*
Художник — *Н. Биржаков*
Корректоры — *Н. Лукина, М. Одинокова*
Верстка — *Р. Гришанов*

ББК 32.973-018я7
УДК 681.3.06(075)

Юркин А.
Ю74 Задачник по программированию. — СПб.: Питер, 2002. — 192 с.

ISBN 5-318-00399-0

Автор этой книги уверен, что практическое решение разнообразных по содержанию задач дает в обретении программистских навыков гораздо больший эффект, чем только лекционный курс и изучение многочисленных учебников, — нельзя научиться плавать в сухом бассейне. При формировании материала для задачника проработан не один десяток сборников и учебников. Из них по крупицам отобраны наиболее интересные формулировки; многие задачи являются оригинальными и родились в длительных дискуссиях с коллегами. В списке литературы приведены книги, использованные при подготовке сборника. Они могут быть полезны не только как источник дополнительных задач, но и иметь самостоятельное практическое значение, так как многие из них содержат хороший материал по технологии программирования.

ISBN 5-318-00399-0

Лицензия ИД № 05784 от 07.09.2001.
Налоговая льгота – общероссийский классификатор продукции
ОК 005-93, том 2; 95 3000 – книги и брошюры.
Подписано к печати 07.03.2002. Формат $84\times108^1/_{32}$. Усл. п. л. 13,44.
Доп. тираж 6000 экз. Заказ № 767.
ООО **«Питер Принт»**. 196105, Санкт-Петербург, ул. Благодатная, д. 67в.
Отпечатано с готовых диапозитивов в ГИПК «Лениздат» (типография им. Володарского)
Министерства РФ по делам печати, телерадиовещания и средств массовых коммуникаций.
191023, Санкт-Петербург, наб. р. Фонтанки, 59.

Содержание

Введение

Основу книги составил материал, опубликованный в пособии [35], выпущенном автором в 1999 году в Алтайском государственном университете. Оно в основном было написано для нужд раздела «Классическое программирование» курса «Информатика», преподаваемого на математическом факультете. До этого на протяжении многих лет материал прошел все стадии испытаний: от раздаточного материала в виде карточек с заданиями до серии методических пособий. За это время накопилось немало замечаний к формулировкам задач, появились дополнительные задачи, накопился опыт их использования.

Что такое программирование: ремесло, наука или искусство? По-моему, в нем можно найти все три начала, все зависит от характера и уровня подготовки программиста: кодирование заданного алгоритма — ремесло; разработка эффективных приемов и технологий — наука; поиск уникальных по эффективности и возможностям алгоритмов — искусство.

Нельзя овладеть программированием, прочитав сколь угодно много руководств и прослушав какой угодно курс лекций, как нельзя научиться плавать в сухом бассейне. Только продолжительное общение с компьютером, многочисленные эксперименты и обращение к HELP-средствам позволят овладеть ремеслом; если эти навыки подкрепить теоретически — и наукой, а наличие изобретательского зуда позволит дорасти до искусства.

Автор надеется помочь читателям продвинуться на каждом из этих уровней: от кодирования очевидных алгоритмов до пробуждения самолюбия в поисках эффективного решения сложных задач. При поиске задач для задачника проработан не один десяток сборников и учебников. Из них по крупицам отобраны наиболее интересные формулировки (например, Задача Иосифа: 5.11); многие задачи являются оригинальными и родились в длительных дискуссиях с коллегами (в частности с, А. Максимовым), за что им огромная признательность. Автор считает, что применение разнообразных по содержанию задач дает гораздо больший эффект, чем только лекционный курс и изучение многочисленных учебников.

В списке литературы приведены книги, использованные при подготовке сборника. Они могут не только быть полезны как источник дополнительных задач, но и иметь самостоятельное практическое значение, так как многие из них содержат хороший материал по технологии программирования.

К задачам при их подборе предъявлялись следующие требования:

- **типичность**, то есть применение в решении часто используемых приемов программирования;
- **содержательность**, то есть формулировка задач в терминах предметной области, с предложением самостоятельно провести математизацию задачи (вывести необходимые формулы) и алгоритмизацию (найти или разработать метод решения);
- **нетривиальность**, то есть минимум похожих задач, решаемых по одному алгоритму.

Таким образом, сделана попытка максимально охватить многообразие приемов алгоритмизации и программирования, дать возможность поупражняться в поиске эффективных алгоритмов. Многие задачи представлены в вербальной постановке, требующей предварительной формализации и алгоритмизации.

Часть I задачника содержит простейшие задачи для начинающих. Часть II посвящена типичным приемам при работе с массивами (то есть манипуляциям данными в оперативной памяти): линейному поиску, преобразованиям. В части III собраны разнообразные прикладные задачи: от геометрии и комбинаторики до элементов текстового анализа и компьютерной мультипликации, а также методы сортировки и задачи, приводящие к переборным и рекурсивным алгоритмам. Часть IV задачника посвящена базам данных и электронным таблицам. При этом задачи разбиты по темам так, чтобы в каждой теме было не менее 25–30 задач, что дает возможность преподавателю случайно или принудительно распределять их среди студенческой группы.

Задачи, как правило, не требуют математических или других специальных знаний: для решения большинства из них достаточно школьного курса и лишь в некоторых используются элементы аналитической геометрии, векторной алгебры, математического анализа либо других математических дисциплин (в последнем случае необходимые сведения приведены в условии задачи либо в ссылках на литературу). Это позволяет использовать приведенные задачи не только для математических, но и для других естественнонаучных и технических специальностей, а выборочно — для «гуманитариев» и школьников.

Задачи максимально свободны от привязки к языку программирования. Так, первые три части — классическое программирование — реализуются на любой версии Бейсика, Паскаля, С, Фортрана, Delphi и др. (при наличии разве что графических средств). Четвертая часть ориентирована на СУБД типа xBase, FoxPro, Delphi, Access (программы могут быть реализованы на любой другой СУБД), а также на электронные таблицы типа SuperCalc, Excel и др.

Задачи были испытаны более чем на десяти потоках студентов естественнонаучных специальностей АГУ; автор выражает им свою признательность за долготерпение и помощь в уточнении формулировок, в оценке сложности задач, в разработке некоторых примеров программ (отметим, что приведенные примеры реализованы в среде Turbo Pascal 7.0 фирмы Borland). Автор также глубоко признателен коллегам, участвовавшим в этой «обкатке», и конкретно — А. Максимову, Л. Смоляковой, Д. Иванову и др., — за помощь в подборе задач и устранении недостатков формулировок. Некоторые из студентов уже стали известными программистами и с благодарностью вспоминают первые навыки, полученные с помощью этих задач.

Несколько слов к преподавателям об использовании предлагаемой подборки в учебном процессе. Для первого курса выделяются первые 16 тем (случайным образом сформированные индивидуальные пакеты заданий на весь учебный год выдаются студентам в сентябре; дальше — самостоятельная работа с консультациями преподавателя и защита по мере реализации и испытания программ). Разная стартовая подготовка абитуриентов позволяет формировать индивидуальные требования, по результатам решения выставлять досрочные оценки либо назначать дополнительные консультации. На старте же оглашаются критерии оценок (например, для получения удовлетворительной оценки за год без устного экзамена достаточно своевременно решить 9 задач; результаты на экзамене прибавляются к накопленной сумме баллов, и оценка выставляется по итоговой сумме). При затягивании сроков защиты вклад задач в экзаменационную оценку уменьшается; при повышенном качестве реализации алгоритма — увеличивается.

Однако опыт показал, что, несмотря на все усилия, задачи не удалось сбалансировать по сложности в пределах одной темы. Поэтому была сделана попытка проранжировать сложность задач в баллах и изменить критерии итоговых оценок. Уровень сложности задач первых 16 тем оценен в баллах («вес» каждой задачи проставлен в скобках после ее номера). При этом ранжирование проведено относительное, в пределах каждого раздела. Разделы расположены, как правило, в порядке возрастания сложности задач, что подразумевает рост квалификации обучаемых при переходе от темы к теме. При оценке «весов» учитывались мнения как преподавателей, так и студентов. Хотелось бы получить критические отзывы коллег на нашу технологию.

Для второго курса предлагаются задачи из 17-й темы (близкий к реальным условиям проект информационно-поисковой системы (ИПС) вместе с умением описывать программные средства в среде Microsoft Office или на других платформах), а также знакомство с электронными таблицами — в 18-й теме.

Типографское соглашение: в условиях задач, если это не оговорено явно, БОЛЬШИМИ латинскими буквами обозначены агрегаты данных (массивы, структуры, множества, геометрические точки), малыми буквами — простые (скалярные) переменные. При этом, как правило, буквами *I*, *J*, *K*, *L*, *M*, *N* обозначаются целочисленные агрегаты; буквами *i*, *j*, *k*, *l*, *m*, *n* — целочисленные переменные. Размерности массивов записываются в круглых скобках через запятую, например *A(m,n)* — матрица, содержащая *m* строк и *n* столбцов.

От издательства

Ваши замечания, предложения, вопросы отправляйте по адресу электронной почты comp@piter.com (издательство «Питер», компьютерная редакция).

Мы будем рады узнать ваше мнение!

Подробную информацию о наших книгах вы найдете на web-сайте издательства http://www.piter.com.

О технологии программирования

Испытание, тестирование, отладка, защита разработки

В разработке программ традиционно выделяют следующие этапы: формализация постановки (перевод задачи на язык математической символики), выбор либо разработка методов решения, разработка алгоритма, кодирование (перевод алгоритма на алгоритмический язык в соответствии с его синтаксисом), тестирование, отладка, защита, сдача в эксплуатацию, сопровождение (авторский надзор). Формализация и алгоритмизация часто носят нестандартный характер и не допускают единых рецептов; внедрение и сопровождение — процессы, выходящие за рамки настоящего задачника. Остановимся подробнее на отладке и тестировании.

Отладка — процесс выявления, локализации и устранения ошибок в алгоритме и реализующей его программе — осуществляется с помощью тестирования. Не будем останавливаться на выявлении синтаксических ошибок — это с разной эффективностью реализуется трансляторами; речь пойдет о программе, которая благополучно транслируется, принимает исходные данные и выдает (хоть какие-то) результаты. Задача — убедиться в корректности алгоритма, то есть получить уверенность, что программа выдает результаты, соответствующие задаче и исходным данным. Показано, что посредством испытаний либо теоретически невозможно доказать правильность программы; можно лишь спровоцировать появление и устранить в ней как можно больше ошибок.

Тест — совокупность исходных данных для программы вместе с ожидаемыми результатами (с учетом формы представления последних). Тесты разрабатываются **до**, а не **вовремя** или **после** разработки программы, дабы избежать провокационного влияния стереотипов алгоритма на тестирование. Готовится не один тест, а их совокупность — *набор тестов*, призванный охватить максимум ситуаций. Испытание программы проводится сразу на всем наборе с протоколированием и анализом результатов.

Набор тестов называется *полным*, если он позволяет активизировать все ветви алгоритма. Набор тестов назовем *не избыточным*, если удаление из него любого теста лишает его полноты. Таким образом, искусство тестирования сводится к разработке полного и не избыточного набора тестов, а технология — к испытанию программы на всем наборе после внесения в нее каждого исправления. Удачно подобранные тесты позволяют не только констатировать факт наличия ошибок, но и *локализовать* их, то есть найти место в программе, виновное в получении неверных результатов.

В наборе тестов выделяют три группы:

- «тепличные» — проверяющие программу при корректных, нормальных исходных данных самого простого вида;
- «экстремальные» — на границе области определения, в ситуациях, которые могут произойти и на которые нужно корректно реагировать;
- «запредельные» — за границей области определения (а, возможно, и здравого смысла) — ситуации, бессмысленные с точки зрения постановки задачи, но которые могут произойти из-за ошибок пользователя или программ, поставляющих исходные данные для тестируемой программы.

Требование надежности программирования: принимать данные, если они корректны, и получать для них правильные результаты либо отвергать их как некорректные, по возможности с анализом некорректности.

Подготовка тестов может оказаться довольно трудоемкой работой; но в некоторых случаях этот процесс можно автоматизировать. Так, некоторые задачи некритичны к содержимому входных массивов; правильность результата можно проверить визуальным сопоставлением с исходными данными, каковы бы они ни были. Один из приемов — *рандомизация* — подготовка случайных исходных данных.

Практически во всех трансляторах реализован *датчик случайных чисел*, — как правило, стандартная функция с именем Random, возвращающая случайное число из интервала $[0;1)$. Если такого датчика нет, можно воспользоваться алгоритмом, описанным в разделе 10. С помощью простейшего датчика можно получать практически любые распределения. Так, если ξ равномерно распределено на $[0;1)$, то выражение $x=(b-a)\xi$ дает действительное число, равномерно распределенное на отрезке $[a;b)$, а конструкция на Паскале `k:=Ord(p>Random)` присваивает переменной `k` единицу с вероятностью p и нуль — с вероятностью $(1-p)$. Аналогично можно генерировать монотонные (возрастающие или убывающие) последовательности (например обращение в цикле: `a[i]:=a[i-1]+Random` позволяет получить случайный неубывающий массив), а также «замусоренные» массивы — частично искаженные случайным шумом.

Другой прием — программная подготовка тестов в виде массивов или файлов существенной размерности, обладающих заданными свойствами, с известным ожидаемым результатом основной программы. Для этого необходимо написать специальную программу — генератор тестов, но это менее трудоемко, чем подготовка тестов вручную.

Таким образом, защита разработки сводится к демонстрации работоспособности программы на совокупности тестов, а именно к совместной защите набора тестов, алгоритма и программы.

Эффективность алгоритмов и программ

Эффективность алгоритма определяется потребляемыми ресурсами компьютера, а именно *быстродействием* (по количеству выполняемых операций, с учетом трудоемкости каждой из них, то есть в конечном итоге времени решения задачи определенной размерности) и общим *объемом оперативной* *памяти*, выделяемой (запрашиваемой) под данные. Эти показатели порой противоречивы: повышение быстродействия может потребовать дополнительных расходов памяти, либо наоборот. Если можно улучшить один показатель без ущерба для другого, следует этого добиваться; при возникновении же дилеммы в современных условиях следует отдавать предпочтение экономии памяти в ущерб производительности, так как тактовая частота компьютеров растет опережающими темпами в сравнении с объемом оперативной памяти.

Следует иметь в виду, что алгоритмы и программы отлаживаются и испытываются, как правило, на модельных примерах небольшой размерности, так что на современных компьютерах не удается почувствовать их трудоемкости. Однако всегда подразумевается использование разработанных программ на реальных задачах существенной размерности, где показатели эффективности становятся уже критичными. В то же время некоторые задачи, имеющие факториальную зависимость трудоемкости от размерности, в состоянии «подвесить» любой современный компьютер даже на маленьких задачах, так что проблема эффективности не снимается с ростом производительности компьютеров.

Приведем ряд мер, которые можно рекомендовать для повышения эффективности.

1. Не использовать рабочие массивы того же порядка размерности, что и обрабатываемый или создаваемый, если это возможно. При обработке двумерного массива допустимо выделение одномерного рабочего массива для временного хранения строки или столбца матрицы.
2. Выбирать, где это возможно, короткие типы данных: Byte вместо Word; String[20] вместо String и т. д.
3. Использовать поименованные константы вместо неоднократного повторения констант-«близнецов».
4. При обращении к процедурам (особенно рекурсивным) параметры передавать преимущественно по адресу, а не по значению. Переменные, используемые в процедурах как рабочие, объявлять локальными, а не глобальными.

5. Выбирать алгоритмы, эффективные по числу операций, оценив предварительно порядок зависимости трудоемкости от размерности (логарифмическая, линейная, полиномиальная, факториальная и т. д.).
6. Избегать вычислений в циклах выражений, не зависящих от параметра цикла (например, `Sin(Pi/n)`, где `n` не меняется в цикле) имея в виду, что трансцендентные функции вычисляются трудоемким разложением в ряд.
7. Прекращать вычисления, когда результат достигнут, либо очевидно, что он не может быть достигнут за приемлемое время. Для этого использовать циклы типа `while ... do` или `repeat ... until` вместо `for ... do`. Не рекомендуется обращаться к средствам принудительного завершения типа `GoTo`, `Halt`, `Break` и др., так как они сильно снижают наглядность программы и часто приводят к неожиданным последствиям.
8. Выбирать, где это возможно, наименее трудоемкие операции. Так, например, выражение `n div k` вычисляется быстрее, чем `Trunc(n/k)`, к тому же дает гарантированно точный результат; `Ord(Odd(n))` более эффективно, чем `n mod 2`.

Часть I
Задания для начинающих

В части I приведены задачи, предназначенные для начинающих изучать программирование. Эти задачи требуют знания основных конструкций алгоритмического языка и предусматривают использование типичных технологических приемов программирования без учета особенностей конкретной версии транслятора.

Многие задачи имеют не единственное решение. При этом критериями качества программы служат следующие показатели (по убыванию важности):

- оригинальность решения;
- объем памяти, занимаемой программой (с учетом памяти, отводимой под массивы);
- трудоемкость вычислений, то есть эффективность алгоритма;
- лаконичность и наглядность программы, включая наличие и качество комментариев.

Если исходные данные задачи или размерности массивов не заданы, следует разработать полную совокупность тестовых данных.

Приветствуется обобщение постановки задачи, то есть замена частного случая более общим.

1. Линейные алгоритмы

Задачи этой темы сводятся к разработке и программированию простых линейных алгоритмов, поэтому в решении не стоит использовать операторы цикла и массивы.

Многие задачи даны в содержательной постановке, то есть в терминах той предметной области, которой они обязаны своим происхождением. Для решения требуется провести математическую и алгоритмическую постановки, то есть вывести (или найти в литературе) необходимые формулы и разработать алгоритм.

Во многих задачах подразумевается использование целочисленной арифметики: вычисление целого частного от деления и остатка: операции div и mod в Паскале и др., а также вычисления с процентами, которым в средней школе уделяется недостаточно внимания.

Пример. Правительство гарантирует, что инфляция в новом году составит $p\%$ в месяц. Какого роста цен за год можно ожидать?

Решение. Простое произведение $12p\%$ не будет верным, нужно найти так называемые сложные проценты. Если за каждый месяц цены возрастут в $1+p/100$ раз, то за год рост цен составит $(1+p/100)^{12}$ раз, или прирост в процентах:

$$s = \left[\left(1+\frac{p}{100}\right)^{12} - 1\right] \cdot 100\%.$$

Программа на Паскале в этом случае может иметь следующий вид.

```
Program Super_Procent (Input, Output);
   var a, p, s: Real;
begin
Write ('Введите процент месячной инфляции ');
   ReadLn (p);
   a:= Exp(Ln(1+p/100)*12);
         {кратность роста    цен}
{Так как в Паскале нет возведения в степень, используется
тождество: a^b = exp(ln(a)*b)}
   s:=(a-1)*100;
   Writeln ('Годовой рост цен составит ', a:10:2,
         ' раз, или' s:10:2,' процентов');
end.
```

Для испытания программы можно взять тесты (p – процент месячной инфляции; s – результат – годовой прирост цен в процентах):

- p = 0%; s = 0%;
- p = 1%; s = 12,68%;
- p = 10%; s = 213,84%.

Задачи по теме «Линейные алгоритмы»

1.1 (6 б.) **Из градусов в радианы.** Угол α задан в градусах, минутах и секундах. Найти его величину в радианах (с максимально возможной точностью).

Тестирование: рекомендуется проверить работоспособность программы для углов, больших развернутого, а также для отрицательных углов.

1.2 (7 б.) **Из радианов в градусы.** Решить задачу, обратную предыдущей, то есть перевести заданную величину угла из радианной меры в градусную.

1.3 (7 б.) **Из дюймов в метры.** Длина отрезка задана в дюймах (1 дюйм = 2,54 см). Перевести значение длины в метрическую систему, то есть выразить ее в метрах, сантиметрах и миллиметрах. Так, например, 21 дюйм = 0 м 53 см 3,4 мм.

1.4 (7 б.) **Временной интервал.** Заданы моменты начала и конца некоторого промежутка времени в часах, минутах и секундах (в пределах одних суток). Найти продолжительность этого промежутка в тех же единицах измерения.

1.5 (8 б.) **Такси.** В такси одновременно сели три пассажира. Когда вышел первый пассажир, на счетчике было p_1 рублей; когда вышел второй — p_2 рублей. Сколько должен был заплатить каждый пассажир, если по окончании поездки счетчик показал p_3 рублей? Плата за посадку составляет p_0 рублей.

Тестирование: общая сумма оплаты пассажирами должна совпадать с показанием счетчика по окончании поездки. Рассмотрим крайние ситуации. По справедливости, если все три пассажира вышли одновременно, они должны заплатить по $(p_0 + p_3)/3$ руб. Если же первый и второй пассажиры «передумали ехать», они платят по $p_0/3$ руб., а оставшаяся сумма ложится на счет третьего пассажира.

1.6 (7 б.) **Коммерция.** Коммерсант, имея стартовый капитал k рублей, занялся торговлей, которая ежемесячно увеличивает капитал на p%. Через сколько лет он накопит сумму s, достаточную для покупки собственного магазина?

1.7 (7 б.) **Селекция.** Селекционер вывел новый сорт зерновой культуры и снял с опытной делянки k кг семян. Посеяв 1 кг семян, можно за сезон собрать p кг семян. Через сколько лет селекционер сможет засеять новой культурой поле площадью s га, если норма высева n кг/га?

1.8 (8 б.) **Среднегодовая производительность труда.** За первый год производительность труда на предприятии возросла на p_1 %, за второй и третий — соответственно на p_2 и p_3 %. Найти среднегодовой прирост производительности (в процентах).

Тестирование, алгоритмизация: если ежегодный прирост постоянен, то и среднегодовой прирост p такой же: $p = p_1 = p_2 = p_3$. Общий прирост за 3 года в общем случае составит

$$p_\Sigma = \left[\left(1+\frac{p_1}{100\%}\right)\left(\cdot 1+\frac{p_2}{100\%}\right)\left(1+\frac{p_3}{100\%}\right)-1\right]\cdot 100\%.$$

Тот же результат можно получить при среднегодовом приросте p:

$$p_\Sigma = \left[\left(1+\frac{p}{100\%}\right)^3 -1\right]\cdot 100\%.$$

Остается найти величину p.

1.9 (5 б.) **Кубическое уравнение.** Заданы три корня кубического уравнения: x_1, x_2, x_3. Найти коэффициенты этого уравнения.

1.10 (5 б.) **Квадратное уравнение.** Найти корни квадратного уравнения, заданного своими коэффициентами, с положительным дискриминантом; подстановкой в уравнение убедиться в погрешности вычислений.

1.11 (5 б.) **Комплексное число.** Заданы действительная и мнимая части комплексного числа $z = x + iy$. Преобразовать его в тригонометрическую форму и напечатать в виде выражения: $z = r(\cos\varphi + i\sin\varphi)$.

Для справки: $r = \sqrt{x^2 + y^2}$; $\varphi = \operatorname{arctg}\frac{y}{x}$.

1.12 (8 б.) **Экстремальные точки маятника.** Заданы координаты точки подвески математического маятника $A(x_0, y_0, z_0)$ и координаты одной из точек его наивысшего подъема $B(x_1, y_1, z_1)$. Найти координаты самой низкой точки траектории и другой наивысшей точки подъема.

1.13 (6 б.) **Пересекающиеся прямые.** Заданы уравнения двух пересекающихся прямых на плоскости: $y = k_1 + b_1$; $y = k_2 + b_2$. Найти (в градусах и минутах) угол между ними, используя формулу: $\operatorname{tg}\varphi = \frac{(k_2 - k_1)}{(1 + k_1 k_2)}$.

1.14 (8 б.) **Русские неметрические единицы длины:** 1 верста = 500 саженей; 1 сажень = 3 аршина; 1 аршин = = 16 вершков; 1 вершок = 44,45 мм. Длина некоторого отрезка составляет p метров. Перевести ее в русскую неметрическую систему.

1.15 (9 б.) **«Косой» квадрат.** У квадрата $ABCD$ на плоскости известны координаты двух противоположных вершин — точек A и C. Найти координаты точек B и D.

Примечание. Расположение квадрата произвольно; его стороны не обязательно параллельны координатным осям.

1.16 (5 б.) **Стороны — по высоте.** В равнобедренном прямоугольном треугольнике известна высота h, опущенная на гипотенузу. Найти стороны треугольника.

1.17 (5 б.) **Длина высоты.** Треугольник ABC задан длинами своих сторон. Найти длину высоты, опущенной из вершины A.

Экстремальные тесты: сумма двух сторон равна третьей; одна из сторон равна нулю.

1.18 (7 б.) **Задача жестянщика.** Из круга радиуса r вырезан прямоугольник, большая сторона которого равна a. Найти максимальный радиус круга, который можно вырезать из полученного прямоугольника?

Экстремальные тесты: $a = 2r$; $a = r\sqrt{2}$.

1.19 (5 б.) **Вершина параболы.** Найти координаты вершины параболы $y = ax_2 + bx + c$.

1.20 (5 б.) **Приближение $\sin x$.** Функция $y = \sin x$ на отрезке $[0;\pi/2]$ хорошо аппроксимируется разложением: $y = x - x^3/6 + x^5/120$. Для заданного значения аргумента x вычислить y по этой формуле и сравнить с точным значением, вычисленным с помощью стандартной функции Sin.

1.21 (10 б.) **Движение без топлива?** Владелец автомобиля приобрел новый карбюратор, который экономит 50% топлива, новую систему зажигания, которая экономит 30% топлива, и поршневые кольца, экономящие

20% топлива. Верно ли, что его автомобиль теперь сможет обходиться совсем без топлива? Найти фактическую экономию для произвольно заданных сэкономленных процентов.

Задачи 1.22–1.27 посвящены **«решению треугольников»**. Треугольник задается координатами своих вершин на плоскости: $A(x_1, y_1)$, $B(x_2, y_2)$, $C(x_3, y_3)$.

1.22 (5 б.) Найти площадь треугольника ABC.

1.23 (6 б.) Найти сумму длин медиан треугольника ABC.

1.24 (7 б.) Найти точку пересечения биссектрис треугольника ABC (центр вписанной в него окружности).

1.25 (7 б.) Найти внутренние углы треугольника ABC (в градусах).

1.26 (7 б.) Найти длину и основание высоты, опущенной из вершины A на сторону BC.

1.27 (8 б.) Найти точку D, симметричную точке A относительно стороны BC.

Векторная алгебра. В задачах 1.28–1.31 трехмерные векторы заданы своими координатами, например, $A = (x_a, y_a, z_a)$. Требуется найти геометрические характеристики, определяющие их параметры и взаимное расположение.

1.28 (7 б.) Найти угол (в градусах) между векторами A и B, используя формулу:

$$\cos\varphi = \frac{(A,B)}{|A|\cdot|B|} = \frac{x_a x_b + y_a y_b + z_a z_b}{\sqrt{x_a^2 + y_a^2 + z_a^2}\cdot\sqrt{x_b^2 + y_b^2 + z_b^2}}.$$

1.29 (7 б.) Найти объем пирамиды, построенной на векторах A, B, C, как на сторонах.

1.30 (6 б.) Найти длину диагонали параллелепипеда, построенного на векторах A, B, C, как на сторонах.

1.31 (7 б.) На тело действуют две силы, заданные векторами A и B. Найти величину и направление (углы с координатными осями) их равнодействующей.

Замечание. Углы с координатными осями вектора $A = (x_a, y_a, z_a)$ можно найти, используя *направляющие косинусы:*

$$\cos\alpha = \frac{x_a}{|A|} = \frac{x_a}{\sqrt{x_a^2 + y_a^2 + z_a^2}}, \ \cos\beta = \frac{y_a}{|A|}, \ \cos\gamma = \frac{z_a}{|A|}.$$

1.32 (7 б.) **Округленное время.** Текущее время (часы, минуты, секунды) задано тремя переменными: h, m, s. Округлить его до целых значений минут и часов. Например, 14 ч 21 мин 45 с преобразуется в 14 ч 22 мин или 14 ч, а 9 ч 59 мин 23 с — соответственно в 9 ч 59 мин или 10 ч.

1.33 (8 б.) **Цена на молоко.** Животновод в начале каждой зимы повышает отпускную цену на молоко на p%, а каждым летом — снижает на столько же процентов. Изменится ли цена на молоко и если да, то в какую сторону и на сколько через n лет?

1.34 (7 б.) **Переправа.** Чапаеву надо под прямым углом к фарватеру преодолеть реку Урал шириной b м. Его скорость в стоячей воде v_1 м/с; скорость течения реки — v_2 м/с. Под каким углом к фарватеру он должен плыть, чтобы его «не снесло»? Сколько времени займет переправа? Как изменится решение, если посредине реки Чапаева ранили в руку, и его скорость с v_1 м/с упала до v_3 м/с?

1.35. (9 б.) **Русская пирамида-1.** Сколько кругов заданного радиуса r можно вырезать из правильного треугольника со стороной a?

1.36 (9 б.) **Русская пирамида-2.** Какова должна быть длина стороны правильного треугольника a, чтобы из него можно было вырезать n кругов радиуса r?

2. Разветвляющиеся алгоритмы

Задачи этой темы посвящены использованию условных операторов; следует в решениях обойтись без циклов и массивов. Применение операторов ветвления позволяет использовать простейшую защиту программы от сбоев: контроль входных данных и промежуточных результатов.

Пример. Факультету выделен стипендиальный фонд в размере f р./мес. Результаты сессии таковы: n_1 «отличников», n_2 «хорошистов», n_3 «троечников». Повышенная стипендия (для отличников) составляет s_1 р., обычная — s_2 р.; задолжники стипендии лишаются. Сколько студентов каждой категории могут получать стипендию и каков будет остаток фонда на материальную помощь малоимущим?

Решение можно записать в виде следующей Паскаль-программы:

```
Program Stipen (Input. Output);
     var n1. n2. n3. k1. k2. k3. s. s1. s2.
          fond: Integer;
begin
     Write ('Каков размер фонда? ');
     ReadLn (fond);
     Write ('Сколько в категориях '.
          '"отл."."хор."."удовл."? ');
     ReadLn (n1. n2. n3);
     Write ('Размер повышенной и нормальной '.
               'стипендий: ');
     ReadLn (s1. s2);
     if fond > s1*n1 then k1 := n1
          else k1 := fond div s1;
     fond := fond - s1*k1;
```

```
        {остаток для остальных категорий}
    if fond > s2*n2 then k2 := n2
        else k2 := fond div s2;
    fond := fond - s2*k2;
              {остаток для троечников}
    if fond > s2*n3 then k3 := n3
        else k3 := fond div s2;
    fond := fond – s2*k3; {остаток - резерв}
    WriteLn ('Выдавать стипендию:');
    WriteLn (k1,' отличникам;');
    if k2 > 0 then
        WriteLn (k2,' хорошистам;');
    if k3 > 0 then
        WriteLn (k3,' троечникам:');
    WriteLn ('Резерв фонда составит ',
                      fond,' рублей')
end.
```

Задачи по теме «Разветвляющиеся алгоритмы»

Свойства и виды треугольников (задачи 2.1–2.4)

2.1 (5 б.) Заданы три числа: a, b, c. Определить, могут ли они быть сторонами треугольника, и если да, то определить его тип: равносторонний, равнобедренный, разносторонний.

Замечание. Условия существования треугольника:

$a \le b+c$; $b \le a+c$; $c \le a+b$.

Нельзя исключать экстремальных случаев, когда одна (или несколько) сторон равны нулю либо когда одно из неравенств переходит в равенство (треугольник нулевой площади).

2.2 (6 б.) Треугольник задан длинами своих сторон: a, b, c. Определить, является ли он тупоугольным, прямоугольным или остроугольным.

Замечание. Достаточно, используя теорему косинусов, найти **знаки косинусов** внутренних углов треугольника, не вычисляя самих углов (они могут быть нулевыми или развернутыми).

2.3 (6 б.) Треугольник задан координатами своих вершин на плоскости: $A(x_a, y_a)$, $B(x_b, y_b)$, $C(x_c, y_c)$. Определить, является он прямо-, остро- или тупоугольным.

Замечание. Не следует отбрасывать экстремальные случаи, когда вершины треугольника совпадают или лежат на одной прямой. Например, треугольник с нулевой стороной обладает свойством прямоугольного и имеет два прямых угла!

2.4 (7 б.) Числа *a*, *b*, *c* тогда и только тогда являются сторонами треугольника, когда существуют такие положительные *x*, *y*, *z*, что

$$\begin{cases} a = x + y; \\ b = y + z; \\ c = x + z. \end{cases}$$

Свойства и виды четырехугольников (задачи 2.5, 2.6)

2.5 (8 б.) Четырехугольник *ABCD* задан координатами своих вершин на плоскости: $A(x_a, y_a)$, $B(x_b, y_b)$, $C(x_c, y_c)$, $A(x_a, y_a)$, $D(x_d, y_d)$. Проверить, является ли он выпуклым.

Замечание. Есть несколько способов проверки выпуклости: анализ линейных неравенств, задаваемых сторонами; разбиение четырехугольника на треугольники со сравнением сумм их площадей и другие.

2.6 (8 б.) Четырехугольник *ABCD* задан координатами своих вершин на плоскости: $A(x_a, y_a)$, $B(x_b, y_b)$, $C(x_c, y_c)$, $A(x_a, y_a)$, $D(x_d, y_d)$. Определить тип четырехугольника: прямоугольник, параллелограмм, трапеция, произвольный четырехугольник. Учесть погрешность вычислений.

Замечание. Для устранения дополнительных источников погрешности рекомендуется использовать аппарат векторной алгебры: коллинеарность, равенство и ортогональность векторов — сторон четырехугольника.

2.7 (6 б.) **Треугольник и круги.** Лежит ли заданный на плоскости треугольник ABC в области пересечения заданных кругов: $(x-a_1)^2+(y-b_1)^2 \le r_1^2$; $(x-a_2)^2+(y-b_2)^2 \le r_2^2$?

2.8 (7 б.) **Кирпич.** Пройдет ли кирпич со сторонами a, b и c сквозь прямоугольное отверстие со сторонами r и s? Стороны отверстия должны быть параллельны граням кирпича.

2.9 (6 б.) **Шар и ромб.** Может ли шар радиуса r пройти через ромбообразное отверстие с диагоналями p и q?

2.10 (7 б.) **Посылка.** Можно ли коробку размером $a \times b \times c$ упаковать в посылку размером $r \times s \times t$? «Углом» укладывать нельзя.

2.11 (8 б.) **Задача жестянщика.** Можно ли из круглой заготовки радиуса r вырезать две прямоугольные пластинки с размерами $a \times b$ и $c \times d$?

2.12 (8 б.) **Планировка.** Можно ли на прямоугольном участке застройки размером a на b метров разместить два дома размером в плане p на q и r на s метров? Дома можно располагать только параллельно сторонам участка.

2.13 (7 б.) **Две окружности.** Проверить, лежит ли окружность $(x-a_1)^2+(y-b_1)^2=r_1^2$ целиком внутри окружности $(x-a_2)^2+(y-b_2)^2=r_2^2$ или наоборот.

2.14 (7 б.) **Треугольник и точка.** Лежит ли точка $M(x_m, y_m)$ внутри треугольника, заданного координатами своих вершин $A(x_A, y_A)$, $B(x_B, y_B)$, $C(x_C, y_C)$ на плоскости?

2.15 (10 б.) **Общая точка.** Два отрезка на плоскости заданы координатами своих концов. Определить, имеют ли эти отрезки общие точки.

Замечание. Необходимо рассмотреть различные случаи взаимной ориентации отрезков: на одной прямой, на параллельных или пересекающихся прямых.

Тестирование должно предусмотреть все такие ситуации.

2.16 (6 б.) **Кратные пары.** Среди заданных целых чисел k, l, m найти пары кратных.

2.17 (5 б.) **Деление на 3.** Как известно, число делится на 3 тогда и только тогда, когда сумма его цифр делится на 3. Проверить этот признак на примере заданного трехзначного числа.

Замечание. Теоретическое утверждение о признаке делимости предлагается проверить на примере любого вводимого числа. Признак считается доказанным, но не будет лишним поиск для него контрпримеров.

2.18 (6 б.) **Ориентация.** Заданы координаты вершин треугольника ABC на плоскости. Вывести их в порядке обхода по часовой стрелке (для проверки достаточно рассмотреть знаки внутренних углов).

2.19 (7 б.) **Привал.** Путник двигался t_1 часов со скоростью v_1, затем t_2 часов — со скоростью v_2 и t_3 часов — со скоростью v_3. За какое время он одолел первую половину пути, после чего запланировал привал?

2.20 (7 б.) **Как успеть подешевле?** Можно ехать на такси со скоростью v_1 км/ч и оплатой p_1 р./км либо идти пешком со скоростью v_2 км/ч бесплатно. Как с наименьшими затратами преодолеть путь s за время t, если это возможно? Каковы эти затраты?

Тестирование. Рекомендуется рассмотреть «запредельные» случаи: когда времени слишком мало, чтобы успеть даже на такси, либо слишком много, так что и пешком можно с запасом успеть до отхода поезда.

2.21 (10 б.) **Задача о смесях.** Имеются три раствора полезного вещества с концентрациями p_1, p_2 и p_3 каждый и стоимостью s_1, s_2 и s_3 соответственно. Можно ли смешать их так, чтобы получить раствор с заданной концентрацией p наименьшей стоимости?

Указание. Пусть $\alpha_1, \alpha_2, \alpha_3$ — долевые содержания растворов в смеси. Тогда для получения заданной концентрации p необходимо: $p_1\alpha_1 + p_2\alpha_2 + p_3\alpha_3 = p$.

Кроме того, нужно учесть условие «комплектности» смеси:

$$\alpha_1 + \alpha_2 + \alpha_3 = 1; \quad \alpha_1 \geq 0; \quad \alpha_2 \geq 0; \quad \alpha_3 \geq 0.$$

При этих условиях необходимо найти наименьшее значение линейной функции: $s = s_1\alpha_1 + s_2\alpha_2 + s_3\alpha_3 \to \min\limits_{\alpha_1,\alpha_2,\alpha_3}$.

С учетом ограничений задача сводится к минимизации линейной функции одного переменного на отрезке, однако искомые выражения и условия получаются достаточно громоздкими. Можно показать, что в решении будут участвовать не более двух растворов. Тогда достаточно среди вариантов:

а) $\alpha_1 = 0$; б) $\alpha_2 = 0$; в) $\alpha_3 = 0$.

выбрать оптимальный, и затем провести необходимые расчеты.

2.22 (7 б.) **Голодная зима.** Суточный рацион коровы составляет u кг сена, v кг силоса и w кг комбикорма. В хозяйстве, содержащем стадо из k голов, осталось s центнеров сена, t тонн силоса и f мешков комбикорма по 50 кг. Сколько еще дней хозяйство сможет кормить коров по полному рациону? Какой из кормов кончится раньше других?

2.23 (7 б.) **Факультеты.** В Алтайском госуниверситете принято, что старшая цифра номера студенческой группы означает номер факультета, средняя — последнюю цифру года поступления, младшая — порядковый номер группы на курсе. Продолжительность обучения — не более 6 лет (магистратура). Дан номер группы студента АГУ и текущий год. Напечатать, в каком году он поступил и на каком факультете учится. Например, гр. **432**, **1996** г. — факультет **математический**, год поступления **1993**. Для справки приведены номера факультетов:

1. исторический;
2. экономический;
3. юридический;
4. математический;
5. физический;

6. химический;
7. биологический;
8. филологический;
9. географический;
10. социологический.

Тестирование. Предусмотреть невозможные ситуации, например, гр. 521, год 2001.

Шахматы и шашки. В задачах 2.24–2.27 позицию каждой шахматной фигуры или шашки можно задавать в обычной нотации (например, *d7*) или парой чисел — координат фигуры (например, 4;7). При тестировании полезно проверить алгоритм на недопустимых ситуациях, когда несколько фигур стоят на одном поле.

2.24 (8 б.) На шахматной доске стоят черный король и три белые ладьи (ладья бьет по горизонтали и вертикали). Проверить, не находится ли король под боем, а если есть угроза, то от кого именно.

2.25 (8 б.) На шахматной доске стоят черный король и белые ладья и слон (ладья бьет по горизонтали и вертикали, слон — по диагоналям). Проверить, есть ли угроза королю и если есть, то от кого именно. Учесть возможность защиты (например, ладья не бьет через слона).

2.26 (8 б.) На шахматной доске стоят три ферзя (ферзь бьет по вертикали, горизонтали и диагоналям). Найти те пары из них, которые угрожают друг другу.

2.27 (10 б.) В шашечном эндшпиле остались белая дамка и две черных пешки, позиции которых известны. Ход белых. Сможет ли дамка срубить одну или сразу обе пешки?

2.28 (8 б.) **Вклад.** Банк предлагает 3 вида срочных вкладов: на 3 месяца под p_1%, на 6 месяцев под p_2% и на год под p_3%. Какой из вкладов наиболее выгоден для вкладчика?

2.29 (6 б.) **Мой возраст.** Для заданного $0 \le n \le 200$, рассматриваемого как возраст человека, вывести фразу вида: «Мне 21 год», «Мне 32 года», «Мне 12 лет».

2.30 (8 б.) **Отрезки на плоскости.** Найти расстояние между двумя произвольно заданными на плоскости отрезками. Определение расстояния между множествами геометрических точек можно найти например, во введении к теме 8.

2.31 (7 б.) **Встреча.** Из пункта *A* в пункт *B* выехал велосипедист со скоростью v_0 км/час. Одновременно навстречу ему из пункта *B* двинулся «автостопом» другой путник. s_1 м он двигался со скоростью v_1 м/час, s_2 м — со скоростью v_0 км/час, s_3 м — со скоростью v_3 км/час. Через сколько часов после старта и в какой точке путники встретились?

2.32 (6 б.) **Треугольник из круга.** Из круга какого наименьшего радиуса можно вырезать треугольник со сторонами a, b, c?

Указание. Пусть c — большая из сторон треугольника. Если угол C — тупой, сторона c совпадает с диаметром круга, и его радиус: $r = c/2$. В противном случае имеем описанную окружность:

$$r = \frac{abc}{4\sqrt{p(p-a)(p-b)(p-c)}},$$

где $p = (a + b + c)/2$ — полупериметр треугольника.

3. Циклические и итерационные алгоритмы

В задачах этого раздела реализуется тот или иной циклический процесс, который выполняется либо за заранее известное число шагов, либо до достижения некоторого условия (итерационные алгоритмы). В последнем случае полезно подстраховаться от зацикливания («вечного цикла»), которое может возникнуть из-за разных ошибок в программе и алгоритме, из-за некорректных данных либо вследствие накопления погрешностей. Для этого (хотя бы на этапе отладки) достаточно поставить лимит числа шагов (с выдачей сообщения в случае его исчерпания).

В итерационных алгоритмах заданная погрешность используется для проверки модуля разности найденного приближенного и точного значений, однако если последнее неизвестно, допустимо оценивать разность между соседними итерациями либо, например, при решении уравнений, модуля разности левой и правой частей уравнения или при отыскании корня функции — модуля ее значения. Тонкости применения этих подходов оставим специальным дисциплинам — программисту часто приходится разрабатывать алгоритмы, не владея в полном объеме соответствующей теорией.

Значащими цифрами числа называются все цифры в его десятичной записи, кроме крайних левых и крайних правых нулей.

Не стоит выделять память в виде массивов для хранения промежуточных итераций — достаточно получения оконча-

2 Зак № 767

тельного результата и (в итерационных алгоритмах) числа шагов, проделанных до достижения условия — для оценки скорости сходимости алгоритма. На этапе отладки можно просто распечатывать промежуточные результаты (трассировать программу).

Часто в задачах (например в задаче 3.29) для вычисления очередного слагаемого удобно рекуррентно использовать предыдущее слагаемое, а не организовывать дополнительный (внутренний) цикл.

Выражения для исследуемых функций, разумеется, не поддаются вводу; их полезно оформить в виде подпрограмм или просто отдельно выделенной прокомментированной строкой.

Пример. Проверить численно справедливость следующего разложения: $e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!} + \ldots$ и оценить скорость сходимости, найдя число слагаемых, необходимое для достижения заданной погрешности ε.

Решение. Для заданного x вычислим левую часть, используя встроенную функцию Exp(x), и будем вычислять частичную сумму ряда правой части $s_n = \sum_{i=0}^{n} \frac{x^i}{i!}$ до тех пор, пока она не будет отличаться от левой части менее чем на заданную погрешность ε.

Заметим, что для вычисления каждого слагаемого ряда $u = \frac{x^i}{i!}$ требуется возведение в степень (трудоемкая операция) и вычисление факториала (а это — дополнительный цикл). Однако нетрудно видеть, что каждое очередное слагаемое можно рекуррентно вычислить через предыдущее: $u_i = \frac{x}{i} u_{i-1}$, что требует всего двух операций.

Итак, решение можно проиллюстрировать следующей Паскаль-программой:

```
Program Iteration (Input. Output):
   var x. Eps. y. s. u: Real:
          n: Integer:
   const Limit = 100:          {ограничение на число шагов}
```

```
begin
   Write ('Задай аргумент и погрешность: ');
   ReadLn (x, Eps);
   y := Exp(x); {левая часть}
   s := 1; {частичная сумма}
   u := 1; {первое слагаемое}
   n := 1; {число шагов}
   repeat
         u := x/n*u; {очередное слагаемое}
         s := s+u;
         n := n+1;
   until (Abs (y-s) <= Eps) or (n>=Limit);
   if n >= Limit then
         WriteLn(n, ' шагов не хватило',
                     ' для достижения точности')
   else
         begin
               WriteLn ('Левая часть: ', y:15:6);
               WriteLn ('Правая часть: ', s:15:6);
               WriteLn ('Погрешность ',Eps:10:6,
                           ' достигнута за ',n, ' шагов');
         end
end.
```

Замечание. Владение циклическими операторами позволяет в полной мере реализовать «дуракоустойчивость» программы, то есть защиту от некорректных исходных данных. Напомним, что программа должна отвергать при вводе некорректные данные и устойчиво работать при корректных. Пусть, например, по условию задачи $0 < x \leq \pi$, кроме того, разумно потребовать $\varepsilon > 0$. Тогда ввод можно организовать следующим образом:

```
var Okey: Boolean;
         . . .
   repeat {повторять, пока не поумнеет}
         Write ('Задай аргумент от 0 до ',Pi,

погрешность ');
         ReadLn (x, eps);
         Okey := (x>0) and (x<=Pi) and (Eps>0);
         if not Okey then
               WriteLn
                     ('Что-то здесь не так, повторите');
   until Okey;
```

Кроме того, после отладки итерационной программы полезно определить ее область применения, то есть найти, при каких x и ε процесс сходится за приемлемое время.

Задачи по теме «Циклические и итерационные алгоритмы»

3.1 (5 б.) **Четность функции.** Численно убедиться, является ли заданная функция $y = f(x)$ четной или нечетной на заданном отрезке $-a \le x \le a$. Учесть погрешность вычислений и возможные точки разрыва функции. Проверить, например, для функций $y = x^4$, $y = \mathrm{tg}x$, $y = e^x$, вычисляя их на отрезке $[-5; 5]$ с шагом 0,1.

3.2 (5 б.) **Периодические функции.** Утверждается, что функция $y = f(x)$ периодическая с периодом T. Проверить это численно, вычислив функцию с постоянным шагом на отрезке $[0;5T]$. Учесть погрешность вычислений и возможные точки разрыва функций. Проверить на примере функций: $y = \sin^2 x$, $y = \mathrm{tg}\,x$ $(T = \pi)$; $y = \dfrac{1}{x}\sin x$ $(T = 2\pi)$.

3.3 (6 б.) **Точки внутри эллипса.** Для заданных a и b найти все точки с целочисленными координатами, находящиеся внутри эллипса $x^2/a^2 + y^2/b^2 \le 1$. Полезно, используя процедуру GotoXY в Паскале, вывести найденные координаты точек в форме эллипса.

3.4 (8 б.) **Площади прямоугольников.** Прямоугольник на плоскости $a \le x \le b$; $c \le y \le d$ задается четырьмя числами (его габаритами): a, b, c, d. Последовательно вводятся габариты n прямоугольников. В процессе ввода находить площадь их пересечения, не запоминая самих габаритов.

3.5 (7 б.) **Коммерция.** Предприниматель, начав дело, взял кредит размером k рублей под p процентов годовых и вложил его в свое дело. По прогнозам, его дело должно давать прибыль r рублей в год. Сможет ли он накопить сумму, достаточную для погашения кредита, и если да, то через сколько лет?

3.6 (8 б.) **Время обработки.** Каждая из деталей должна последовательно пройти обработку на каждом из трех станков. Продолжительности обработки каждой детали на каждом станке вводятся группами по 3 числа, до исчерпания ввода. Сколько времени займет обработка всех деталей?

3.7 (9 б.) **Время обслуживания.** Для каждого посетителя парикмахерской (с одним мастером) известны следующие величины: t — момент его прихода и τ — продолжительность его обслуживания. Сколько клиентов обслужит мастер за смену продолжительностью T? Сколько рабочего времени он потратит на обслуживание?

3.8 (8 б.) **Отскоки.** Материальная точка бросается на горизонтальную плоскость под углом α к ней со скоростью v_0. При каждом ударе о плоскость кинетическая энергия точки уменьшается в β раз. Найти абсциссы первых n точек касания. Сопротивлением воздуха пренебречь.

3.9 (7 б.) **Голодная зима.** Суточный рацион коровы составляет u кг сена, v кг силоса и w кг комбикорма. В хозяйстве, содержащем стадо из k голов, осталось s кг сена, t кг силоса и f кг комбикорма. В стаде ежедневно погибает p% коров; ежедневно q% оставшегося сена сгнивает; r% силоса разворовывается колхозниками; t% комбикорма распродает зав. фермой. Когда нельзя будет кормить всех оставшихся коров по полному рациону? Какой из видов кормов кончится раньше других?

3.10 (7 б.) **Расписание.** Известно время начала и окончания (например, 6:00 и 24:00) работы некоторого пригородного автобусного маршрута с одним автобусом на линии, а также протяженность маршрута в минутах (в один конец) и время отдыха на конечных остановках. Составить суточное расписание этого маршрута (моменты отправления с конечных пунктов) без учета времени на обед и пересменку.

3.11 (8 б.) **Из миль в километры.** Получить таблицу пересчета миль в километры и обратно (1 миля = 1,609344 км) для расстояний, не превышающих k км, в следующем виде:

мили	км
0,6214	1,0000
1,0000	1,6093
1,2428	2,0000
1,8641	3,0000
2,0000	3,2187
...	...

3.12 (9 б.) **Бином Ньютона.** Для заданных m и x вычислить *бином Ньютона* $(1+x)^m$ непосредственно и по формуле разложения в ряд:

$$(1+x)^m = \sum_{i=0}^{m} C_m^i x^i.$$

Для вычисления C_m^i можно использовать рекуррентное соотношение:

$$C_m^{i+1} = C_m^i \frac{m-i}{i+1};\ C_m^0 = 1,$$

либо классическую формулу $C_m^n = \dfrac{m!}{n!(m-n)!}$. Какой из подходов эффективнее?

Проверка замечательных пределов (задачи 3.13, 3.14)

3.13 (5 б.) Проверить численно *первый замечательный предел* $\lim\limits_{x\to 0}\dfrac{\sin x}{x} = 1$, задавая x значения $1; \frac{1}{2}; \frac{1}{4}; \frac{1}{8}; \ldots$ до тех пор, пока левая часть равенства не будет отличаться от правой менее чем на заданную погрешность ε.

3.14 (5 б.) Проверить численно *второй замечательный предел*: $\lim\limits_{n\to\infty}\left(1+\dfrac{1}{n}\right)^n = e$, задавая n значения 1, 2, 3... При каком n исследуемое выражение отличается от e менее чем на заданную погрешность ε?

Сходимость итерационных процессов (задачи 3.15–3.18)

3.15 (9 б.) Сравнить *скорость сходимости* (число слагаемых для достижения заданной точности ε) следующих разложений числа π:

$$\pi = 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \ldots\right);$$

$$\pi = 3 + 4\left(\frac{1}{2\cdot3\cdot4} - \frac{1}{4\cdot5\cdot6} + \frac{1}{6\cdot7\cdot8} - \ldots\right);$$

$$\pi = \sqrt{6\left(1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \ldots\right)}.$$

3.16 (12 б.) Сравнить скорость сходимости при вычислении числа e с помощью ряда и бесконечной дроби:

$$e = 2 + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \ldots;\quad e = 1 + \cfrac{1}{1 - \cfrac{1}{2 + \cfrac{1}{3 - \cfrac{1}{2 + \cfrac{1}{5 - \ldots}}}}}.$$

3.17 (8 б.) Сколько сомножителей надо взять в произведении: $\prod_{k=1}^{\infty}\left(1 + \frac{(-1)^k}{2k+1}\right) = \frac{\sqrt{2}}{2}$, чтобы равенство выполнялось до шестой значащей цифры, то есть с погрешностью не более 10^{-6}?

3.18 (7 б.) Известно равенство: $\prod_{k=2}^{\infty}\left(1 - \frac{1}{k^2}\right) = \frac{1}{2}$. Сколько сомножителей надо взять в произведении, чтобы равенство выполнялось до пятой значащей цифры, то есть с погрешностью не более 10^{-5}?

Рекуррентные формулы (задачи 3.19–3.22)

3.19 (8 б.) Для заданных a и p вычислить $x = \sqrt[p]{a}$, используя рекуррентную формулу:

$$x_{n+1} = \frac{x_n}{p^2}\left[(p^2-1) + \frac{1}{2}(p+1)\frac{a}{x_n^p} - \frac{1}{2}(p-1)\frac{x_n^p}{a}\right].$$

Сколько итераций надо выполнить, чтобы для заданной погрешности ε было справедливо соотношение $|x_{n+1} - x_n| \le \varepsilon$? При каких начальных приближениях x_0 процесс сходится?

3.20 (7 б.) Для заданных a и p вычислить $x = \sqrt[p]{a}$ по *рекуррентному соотношению Ньютона*:

$$x_{n+1} = \frac{1}{p}\left[(p-1)x_n + \frac{a}{x_n^{p-1}}\right]; \quad x_0 = a.$$

Сколько итераций надо выполнить, чтобы для заданной погрешности ε выполнялось соотношение: $|x_{n+1} - x_n| \le \varepsilon$?

3.21 (7 б.) Вычислить $x = \sqrt[3]{a}$ для заданного значения a, используя рекуррентное соотношение:

$$x_{n+1} = \frac{1}{3}\left(x_n + 2\sqrt{\frac{a}{x_n}}\right); \quad x_0 = a.$$

Сколько итераций надо выполнить для достижения заданной погрешности ε, используя условие: $|x_{n+1} - x_n| \le \varepsilon$?

3.22 (7 б.) Для заданного $x > 1$ вычислить $y = \sqrt{x}$ по итерационной формуле: $y_i = \frac{1}{2}\left(y_{i-1} + \frac{x}{y_{i-1}}\right)$ с заданной погрешностью ε, задав начальное приближение $y_0 = x$. Сравнить с результатом использования встроенной функции. Сколько итераций пришлось выполнить?

3.23 (6 б.) **Текущее среднее.** Числа $x_1, x_2, \ldots$ последовательно поступают с устройства ввода. Все числа хранить в памяти нет необходимости; после ввода каждого числа нужно вычислить и напечатать среднее значение всех введенных чисел: $s_n = \frac{1}{n}\sum_{k=1}^{n} x_k$.

3.24 (6 б.) **Композицией n-го порядка** $f^{[n]}(x)$ функции $f(x)$ назовем результат n-кратного вычисления функции f, то есть $f^{[1]}(x) = f(x)$, $f^{[2]}(x) = f(f(x))$, и так далее. Для заданных n и x вычислить $(\exp\ln)^{[n]}(x)$ и $\exp^{[n]}\ln^{[n]}(x)$, результаты сравнить с x, *то есть вывести значения аргумента, композиции функций и разности между ними*.

3.25 (9 б.) **Число сочетаний.** Для заданных m и n вычислить *число сочетаний* C_m^n непосредственно: $C_m^n = \frac{m!}{n!(m-n)!}$ и по рекуррентным формулам:

- $C_m^n = \frac{m-n+1}{n} C_m^{n-1}$, $C_m^1 = m$;
- $C_m^n = C_{m-1}^{n-1} + C_{m-1}^n$; $C_m^m = 1$.

Сравнить *время вычислений* (или число операций) по каждой формуле.

В задачах 3.26–3.41 задана функция и ее разложение в ряд или произведение. Численно убедиться в справедливости равенства, для чего для заданного значения аргумента x вычислить левую его часть и разложение, стоящее в правой части, с заданной погрешностью ε. Испытать разложение на сходимость при разных значениях аргумента, оценить скорость сходимости, для чего вывести число итераций n (слагаемых или сомножителей), необходимых для достижения заданной точности. В некоторых задачах указан интервал допустимых значений аргумента x, при которых сходимость гарантируется.

3.26 (6 б.) $\ln x = 2\left[\frac{x-1}{x+1} + \frac{1}{3}\left(\frac{x-1}{x+1}\right)^3 + \ldots + \frac{1}{2n-1}\left(\frac{x-1}{x+1}\right)^{2n-1} + \ldots\right]$, $x > 0$.

3.27 (6 б.) $\ln(1-x) = -\left(x + \frac{x^2}{2} + \frac{x^3}{3} + \ldots + \frac{x^n}{n} + \ldots\right), \quad x < 1.$

3.28 (6 б.) $\ln|\sin x| = -\ln 2 - \cos 2x - \frac{\cos 4x}{2} - \ldots - \frac{\cos 2nx}{n} - \ldots,$
$0 < x < \pi.$

3.29 (7 б.) $a^x = 1 + \frac{x \ln a}{1!} + \frac{(x \ln a)^2}{2!} + \ldots + \frac{(x \ln a)^n}{n!} + \ldots$

3.30 (7 б.) $\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \ldots + (-1)^{n-1} \frac{x^{2n-1}}{(2n-1)!} + \ldots$

3.31 (7 б.) $\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots + (-1)^n \frac{x^{2n}}{(2n)!} + \ldots$

3.32 (6 б.) $x = 2\left(\sin x - \frac{\sin 2x}{2} + \frac{\sin 3x}{3} - \ldots + (-1)^{n-1} \frac{\sin nx}{n} + \ldots\right),$
$-\pi < x < \pi.$

3.33 (6 б.) $\sin x = x\left(1 - \frac{x^2}{\pi^2}\right)\left(1 - \frac{x^2}{4\pi^2}\right)\ldots\left(1 - \frac{x^2}{(n-1)^2\pi^2}\right)\ldots$

3.34 (6 б.) $\cos x = \left(1 - \frac{4x^2}{\pi^2}\right)\left(1 - \frac{4x^2}{9\pi^2}\right)\ldots\left(1 - \frac{4x^2}{(2n-1)^2\pi^2}\right)\ldots$

3.35 (6 б.) $\operatorname{arctg} x = x - \frac{x^3}{3} + \frac{x^5}{5} - \ldots + (-1)^n \frac{x^{2n+1}}{2n+1} + \ldots$

3.36 (7 б.) $\frac{e^x + e^{-x}}{2} = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots + \frac{x^{2n}}{(2n)!} + \ldots$

3.37 (7 б.) $\frac{e^x - e^{-x}}{2} = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \ldots + \frac{x^{2n+1}}{(2n+1)!} + \ldots$

3.38 (6 б.) $\frac{\pi^2}{8} - \frac{\pi}{4}|x| = \frac{\cos 3x}{3^2} + \frac{\cos 5x}{5^2} + \ldots + \frac{\cos(2n+1)x}{(2n+1)^2} + \ldots,$
$|x| < 1.$

3.39 (6 б.) $\frac{1}{4}\ln\frac{1+x}{1-x}+\frac{1}{2}\operatorname{arctg}x = x+\frac{x^5}{5}+\ldots+\frac{x^{4n+1}}{4n+1}+\ldots,$
$-1 < x < 1.$

3.40 (7 б.) $(1+2x^2)e^{x^2} = 1+3x^2+\ldots+\frac{2n+1}{n!}x^{2n}+\ldots$

3.41 (6 б.) $\frac{1}{4}\left(x^2-\frac{\pi^2}{3}\right) = -\cos x+\frac{\cos 2x}{2^2}-\ldots+(-1)^n\frac{\cos nx}{n^2},$
$\frac{\pi}{5}\le x\le\pi.$

3.42 (6 б.) **Расписание звонков.** В учебном заведении задается начало учебного дня, продолжительность «пары» или урока, продолжительность обычного и большого перерывов (и их «место» в расписании), количество пар (уроков). Получить расписание звонков на весь учебный день.

3.43 (7 б.) **Текущая стоимость оборудования.** Фирма ежегодно на протяжении n лет закупала оборудование стоимостью соответственно $s_1, s_2, \ldots, s_n$ р. в год (эти числа вводятся и обрабатываются последовательно). Ежегодно в результате износа и морального старения (амортизации) все имеющееся оборудование уценяется на p%. Какова общая стоимость накопленного оборудования за n лет?

3.44 (6 б.) **Вырубка леса.** Леспромхоз ведет заготовку деловой древесины. Первоначальный объем ее на территории леспромхоза составлял p кубометров. Ежегодный прирост составляет k %. Годовой план заготовки — t кубометров. Через сколько лет в бывшем лесу будут расти одни опята?

3.45 (7 б.) **Гуси и кролики.** У гусей и кроликов вместе $2n$ лап. Сколько может быть гусей и кроликов (вывести все возможные сочетания)?

3.46 (6 б.) Для заданных значений n и x вычислить выражение: $s = \sin x + \sin\sin x + \ldots + \underbrace{\sin\sin\ldots\sin x}_{n\ \text{раз}}$.

3.47 (8 б.) Для заданного ε найти наименьшее n такое, что $2^n/n! < \varepsilon$. Вывести все члены последовательности от 1-го до n-го.

4. Простейшие операции над массивами

При работе с массивами рекомендуется опробовать механизм использования массивов переменной размерности. Если (например, в Паскале) при объявлении массивов размерность задается константами, следует объявлять максимально разумные размерности, а затем вводить переменные размерности и циклы организовывать уже по ним.

Обращаем внимание на аккуратность использования индексов: общепринято в матрицах, например, первым индексом обозначать номер строки, а вторым — номер столбца (или количество соответственно).

При обработке массивов (вводимых с клавиатуры или генерируемых случайно) рекомендуется использовать «эхопечать» — вывод на дисплей всего введенного массива в наглядной форме, с рациональным использованием площади экрана — для визуального контроля правильности ввода и демонстрации соответствия результатов введенным данным.

Следует максимально ограничивать себя в выделении дополнительных (рабочих) массивов размерности того же порядка, что и обрабатываемые, а также искать эффективные по трудоемкости алгоритмы: во многих задачах возможно решение за один проход (просмотр) массива, хотя провокационно напрашивается многопроходный алгоритм. Кроме того, в «пассивных» задачах, посвященных анализу или поиску в заданном массиве, следует воздерживаться от искажений массива в своих целях.

Пример. Реализуем на массивах выполнение строевых команд «ряды сдвой» и «сомкнись». То есть из массива $A(2n)$ элементы с четными индексами нужно перенести в начало массива $B(n)$, а оставшиеся — сдвинуть к началу массива A.

Пересылка из A в B очевидна и выполняется в одном цикле с пересчетом индексов. Для удаления образовавшихся «пустот» в A возможны следующие подходы:

- передвинуть все элементы, начиная с 3-го и до конца, на одну позицию, затем начиная с 5-го — еще на одну позицию, и так далее — за несколько проходов исходного массива;
- передвигать каждый элемент массива A на новое место, вычисляя каждый раз, на сколько позиций он сдвинется — при этом требуется только один проход;
- совместить пересылку в B со сдвигом в A — все в одном цикле.

Промоделируйте каждый из этих подходов, представив себя в роли командира солдат! Очевидно, наиболее эффективен третий прием, он и реализован в предлагаемой программе.

```
Program Move (Input, Output);
   var          A: Array [1..20] of Integer;
                B: Array [1..10] of Integer;
                k,n,i: Byte;
begin
   Write
         ('Сколько всего чисел? (четное число)');
   ReadLn (n);
   for i:=1 to n do
         begin
                Write (i,'-й элемент ');
                ReadLn (A[i]);
         end;
                {эхопечать введенного массива}
   WriteLn ('Заданный массив':45);
   for i:=1 to n do Write (A[i]:4);
   WriteLn;
   k:=n div 2;
   for i:=1 to k-1 do
         begin
```

```
            B[i]:=A[2*i]; {пересылка}
            A[i+1]:=A[2*i+1]
                    {сдвиг очередного из оставшихся}
        end;
    B[k]:=A[2*k];
        {последняя пересылка: сдвигать нечего}
    {распечатка результатов}
    WriteLn ('Результат: массив A':25);
    for i:=1 to k do Write (A[i]:4);
    WriteLn;
    WriteLn ('массив B':25);
    for i:=1 to k do Write (B[i]:4);
    WriteLn
end.
```

Испытывать эту программу можно на любом массиве (размерностью до 20) любых разнородных чисел (желательно, не более чем 3-разрядных — для удобства восприятия результатов).

Задачи по теме «Простейшие операции над массивами»

4.1 (5 б.) **Разделение по знаку.** В массиве $C(n)$ подсчитать количество отрицательных и сумму положительных элементов.

4.2 (5 б.) **Из строки в матрицу.** Элементы одномерного массива $A(n^2)$ построчно расположить в матрице $B(n,n)$.

Среднее значение. В задачах 4.3–4.5 использованы элементы математической статистики.

4.3 (6 б.) **Центрирование массива.** От каждого из заданных m чисел $x_1, x_2, ..., x_m$ отнять их *среднее арифметическое*:

$$x_{ср} = \frac{1}{m}\sum_{i=1}^{m} x_i; \quad \tilde{x}_i = x_i - x_{ср}, \quad i = 1, 2, ..., m.$$

Результаты разместить на месте исходных данных.

4.4 (6 б.) **Дисперсия — 1.** Вычислить *среднее значение* $x_{ср}$ и *дисперсию* d_x для заданного массива $X(k)$ наблюдений:

$$x_{ср} = \frac{1}{k}\sum_{i=1}^{k} x_i; \quad d_x = \frac{1}{k-1}\sum_{i=1}^{k}(x_i - x_{ср})^2.$$

4.5 **(6 б.) Дисперсия — 2.** Вычислить среднее значение $x_{ср}$ и дисперсию d_x для заданного массива $X(k)$ наблюдений более эффективным (по сравнению с предыдущей задачей) способом, за один проход массива:

$$s_1 = \sum_{i=1}^{k} x_i; \quad s_2 = \sum_{i=1}^{k} x_i^2; \quad x_{ср} = \frac{s_1}{k}; \quad d_x = \frac{s_2}{k-1} - \frac{s_1^2}{k(k-1)}.$$

4.6 **(6 б.) Угол между векторами.** Найти угол между векторами $A(n)$ и $B(n)$, используя формулу:

$$\cos\varphi = \frac{(A,B)}{|A|\cdot|B|} = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}}.$$

Преобразование элементов матриц и векторов: задачи 4.7–4.10.

4.7 (6 б.) В матрице $Z(m, m)$ каждый элемент разделить на диагональный, стоящий в том же столбце.

4.8 (7 б.) В массиве $C(m)$ каждый третий элемент заменить полусуммой двух предыдущих, а стоящий перед ним — полусуммой соседних с ним элементов. Дополнительный (рабочий) массив не использовать.

4.9 (6 б.) В матрице $A(m, n)$ все ненулевые элементы заменить обратными по величине и противоположными по знаку.

4.10 (7 б.) Найти среднее арифметическое элементов каждой строки матрицы $Q(l, m)$ и вычесть его из элементов этой строки.

Пересылка элементов из массива в массив: задачи 4.11–4.16

4.11 (6 б.) Задана матрица $A(k, l)$. Найти вектор $B(l)$, каждый элемент которого равен среднему арифметическому элементов соответствуюшего столбца матрицы A.

4.12 (6 б.) Все ненулевые элементы матрицы $D(k, l)$ расположить в начале массива $E(k\cdot l)$ и подсчитать их количество.

4.13 (7 б.) Дан массив $A(n)$. Все положительные его элементы поместить в начало массива $B(n)$, а отрицательные элементы — в начало массива $C(n)$. Подсчитать количество тех и других.

4.14 (6 б.) Элементы заданного массива $T(k)$ расположить в обратном порядке: $t_k, t_{k-1}, \ldots, t_2, t_1$.

4.15 (7 б.) Заданы массивы $A(m)$ и $B(n)$. Получить массив $C(m+n)$, расположив в начале его элементы массива A, а затем — элементы массива B.

4.16 (6 б.) Все четные элементы целочисленного массива $K(n)$ поместить в массив $L(n)$, а нечетные — в массив $M(n)$. Подсчитать количество тех и других.

Индексы массивов и коэффициенты многочленов (задачи 4.17–4.20)

4.17 (6 б.) В массиве $A(n)$ найти и напечатать номера (индексы) локальных максимумов, то есть таких a_i, что $a_{i-1} < a_i > a_{i+1}$.

4.18 (7 б.) В массиве $Z(2n)$ каждый элемент с четным индексом поменять местами с предыдущим, то есть получить последовательность чисел $z_2, z_1, z_4, z_3, \ldots, z_{2n}, z_{2n-1}$.

4.19 (8 б.) Многочлен $P_n(x)$ задан массивом своих коэффициентов $A(n+1)$. Найти массив коэффициентов производной этого многочлена.

4.20 (8 б.) Многочлен $P_n(x)$ задан массивом своих коэффициентов $A(n+1)$. Вычислить значение многочлена для заданного значения x. Полезно использовать *схему Горнера*, требующую всего n трудоемких операций умножения:

$$a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0 = ((\ldots(a_n x + a_{n-1})x + a_{n-2})x + \ldots +$$
$$+ a_1)x + a_0.$$

4.21 (5 б.) **Шахматная доска.** Целочисленный массив $K(n, n)$ заполнить нулями и единицами, расположив их в шахматном порядке.

4.22 (9 б.) **Плюсы и минусы.** В массиве $Z(m)$ найти число чередований знака, то есть число переходов с минуса на плюс или с плюса на минус. Например, в последовательности 0, –2, 0, –10, 2, –1, 0, 0, 3, 2, –3 четыре чередования (как известно, нуль не имеет знака).

4.23 (8 б.) **Латинский квадрат.** Латинским квадратом порядка n называется квадратная таблица размером $n \times n$, каждая строка и каждый столбец которой содержит все числа от 1 до n. Для заданного n в матрице $L(n; n)$ построить латинский квадрат порядка n.

4.24 (7 б.) **Нарастающий итог.** В массиве $A(n)$ каждый элемент, кроме первого, заменить суммой всех предыдущих элементов.

4.25 (7 б.) **Разные соседи.** Заполнить матрицу заданного размера $M(k, l)$ числами 1, 2, 3, 4 так, чтобы по горизонтали, вертикали и диагонали не было одинаковых рядом стоящих чисел.

4.26 (8 б.) **Король и ферзи.** На шахматной доске находятся король и несколько ферзей другого цвета. Проверить, находится ли король под угрозой и если да, кто ему угрожает. Положение фигур задано массивом $K(8, 8)$: 0 — клетка пуста, 1 — король, 2 — ферзь. Ферзь бьет по горизонтали, вертикали и диагоналям.

4.27 (7 б.) **Сессия.** Результаты сессии, состоящей из трех экзаменов, для группы из n студентов представлены матрицей $K(n, 3)$. Оценка ставится по четырехбалльной системе; неявка обозначена единицей. Подсчитать количество неявок, неудовлетворительных, удовлетворительных, хороших и отличных оценок по каждому экзамену.

4.28 (7 б.) **Текущее сглаживание.** Каждый из элементов x_i массива $X(n)$ заменить средним значением первых i элементов этого массива.

4.29 (7 б.) **Текущий минимум.** Каждый из элементов t_i массива $T(m)$ заменить минимальным среди первых i элементов этого массива.

4.30 (8 б.) **Турнирная таблица.** В матрице $K(n, n)$ представлена турнирная таблица соревнований по футболу среди n участников (каждый элемент a_{ij} матрицы есть число голов, забитых i-м участником j-му участнику); все элементы главной диагонали равны нулю. Присвоить каждому диагональному элементу разницу забитых и пропущенных голов соответствующего участника, то есть разность между суммами элементов соответствующих строки и столбца.

4.31 (8 б.) **Циклический сдвиг.** Осуществить циклический сдвиг элементов массива $T(n)$ на m позиций влево, то есть получить массив: $t_{m+1}, ..., t_n, t_1, ..., t_m$. При этом необязательно $m < n$.

4.32 (10 б.) **Суммы по косой.** Просуммировать элементы матрицы $A(n, n)$ по каждой из линий, параллельных главной диагонали. Напечатать полученные суммы.

4.33 (7 б.) **Норма матрицы.** Для заданной матрицы $A(m, n)$ найти ее норму: $\|A\| = \max\limits_{i=\overline{1,m}} \sum\limits_{k=1}^{n} |a_{ik}|$.

4.34 (9 б.) **Расписание автобусов.** Известно время начала (например 6:00) и окончания (24:00) работы некоторого пригородного автобусного маршрута с несколькими (n) автобусами на линии, а также протяженность маршрута в минутах (в один конец) и время отдыха на конечных остановках. Составить суточное расписание этого маршрута (моменты отправления с конечных пунктов) без учета обеда и пересмен.

4.35 (7 б.) **Совместная работа.** Известно время $t_1, t_2, ..., t_n$, за которое некоторую работу может выполнить каждый из n рабочих бригады, работая в одиночку. Сколько времени понадобится бригаде на выполнение этой работы, если они будут работать совместно (и при этом никто не «сачкует»)?

4.36 (8 б.) **Волга-матушка.** Гидрологами исследовано течение реки в некотором сечении: произведена серия замеров по прямой от берега до берега перпендикулярно фарватеру, получены данные: s_i — расстояние от левого берега, м; h_i — глубина реки, м; v_i — скорость течения, м/с, $i = 1, 2, ..., n$. Каков *расход воды в сечении*, то есть сколько кубометров воды протекает через сечение в секунду?

4.37 (9 б.) **Автостоп-2.** Из пункта A в пункт B, между которыми s км, выехал велосипедист с постоянной скоростью v_0 км/ч. Навстречу ему — из пункта B — другой путешественник решил добраться «автостопом» — на разных видах попутного транспорта. Перед каждым участком он τ_i минут «голосует», ожидая попутного транспорта, затем движется t_i часов со скоростью v_i км/час (величины τ_i, t_i, v_i, $i = 1, 2, ..., n$ заданы в соответствующих массивах). Через какое время после старта и на каком расстоянии от пункта A путники встретятся?

Часть II
Работа с массивами

5. Векторы и матрицы

Задачи этой темы подразумевают более сложные операции по обработке массивов: ввод, хранение в разных формах, элементы поиска. Основная цель темы — дать представление о хранении информации большого объема в оперативной памяти, вычислении индексных выражений и адресации элементов, а также о выборе типов данных, оценке занимаемого ими объема.

Пример. Рассмотрим задачу 5.6, в которой требуется каждый элемент заданной матрицы заменить минимальным элементом, выбираемым среди элементов, стоящих не ниже и не правее этого элемента, включая его значение. Возможно несколько подходов к решению.

1. Для поиска минимального значения подматрицы $A'(i, j)$ для каждых i и j «просканировать» всю подматрицу. Это приводит к следующему решению (дадим лишь фрагмент программы — «тело» алгоритма):

```
for j:=1 to m do
 for i:=1 to n do
     begin
          s:=A[i,j];
          for t:=1 to j do
          for l:=1 to i do
               if A[l,t]<s then s:=A[l,t];
          A[i,j]:=s;
 end;
```

   Очевидно, для решения задачи потребуется порядка $(mn)^2$ операций сравнения и присваивания, то есть трудоемкость алгоритма полиномиальная, четвертого порядка от линейного размера матрицы.

2. Нетрудно видеть, что при движении по матрице слева направо и сверху вниз достаточно сравнивать значение каждого элемента с уже измененными его «соседями» слева и сверху, так как они уже содержат в себе минимальное значение среди элементов соответствующих подматриц, с его собственным значением. Особый случай — первая строка, где сравнение идет только с соседом справа, и первый столбец, где сравнение идет только с соседом сверху. Полное решение задачи может выглядеть следующим образом:

```
program Change_5_6_2;
      var
            a: Array[1..10,1..10] of Real;
            n,m,i,j,k,l: Integer;
begin
      Write('сколько строк? ');
      Readln(m);
      Write('сколько столбцов? ');
      Readln(n);
      for i:=1 to m do
            for j:=1 to n do
                  begin
                        Write
                  ('чему = A(',i,',',j,') элемент? ');
                        Readln(a[i,j]);
                  end;
      for j:=2 to n do
            if a[1,j-1]<a[1,j] then
                  a[1,j]:=a[1,j-1];
                  {обработка 1-й строки}
      for i:=2 to m do
            if a[i-1,1]<a[i,1] then
                  a[i,1]:=a[i-1,1];
                  {обработка 1-го столбца}
      for i:=2 to m do
            for j:=2 to n do
                  {обработка оставшейся части матрицы}
                  begin
                        if a[i,j]>a[i-1,j] then
                              a[i,j]:=a[i-1,j];
                  {сравнение с элементом выше данного}
                        if a[i,j]>a[i,j-1] then
                              a[i,j]:=a[i,j-1];
                  {сравнение с элементом левее данного }
```

```
                end;
        for i:=1 to m do
        {распечатка результата}
              begin
                    for j:=1 to n do
                          write(a[i,j]:8:2);
                    writeln;
              end;
end.
```

3. Тот же алгоритм с использованием технологии модульного программирования более лаконичен и нагляден:

```
program Change_5_6_3;
      type Matr = Array[1..10,1..10] of Real;
      var
            A:Matr;
            n,m,i,j,k,l: Integer;
      function Min2(a,b: Real): Real;
      {выбор наименьшего из двух чисел}
            begin
                  if a<b then Min2:=a else Min2:=b
            end; {Min2}
      function Min3(a,b,c: Real): Real;
      {выбор наименьшего из трех чисел}
            begin
                  if (a<b) and (a<c) then Min3:=a
                        else
                              if (b<a) and (b<c) then
Min3:=b
                                    else Min3:=c
            end; {Min3}
      procedure OutMatr
            (m,n:Integer; var Mat:Matr);
            {распечатка матрицы}
            begin
                  for i:=1 to m do
                        begin
                              for j:=1 to n do

write(Mat[i,j]:8:2);
                              writeln;
                        end;
            end {OutMatr};
begin
      Write('сколько строк? ');
      Readln(m);
```

```
        Write('сколько столбцов? ');
        Readln(n);
        for i:=1 to m do
              for j:=1 to n do
                    begin
                          Write
                    ('чему = A(',i,',',j,') элемент? ');
                          Readln(A[i,j]);
                    end;
        Writeln('Заданная матрица:':50);
              {Эхопечать введенного массива}
        OutMatr(m,n,A);
        for j:=2 to n do
              A[1,j]:=Min2(A[1,j],A[1,j-1]);
              {обработка 1-й строки}
        for i:=2 to m do
              A[i,1]:=Min2(A[i,1],A[i-1,1]);
              {обработка 1-го столбца}
        for i:=2 to m do
              for j:=2 to n do
                    {обработка оставшейся части матрицы}
          A[i,j]:=Min3(A[i,j],A[i-1,j],A[i,j-1]);
        Writeln('Полученная матрица:':50);
        OutMatr(m,n,A);
end.
```

4. Еще более лаконичное решение можно получить, если не выделять в отдельные алгоритмы обработку первой строки и первого столбца. Для этого достаточно «окаймить» матрицу сверху и слева достаточно большими числами, например, 10^{38} (при типе элементов матрицы Real). Приведем фрагменты программы, использующей описания и блоки предыдущего варианта:

```
program Change_5_6_4;
      type Matr = Array[0..10,0..10] of Real;
. . .
{описания переменных и процедур}
. . .
begin
. . .
{ввод исходных данных}
. . .
      for i:=0 to m do
            A[I,0]:=1e38;
            {«окаймление» слева}
```

```
        for j:=1 to n do
              A[0,j]:=1e38;
              {«окаймление» сверху}
        Writeln('Заданная матрица:':50);
        OutMatr(m,n,A);
              {Эхопечать введенного массива}
        for i:=1 to m do
              for j:=1 to n do
                    {обработка всей матрицы}
          A[i,j]:=Min3(A[i,j],A[i-1,j],A[i,j-1]);
        Writeln('Полученная матрица:':50);
        OutMatr(m,n,A);
end.
```

В этом решении повышение лаконичности и надежности алгоритма достигнуто за счет некоторого перерасхода памяти (обычно использование рабочих векторов для обработки матриц считается несущественным перерасходом). Трудоемкость по сравнению с предыдущим вариантом практически не изменилась.

Тестирование алгоритма можно провести на любой матрице, задаваемой случайно или с клавиатуры; при использовании эхопечати результат можно проверить визуально. Матрица, элементы которой не возрастают в направлении слева направо и сверху вниз, останется неименной. Аналогично решается задача 5.7.

Задачи по теме «Векторы и матрицы»

5.1 (7 б.) Строки матрицы $A(m,n)$ заполнены не полностью: в массиве $L(m)$ указано количество элементов в каждой строке. Переслать элементы матрицы построчно в начало одномерного массива $T(m \cdot n)$, подсчитать их количество.

5.2 (8 б.) **Улитка.** Матрицу $M(m, n)$ заполнить натуральными числами от 1 до $m \cdot n$ по спирали, начинающейся в левом верхнем углу и закрученной по часовой стрелке.

5.3 (7 б.) Для двух заданных матриц $A(n, n)$ и $B(n, n)$ проверить, можно ли получить вторую из первой применением конечного числа (не более четырех) операций транспонирования относительно главной и побочной диагоналей.

5.4 (7 б.) Куб состоит из n^3 прозрачных и непрозрачных элементарных кубиков. Имеется ли хотя бы один просвет по каждому из трех измерений? Если это так, вывести координаты каждого просвета.

Рекомендация. Для хранения кубика выделить трехмерный массив с базовым типом минимально возможного размера, так как значениями его будут только единицы или нули. При тестировании полезно использовать рандомизацию — случайным образом присвоить элементам 0 или 1, возможно, с указанием вероятности тех и других, с последующей послойной распечаткой.

5.5 (8 б.) Используя ту же структуру данных, что и в предыдущей задаче, построить полностью непрозрачный куб, используя ровно n^2 непрозрачных элементарных кубиков.

5.6 (8 б.) В матрице $A(m, n)$ каждый элемент a_{ij} заменить минимальным среди элементов подматрицы $A'(i, j)$, расположенной в левом верхнем углу матрицы A.

5.7 (8 б.) **Нарастающий итог.** Каждый элемент a_{ij} матрицы $A(m,n)$ заменить суммой элементов подматрицы $A'(i, j)$, расположенной в левом верхнем углу матрицы A.

5.8 (9 б.) Многочлены $P_m(x)$ и $Q_n(x)$ заданы массивами своих коэффициентов. Найти частное и остаток от деления $P_m(x)$ на $Q_n(x)$ (в виде массивов коэффициентов).

5.9 (8 б.) Ввод элементов матрицы $A(m, n)$ осуществляется в произвольном порядке тройками чисел $< i, j, a_{ij} >$. Признаком конца ввода служат три нуля: $< 0, 0, 0 >$. Проверить корректность такого ввода: все ли элементы введены, нет ли попытки повторного ввода или указания несуществующих координат i и j.

Указание. Разрешается выделение дополнительного (рабочего) массива такой же размерности, что и исходный массив, например, логического типа для хранения признаков «заполненности» элементов матрицы.

При попытке повторного ввода пользователю предоставляется выбор: оставить старое значение или заменить его новым. По завершении ввода матрица распечатывается; при этом невведенные элементы замещаются пробелами или другими символами.

5.10 (8 б.) Матрица A вводится извне (с клавиатуры, из файла) построчно; число строк велико и заранее неизвестно, но различных строк не более m. Расположить ее в выделенном массиве; при этом повторяющиеся строки включать единожды.

5.11 (7 б.) **Задача Иосифа [20].** По кругу располагаются n человек. Ведущий считает по кругу, начиная с первого, и выводит («казнит») m-го человека. Круг смыкается, счет возобновляется со следующего после «казненного»; так продолжается, пока «в живых» останется только один человек. Найти номер оставшегося «в живых» человека, а также для заданного n найти такое $m > 1$, при котором «в живых» останется первый.

5.12 (8 б.) Матрицу $A(m, n)$ заполнить следующим образом. Для заданных k и l элементу a_{kl} присвоить значение 1; элементам, окаймляющим его (соседним с ним по вертикали, горизонтали и диагоналям) – значение 2; элементам следующего окаймления — значение 3 и так далее до заполнения всей матрицы.

Примечание. Алгоритм не изменится, если координаты элемента (несуществующего) k и l находятся за пределами матрицы.

5.13 (8 б.) **Работа комбайнера.** Матрицу $K(m, n)$ заполнить следующим образом. Элементам, находящимся на периферии (по периметру матрицы), присвоить значение 1; периметру оставшейся подматрицы – значение 2 и так далее до заполнения всей матрицы.

5.14 (9 б.) **Поворот матрицы.** Рассматривая результат предыдущей задачи как нумерацию слоев матрицы, сдвинуть элементы заданной матрицы в пределах каждого слоя на одну позицию по часовой стрелке.

5.15 (8 б.) **Сглаживание.** Каждый элемент вектора $A(n)$ (кроме двух крайних) заменить выражением:

$a'_i = \dfrac{a_{i-1} + 2a_i + a_{i+1}}{4}$, $i = 2, 3, ..., n-1$, а крайние элементы — выражениями: $a'_1 = \dfrac{a_1 + a_2}{2}$; $a'_n = \dfrac{a_{n-1} + a_n}{2}$.

5.16 (9 б.) **Замочная скважина.** Даны мозаичные изображения замочной скважины и ключа. Пройдет ли ключ в скважину? То есть даны матрицы $K(m_1, n_1)$ и $L(m_2, n_2)$, $m_1 > m_2$, $n_1 > n_2$, состоящие из нулей и единиц. Проверить, можно ли наложить матрицу L на матрицу K (без поворота, разрешается только сдвиг) так, чтобы каждой единице матрицы L соответствовал нуль в матрице K, и если можно, то как (на сколько и в каком направлении следует подвинуть матрицу L по матрице K до выполнения условия)?

(12 б.) **Развитие задачи.** «Ключ» разрешается поворачивать на угол, кратный 90°; его можно также зеркально отображать.

5.17 (8 б.) В массиве $X(m, n)$ каждый элемент (кроме граничных) заменить суммой непосредственно примыкающих к нему элементов по вертикали, горизонтали и диагоналям.

5.18 (7 б.) Содержимое квадратной матрицы $A(n, n)$ повернуть на 90° по часовой стрелке, считая центром поворота центр симметрии матрицы.

5.19 (6 б.) В каждом столбце и каждой строке матрицы $P(n, n)$ содержится строго по одному нулевому элементу. Перестановкой строк добиться расположения всех нулей по главной диагонали матрицы.

5.20 (7 б.) Удалить из массива $A(n)$ нулевые элементы, передвинув на их место следующие элементы без нарушения порядка их следования. В результате должен получиться массив меньшего размера, не содержащий нулей.

5.21 (8 б.) Рациональное алгебраическое выражение $z = \sum_{i=1}^{n} a_i x^{k_i}$, $k_i \le m$, $i = 1, 2, ..., n$ задано массивом $A(n)$ коэффициентов a_i и массивом $K(n)$ соответствующих показателей степеней k_i. Привести в нем подобные члены и сформировать массив коэффициентов полученного многочлена $P_m(x)$ (по возрастанию степеней x).

5.22 (6 б.) **Касса.** В массиве $K(n)$ в порядке убывания представлены достоинства денежных знаков (купюр и монет) валютной системы некоторой страны. Реализовать выдачу в этой системе заданной суммы m минимальным числом денежных знаков.

5.23 (10 б.) **Соседи.** Из элементов массива $A(2n)$ получить массивы $B(n)$ и $C(n)$ следующим образом. Выбрать в массиве A два наиболее близких по значению элемента; меньший из них поместить в массив B, а больший — в массив C. Продолжить выбор из оставшихся элементов до полного заполнения массивов B и C.

5.24 (8 б.) **Колокол.** В массиве $A(n)$ наименьший элемент поместить на первое место, наименьший из оставшихся — на последнее место, следующий по величине — на второе место, следующий — на предпоследнее и так далее — до середины массива.

5.25 (7 б.) Матрица $K(m, m)$ состоит из нулей и единиц. Найти в ней номера (индексы) хотя бы одной строки или хотя бы одного столбца, не содержащих единицы, либо сообщить, что таковых нет.

5.26 (6 б.) С внешнего устройства (с клавиатуры, из файла) вводятся последовательно числа, количество которых велико и заранее неизвестно. Требуется сохранять и в процессе ввода каждого числа распечатывать не более m последних введенных чисел (в порядке их поступления).

5.27 (7 б.) На внешнем носителе (в файле) построчно подготовлены элементы матрицы $A(m, n)$. Известно, что в ее первом столбце не более k ненулевых элементов ($k << m$).

Ввести строки, для которых $a_{i1} \neq 0$, и сформировать из них в памяти матрицу $B(k, n)$. Вся матрица $A(m, n)$ слишком велика и «замусорена», чтобы хранить ее в памяти.

5.28 (7 б.) На внешнем носителе (в файле) построчно подготовлены элементы матрицы $A(m, n)$. Для заданных k и l ввести элементы k-й и l-й строк (пропуская промежуточные) и найти их скалярное произведение: $d = \sum_{j=1}^{n} a_{kj} a_{lj}$. Вся матрица слишком велика, чтобы хранить ее в памяти.

5.29 (7 б.) Матрица $A(m, n)$ вводится построчно; строки поступают в произвольном порядке: указывается номер строки и значения ее элементов. Проверить корректность такого ввода: все ли строки введены и не было ли попытки повторного ввода одной и той же строки.

Разрешается использование одномерного рабочего массива для хранения номеров введенных строк или логических или двоичных переменных — признаков того, что соответствующая строка определена (введена) индикатором определенности строк.

5.30 (8 б.) **Латинский квадрат.** Латинским квадратом порядка n называется квадратная таблица размером $n \times n$, каждая строка и каждый столбец которой содержат все числа от 1 до n. Проверить, является ли заданная целочисленная матрица латинским квадратом.

5.31 (8 б.) **Магический квадрат.** Магическим квадратом порядка n называется квадратная таблица размером $n \times n$, состоящая из чисел $1, 2, ..., n^2$ так, что суммы по каждому столбцу, каждой строке и каждой из двух диагоналей равны между собой. Проверить, является ли заданная целочисленная квадратная матрица магическим квадратом.

(15 б.) **Развитие задачи.** Реализовать любой алгоритм построения магического квадрата заданного размера.

5.32 (9 б.) В трехмерном массиве $K(l, m, n)$, состоящем из нулей и единиц, хранится сеточное изображение некоторого трехмерного тела. Получить в двумерных массивах три проекции (тени) этого тела.

5.33 (10 б.) Многочлены $P_n(x)$ и $Q_m(x)$ заданы своими коэффициентами. Определить коэффициенты их композиции – многочлена $P_n(Q_m(x))$.

5.34 (8 б.) **Отсев.** Удалить в заданном массиве $X(n)$ «лишние» (кроме первого) элементы так, чтобы оставшиеся образовали возрастающую последовательность (за один просмотр массива).

5.35 (8 б.) *Среднестатистическим* назовем элемент массива, если для него модуль разности его значения и среднего арифметического элементов массива достигает минимума. Аналогично, *уникальным* будем называть элемент, для которого такой модуль разности достигает максимума. В заданном массиве $X(m)$ найти номера (индексы) среднестатистического и уникального элементов.

5.36 (9 б.) **Автостоп – 3.** Из пункта A в пункт B, между которыми s км, выехал велосипедист с постоянной скоростью v_0 км/ч. Одновременно с ним в том же направлении другой путник решил добраться «автостопом» – на разных видах попутного транспорта. Перед каждым участком пути он τ_i минут «голосует», затем движется t_i часов со скоростью v_i км/ч (величины τ_i, t_i, v_i, $i = 1, ..., n_i$ заданы в соответствующих массивах). В каких точках пути (в какие моменты времени) путники смогут помахать друг другу рукой?

6. Линейный поиск

В задачах этого раздела требуется найти в массиве элемент, группу элементов или фрагмент массива, которые отвечают заданным требованиям. При этом, как правило, нет необходимости сортировать массив или выделять дополнительные рабочие массивы размерности того же порядка, что и заданный.

Если требуется найти фрагмент массива, обладающий каким-либо свойством («цепочку» элементов), следует в качестве результата показать не только сам фрагмент, но и его местоположение (индексы первого и последнего элементов фрагмента) в массиве.

Если исходный массив упорядочен, целесообразно проверить при вводе упорядоченность и применять бинарный поиск (метод дихотомии, или метод деления пополам).

Пример. Дана матрица $A(m, n)$, состоящая из нулей и единиц. Требуется найти в ней строку, содержащую хотя бы один нуль.

Встречаются следующие подходы к решению.

1. Переставить строки матрицы, упорядочив их по возрастанию количества единиц. Тогда если первая строка содержит нули, она и будет искомой — самой «нулевой».
2. Подсчитать сумму элементов каждой строки матрицы; среди строк выбрать такую, для которой сумма меньше n.
3. Подсчитать количество нулей в каждой строке и выбрать такую строку, в которой оно больше нуля.
4. Найти в матрице первый попавшийся нуль. Строка, его содержащая, и будет искомой.

3 Зак. № 767

Все четыре подхода должны дать правильный результат, но сильно различаются по эффективности. Кроме того, первый подход носит «хирургический» характер, так как матрица в результате такого «поиска» искажается до неузнаваемости и становится непригодной для других приложений.

Рассмотрим житейскую ситуацию с подобной задачей. На оккупированной врагом территории действует группа партизан. Юный разведчик получает задание: проверить, не занята ли некоторая деревушка фашистами, то есть не найдется ли среди домов этой деревни такой дом, в котором есть хотя бы один фашист?

Какой из подходов вы предложите разведчику? В первом варианте ему, очевидно, пришлось бы переселить все семьи (вместе с расквартированными фашистами) по убыванию количества фашистов. Как отнесутся к этому испытуемые? Во втором и в третьем случаях разведчик должен тщательно обследовать все дворы, подсчитывая хозяев или «гостей», и лишь в четвертом случае разведка ведется до обнаружения первых признаков наличия «гостей», а затем — «дай бог ноги». Очевидно, когда речь идет о жизни и смерти, мы выбираем единственно разумное решение, а компьютер — «он железный», пусть работает, на модельных примерах результат будет найден в мгновение ока даже при самом неэффективном алгоритме.

Итак, реализуем четвертый подход — поиск в матрице строки, содержащей хотя бы один нуль.

```
Program Search (Input, Output);
    uses Crt;
    var m,n,i,j,k: Byte;
        A: Array [1..20, 1..40] of 0..1;
        okey: Boolean; {индикатор обнаружения}
begin
    Write
        ('Задай размерность: m<=20, n<=40 ');
    ReadLn (m,n);
    ClrScr;
    for i:=1 to m do
        for j:=1 to n do
            begin
```

```
                        repeat {защита от некорректности}
                              GotoXY (1,24);
                              Write ('Введите элемент');
                              GotoXY (1,i+1);
                              ClrEol;
                              GotoXY ((j-1)*2+1,i);
                              ClrEol;
                              ReadLn(k);
                              if (k<>0) and (k<>1) then
                              begin
                                    GotoXY(1,i+1);
                                    WriteLn('Неверно,
повторите');
                                    Delay (500)
                              end;
                        until (k=0) or (k=1);
                        A[i,j]:=k
                  end;
    организация поиска}
    i:=0;
    repeat
          i:=i+1;
          j:=0;
          repeat
                j:=j+1;
                okey:=(A[i,j]=0);
          until okey or (j>=n);
    until okey or (i>=m);
    GotoXY (1,24);
    if okey then
          WriteLn
                ('Найдена строка с номером ',i)
    else WriteLn
                ('Строки с нулем не нашлось')
end.
```

Тестирование. При испытании программы ее надо обязательно проверить на следующих (небольших) матрицах:

- не содержащих нулей;
- содержащих нули во всех строках;
- содержащих нуль в первой строке.

Кроме того, для настройки ввода-вывода надо предложить матрицу из 20 строк и матрицу из 40 столбцов.

Задачи по теме «Линейный поиск»

6.1 (7 б.) *Седловой* точкой в матрице называется элемент, являющийся одновременно наибольшим в столбце и наименьшим в строке: $a_{pq} = \max_i a_{iq} = \min_j a_{pj}$. Седловых точек может быть несколько (в этом случае они имеют равные значения). В матрице $A(m,n)$ найти седловую точку и ее координаты p, q либо установить, что такой точки нет.

6.2 (7 б.) В матрице $K(m, n)$ первый элемент каждой строки — шифр детали, остальные элементы — характеристики этой детали. Выявить, распечатать и удалить из матрицы номера строк с совпадающими шифрами и несовпадающими характеристиками. Вывести также оставшуюся после резекции матрицу.

6.3 (9 б.) Задано n линейных функций: $y_1 = a_1x + b_1$; $y_2 = a_2x + b_2$; ...; $y_n = a_nx + b_n$. Найти минимум «верхней огибающей» этих функций, то есть кусочно-линейной функции $y(x) = \max_i(a_ix + b_i) \to \min_x$.

Указание. Можно от произвольной точки двигаться по точкам излома огибающей в сторону ее убывания.

6.4 (8 б.) Поле размером $m \times n$ заполнено прозрачными и непрозрачными кубиками. Найти все столбцы поля, все непрозрачные кубики которых невидимы для наблюдателя, расположенного слева.

6.5 (9 б.) Поле размером $m \times n$ заполнено прозрачными и непрозрачными кубиками. Удалить (сделать прозрачными) все непрозрачные кубики, видимые хотя бы с одной из четырех сторон (видимость анализируется до удаления какого-либо кубика).

6.6 (10 б.) В заданном массиве $A(n)$ найти i и j такие, что $\sum_{k=i}^{j} a_k$ максимально.

6.7 (8 б.) В массиве $A(l)$, все элементы которого различны, найти и удалить n наименьших элементов, «поджимая» массив к началу и сохраняя порядок следования остальных элементов $(n \ll l)$.

6.8 (7 б.) В массиве $T(k)$ найти первый и последний нулевые элементы.

6.9 (7 б.) В массиве $L(m)$ найти наиболее длинную цепочку, состоящую из одних нулей.

6.10 (10 б.) Матрица $L(n, k)$ состоит из нулей и единиц. Найти в ней самую длинную цепочку подряд стоящих нулей по горизонтали, вертикали или диагонали.

6.11 (9 б.) В целочисленном массиве $K(n)$ много повторяющихся элементов. Найти (в процентах) частоту появления каждого из m наиболее часто встречающихся элементов $(m \ll n)$.

6.12 (8 б.) Даны два целочисленных массива $K(m)$ и $L(n)$. Найти наибольший элемент массива K, не имеющий себе равных в массиве L.

6.13 (8 б.) Среди элементов массива $Z(m)$ найти k $(k \ll m)$ наибольших. Поиск осуществить за один проход (просмотр) массива Z.

6.14 (7 б.) В целочисленном массиве $L(n)$ найти наиболее длинную цепочку одинаковых подряд стоящих элементов.

6.15 (9 б.) В массиве $M(k)$ много совпадающих элементов. Найти количество различных элементов в нем (не упорядочивая массива).

6.16 (6 б.) Элементы массива $M(n)$ упорядочены по неубыванию (см. раздел 15). Для заданного x найти наименьшее k такое, что $m_k \le x \le m_{k+1}$, либо показать (выдать сообщение), что такового нет. Для поиска полезно применить метод дихотомии (метод деления отрезка пополам).

6.17 (5 б.) В каждой строке матрицы $A(n, n)$ найти наибольший элемент и поменять его местами с соответствующим диагональным элементом.

6.18 (8 б.) Последовательность $a_1, a_2, \ldots, a_k$, называется *пилообразной*, если $a_1 < a_2 > a_3 < a_4 > \ldots > a_k$ либо $a_1 > a_2 < < a_3 > a_4 < \ldots < a_k$. В массиве $A(m)$ найти самую длинную пилообразную последовательность.

6.19 (7 б.) Последовательность $a_1, a_2, \ldots, a_k$ называется *монотонной*, если $a_1 \le a_2 \le \ldots \le a_k$ или $a_1 \ge a_2 \ge \ldots \ge a_k$. В массиве $A(m)$ найти самую длинную монотонную последовательность.

6.20 (6 б.) Утверждается, что массив $A(m)$ целиком (как последовательность) встречается в массиве $B(n)$, $n > m$. Найти место массива A в массиве B или показать, что его в массиве B нет.

6.21 (9 б.) Найти все числа, каждое из которых встречается в каждой строке матрицы $A(m, n)$.

6.22 (7 б.) Найти все числа из массива $B(n)$, встречающиеся более чем в одной строке матрицы $A(m, n)$.

6.23 (8 б.) Найти все числа, встречающиеся в массиве $P(m)$ строго два раза (не упорядочивая самого массива).

6.24 (7 б.) В массиве $Z(n)$ найти наиболее длинную цепочку стоящих подряд попарно различных элементов.

6.25 (8 б.) В массиве $P(n)$ найти самую длинную последовательность, которая является арифметической или геометрической прогрессией.

6.26 (8 б.) **Медиана.** В массиве $A(2n+1)$, не содержащем одинаковых элементов, найти *средний по величине* элемент, то есть такой, что в массиве A ровно n элементов меньше его и столько же элементов больше его. Массив A сохранить (не сортировать), дополнительных массивов не использовать.

6.27 (7 б.) Результаты забега в массовом кроссе представлены целочисленной матрицей $K(n,4)$, где n — число участников; k_{i1}, k_{i2} — момент старта i-го участника в минутах и секундах соответственно; k_{i3}, k_{i4} — аналогично момент финиша, $i = 1, 2, \ldots, n$. Найти номера (индексы) трех призеров забега и их результаты (за один просмотр матрицы).

6.28 (7 б.) В массиве $H(n)$ хранятся значения высот некоторого профиля местности (ее вертикального сечения) с постоянным шагом по горизонтали. Найти области (номера точек измерения высоты), невидимые для наблюдателя, находящегося в точке h_1.

6.29 (7 б.) Имеются результаты n ежедневных измерений количества выпавших осадков. За какую из недель (отрезок времени длиной 7 дней), считая с начала периода измерений, выпало наибольшее количество осадков?

6.30 (6 б.) Дана таблица выигрышей денежной лотереи: $K(n)$ — массив номеров выигравших билетов (упорядочен по возрастанию); $S(n)$ — суммы выигрышей. Определить суммарный выигрыш для пачки купленных билетов с номерами $l_1, l_2, \ldots, l_m$.

6.31 (9 б.) **Крестики-нолики.** Клеточное поле размером $m \times n$ есть результат игры в крестики-нолики на «бесконечном» поле. Проверить, не закончена ли игра выигрышем «крестиков»? Считается, что «крестики» выиграли, если на поле найдется по горизонтали, вертикали или диагонали цепочка, состоящая подряд из 5 крестиков.

6.32 (8 б.) Проверить, не является ли заданная матрица $A(m,n)$ осе- или центросимметричной. При отрицательном ответе — найти хотя бы одну группу элементов-нарушителей симметрии.

6.33 (9 б.) Задан массив, состоящий из n неотрицательных чисел. Найти в нем индекс элемента для которого сумма элементов, стоящих до него, наименее отличается от суммы элементов, стоящих после него.

(12 б.) **Развитие задачи.** Числа хранятся в линейном односвязном списке (см. п. 15) или в файле с последовательным доступом. Найти наиболее эффективные алгоритмы для случаев прямого и последовательного доступа с возможностью использовать рабочий массив размерностью n или без нее.

6.34 (8 б.) Дан массив целых чисел $K(n)$. Найти в нем минимальный k_{min} и максимальный k_{max} элементы. Вывести в порядке возрастания все целые числа из интервала (k_{min}, k_{max}), не встречающиеся в исходном массиве.

6.35 (10 б.) Черный квадрат. В матрице $A(m,n)$, состоящей из нулей и единиц, найти квадрат заданного размера (квадратную подматрицу), состоящий целиком из нулей.

(16 б.) **Развитие задачи.** Найти квадрат наибольшего размера.

Часть III
Прикладные математические задачи

В этой части приведены задачи на приложение программирования в проблемах арифметики, численного и текстового анализа, а также на графическое представление информации. Эти задачи требуют знания большинства конструкций языков программирования и способов представления информации в ЭВМ, навыков программирования типичных алгоритмических задач, пакетов графических процедур, а также некоторых численных методов. При необходимости более глубокие сведения по приводимым методам можно найти в специальной литературе.

Многие задачи (например, 11.20) не содержат конкретных требований к реализации, то есть допускают множество решений разного уровня. В этом случае при оценке программы учитывается не только качество программы (наглядность, надежность, дружественный интерфейс и так далее), но и качество постановки, в том числе и художественный вкус.

7. Арифметика

Задачи этого раздела требуют умения переводить числа из одной системы счисления в другую и выполнять арифметические действия в любой системе. Если не оговорено особо, речь идет о десятичных числах и цифрах, их представляющих (напомним, что во внутреннем представлении чисел используется двоичная система; величина целого числа ограничена разрядностью ЭВМ: например, для 16-разрядных машин наибольшее целое равно 65536).

Целое число называется *простым*, если оно делится только на себя и на единицу.

Значащими цифрами числа будем называть все его цифры, кроме «незначащих» левых и правых нулей (например, числа 004520; 0,0010900; 3240000 содержат по три значащих цифры).

При решении задач не рекомендуется использовать стандартные процедуры перевода чисел в десятичную систему, такие, как STR() в Паскале.

Пример. Рассмотрим задачу 7.14, в которой предлагается среди заданных чисел найти такие, старшая цифра которых есть 9. В программе реализован метод рандомизации для отладки на массиве случайных чисел. Для выявления старшей цифры число «нормализуется», то есть многократным умножением или делением на 10 приводится к диапазону (1,10). Можно было бы более лаконично использовать десятичный логарифм числа и домножить на 10 в соответствующей степени, но вычисление функций Exp и Ln разложением в ряд едва ли повысит эффективность алгоритма по трудоемкости.

Итак, задача может быть решена следующей Паскаль-программой:

```
program OldNum_7_14;
   uses Crt;
   const n=1000;
   type mas=array[1..n] of real;
   var a:mas;
         i,j,razm,st:integer;
         Otvet:char;
         b:real;
   Procedure Rand(n:integer);
   {заполнение массива случайными числами}
         var i:integer;
         begin
               Randomize;
               for i:=1 to n do
               a[i]:=Exp(Random*20-10);
         end; {Rand}
   Procedure Keyb(n:integer);
   {ввод массива с клавиатуры}
         var i:integer;
         begin
               Writeln
                     ('Введите элементы массива':50);
               for i:=1 to n do
                     begin
                           Write('a[',i,']=');
                           Readln(a[i]);
                     end; {Keyb}
         end;
begin
   ClrScr;
   Writeln
         ('Введите размерность массива':50);
   Readln(razm);
   repeat
         WriteLn
         ('Вводим массив с клавиатуры? (Y/N)');
         ReadLn(Otvet);
         Otvet:=UpCase(Otvet);
         if (Otvet<>'Y')and(Otvet<>'N') then
               begin
                     WriteLn
                      ('Вы должны нажать "Y" или '
,'"N"');
                     Delay(5000);
```

```
                      ClrScr;
                end;
    until (Otvet='Y') or (Otvet='N');
    if Otvet='N' then Rand(razm)
          else Keyb(razm);
    WriteLn('Исходный массив':50);
          {эхопечать заданного массива}
    for i:=1 to razm do
          Write (a[i]:20:6);
    Writeln;
    Writeln('Найденные числа:':50);
    for i:=1 to razm do
          begin
                if a[i]<1 then
                      begin
                            b:=a[i];
                            repeat
                                  b:=b*10;
                            until b>=1;
                      end
                      else
                            begin
                                  b:=a[i];
                                  while b>=10 do
                                        b:=trunc(b/10);
          end;
    if trunc(b)=9 then
          begin
                WriteLn(a[i]:20);
                inc(st);
          end;
    if st=20 then {кончился кадр}
          begin
                st:=0;
                      WriteLn
                            ('Нажмите любую '
 ,'клавишу':60);
                      ReadKey;
          end;
    end;
    WriteLn('Нажмите любую клавишу');
    Readkey;
End.
```

Тестирование можно провести на массиве любых чисел либо предоставить программе «самотестироваться» на сгенерированном массиве случайных чисел.

Задачи по теме «Арифметика»

7.1 (8 б.) Натуральное число в p-ичной системе счисления задано своими цифрами, хранящимися в массиве $K(n)$. Проверить корректность такого представления и перевести число в q-ичную систему (возможно, число слишком велико, чтобы получить его внутреннее представление; кроме того, $p \le 10$, $q \le 10$).

7.2 (6 б.) Для натуральных чисел, не превосходящих заданного k, проверить признак делимости на 9 (сумма цифр числа, делящегося на 9, также делится на 9). Распечатать m последних таких чисел ($m << k$).

7.3 (7 б.) Число делится на 11, если разность между суммой цифр, стоящих на нечетных местах, и суммой цифр, стоящих на четных местах, кратна 11. Проверить этот признак для всех натуральных чисел, не превосходящих заданного m, и вывести числа, кратные 11.

7.4 (7 б.) В массиве $A(m)$ хранятся различные вещественные числа (как большие, так и меньшие единицы). Округлить их, оставив в каждом по 3 значащих цифры.

7.5 (6 б.) Для заданного m получить таблицу первых m простых чисел.

7.6 (6 б.) Своими цифрами в массивах $K(m)$ и $L(n)$ заданы два целых числа в p-ичной системе счисления ($p \le 10$). Найти в таком же виде их сумму, не вычисляя самих чисел.

7.7 (8 б.) Возвести заданное вещественное число a в целую степень k, не пользуясь операцией возведения в степень и не производя $(k-1)$ умножений и многократного сложения (так как k велико).

Рекомендация. Сокращение числа умножений может быть достигнуто применением «индийского алгоритма» — по рекуррентной формуле:

$$x^n = \begin{cases} x, \text{ если } n = 1; \\ x^{n \bmod 2}\left(x^{n div 2}\right)^2, \text{ если } n > 1. \end{cases}$$

7.8 (6 б.) Среди заданных натуральных чисел найти такие, десятичная запись которых не содержит одинаковых цифр.

7.9 (6 б.) Найти все натуральные числа, не превосходящие заданного m, сумма цифр в десятичном представлении каждого из которых равна заданному k.

7.10 (7 б.) Пусть m натуральных чисел заданы своими цифрами в q-ичной системе счисления, хранящимися в строках матрицы $K(m, n)$. Найти сумму этих чисел в той же системе, не вычисляя самих чисел $(q \le 10)$.

7.11 (8 б.) Осуществить циклический сдвиг n-разрядного двоичного представления заданного числа k на m позиций вправо, не находя цифр самого двоичного представления и не пользуясь стандартной процедурой сдвига.

7.12 (6 б.) В массивах $K(n)$ и $L(n)$ заданы соответственно числители и знаменатели рациональных чисел вида $x_i = k_i/l_i$, $i = 1, 2, \ldots, n$. Найти наибольшее из этих чисел, не пользуясь операцией деления.

7.13 (5 б.) Напечатать таблицу сложения одноразрядных чисел в p-ичной системе счисления, $p \le 16$.

7.14. (8 б.) В заданном вещественном массиве $A(n)$ найти все числа, у которых старшая значащая десятичная цифра есть 9 (числа сильно различаются по величине).

7.15 (6 б.) Найти все натуральные числа, не превосходящие заданного m и содержащие хотя бы одну девятку в десятичном представлении.

7.16 (9 б.) Перевести заданное целое число в систему римского счета.

Указание. Римские цифры обозначаются следующими латинскими буквами:

1	I	500	D
5	V	1 000	M
10	X	5 000	$\overline{V}$
50	L	10 000	$\overline{X}$
100	C	…	…

7.17 (7 б.) Найти минимальное натуральное m такое, что: $m = k^3 + l^3 = i^3 + j^3$ (k, l, i, j — различные натуральные числа).

7.18 (5 б.) Десятичное представление заданного натурального числа напечатать *вразрядку*, то есть вставить пробелы между цифрами.

7.19 (9 б.) Напечатать столбиком пример на умножение в десятичной системе счисления двух заданных натуральных чисел k и l.

7.20 (10 б.) Напечатать столбиком пример на деление с остатком двух заданных натуральных чисел k и l.

7.21 (10 б.) Произведение двух заданных натуральных чисел больше максимально допустимого значения (не вмещается в разрядную сетку машины). Найти это произведение.

7.22 (6 б.) Найти 20 первых троек *пифагоровых* чисел, то есть целых k, l, m таких, что $k^2 + l^2 = m^2$.

7.23 (6 б.) Любая целочисленная денежная сумма $s > 7$ р. может быть выдана без сдачи «трешками» и «пятерками». Найти для заданной суммы s необходимое количество «трешек» и «пятерок».

7.24 (6 б.) Найти первые m более чем 2-разрядных *чисел-палиндромов*, то есть чисел, десятичная запись которых читается одинаково в прямом и обратном направлениях, например: 373, 426 624.

7.25 (7 б.) Найти все натуральные числа, не превосходящие заданного n и делящиеся на каждую из своих цифр (в десятичной системе счисления).

7.26 (7 б.) Найти все натуральные числа, не превосходящие заданного n, десятичная запись которых есть строго возрастающая или строго убывающая последовательность цифр.

7.27 (7 б.) Каждое из заданных натуральных чисел заменить числом, получающимся при записи его десятичных цифр в обратном порядке.

7.28 (8 б.) Заданное натуральное число n, не превосходящее 1000, записать прописью, то есть вывести соответствующее количественное числительное, например: 375 — «триста семьдесят пять».

7.29 (7 б.) Найти все натуральные числа, не превосходящие заданного m, двоичная запись которых представляет собой симметричную последовательность нулей и единиц (начинающуюся с единицы). Показать десятичную и двоичную записи этих чисел.

7.30 (10 б.) Дробная часть бесконечной десятичной дроби хранится в массиве цифр $K(l)$ (без округления). Проверить, не является ли эта дробь периодической, и если это так, превратить ее в правильную дробь вида m/n.

7.31 (6 б.) Найти все натуральные числа от 1 до 1000, которые совпадают с последними разрядами своих квадратов, например: $25^2 = 625$; $76^2 = 5676$.

7.32 (7 б.) *Число Армстронга* — такое число из k цифр, для которого сумма k-х степеней его цифр равна самому числу, например: $153 = 1^3+5^3+3^3$. Найти все числа Армстронга не более чем из четырех цифр.

7.33 (8 б.) Для заданного натурального числа k найти наименьшее основание p системы счисления, в которой представление числа k не содержит нулей. Напечатать это представление.

7.34 (10 б.) Заданное действительное положительное число a представить в виде *цепной дроби* глубины n. Целые части выделяются по формулам: $k_0 = \mathrm{ent}(a)$; $x_0 = \mathrm{fl}(a)$;

$k_i = \mathrm{ent}\left(\dfrac{1}{x_{i-1}}\right)$; $x_i = \mathrm{fl}\left(\dfrac{1}{x_{i-1}}\right)$, $i = 1, 2, \ldots$ Здесь $\mathrm{ent}(x)$ — целая часть числа x; $\mathrm{fl}(x)$ — его дробная часть. Например,

$$\pi = 3 + \cfrac{1}{7 + \cfrac{1}{15 + \cfrac{1}{1 + \ldots}}}.$$

7.35 (8 б.) **Тренажер по арифметике.** Пользователь-учитель вводит разрядность операндов, тип операции: + — * / (на множестве натуральных чисел) и количество примеров. Компьютер генерирует случайным образом операнды, результат операции и выводит пользователю-ученику серию примеров, в каждом из которых один из операндов или результат «замаскирован», например: 37 * __ = 1591. Ученик вводит пропущенное число (в приведенном примере — 41); компьютер проверяет правильность и ведет статистику ошибок.

7.36 (8 б.) Для заданного натурального n найти наименьшее p — основание системы счисления — такое, что в этой системе представление числа n не содержит нулей. Для убедительности вывести представление числа n во всех системах от 2 до p.

7.37 (9 б.) Натуральное число называется *совершенным*, если оно равно сумме всех своих простых делителей (например, $6=1+2+3$). Найти все совершенные числа, не превосходящие заданного n.

Примечание. На сегодня известно 24 совершенных числа; все они четные. Вопрос о конечности множества совершенных чисел открыт.

7.38 (10 б.) Вычислить факториалы всех чисел от 1 до 100 (число 100! не может быть представлено ни в одном стандартном типе переменных, не поместится на одной строке экрана дисплея).

7.39 (8 б.) Пусть значение функции $f(n)$ равно количеству символов в русской записи количественного числительного n: $f(1)=4$ («один»), $f(3)=3$ («три»), $f(42)=9$ («сорок два») и т. д. Найти все натуральные n, для которых $f(n)=n$.

7.40 (8 б.) Дан пример на умножение, записанный в виде символьной строки, например: 213*3=1144. Определить, в какой системе счисления этот пример справедлив.

7.41 (10 б.) Среди натуральных чисел, не превосходящих заданного числа, найти такие, цифры которых в десятичной записи образуют арифметическую прогрессию.

7.42 (9 б.) Задумано целое число x. Известны числа k, m, n — остатки от деления этого числа на 3, 5 и 7 соответственно. Найти x.

7.43 (9 б.) Найти все числа, большие 10 и меньшие 1000, равные сумме k-х степеней своих цифр, где k — разрядность числа.

7.44 (8 б.) Дано натуральное k. Напечатать k-ю цифру в последовательности 12345678910111213..., в которой выписаны подряд все натуральные числа.

7.45 (8 б.) Дано натуральное k. Вывести k-ю цифру последовательности 149162536..., в которой выписаны подряд квадраты всех натуральных чисел.

7.46 (8 б.) Дано натуральное число. Переставить его десятичные цифры так, чтобы получить максимально возможное число, записанное теми же цифрами.

Примечание. Очевидно, достаточно цифры числа отсортировать по невозрастанию.

7.47 (9 б.) Каково минимальное основание p системы счисления, в которой младшая цифра представления заданного натурального числа k есть m? (Например $k = 179$, $m = 7$, $p = 43$.)

7.48 (9 б.) Каково минимальное основание p системы счисления, в которой представление заданного числа m симметрично (крайний случай: $p = m$).

7.49 (7 б.) Имеется набор гирь весом 1, 2, 5, 10, 20, 50, 100, ... грамм. Как взвесить тело заданной массы m г на равноплечих весах, используя минимальное число гирь?

(10 б.) **Развитие задачи.** Имеется лишь по одной гире каждого номинала. Разрешается класть гири на любую чашку весов. Как уравновесить тело, используя минимальное число гирь? Например, масса 27 г уравновешивается: $27 + 1 + 2 = 10 + 20$ либо $27 + 1 + 2 + 20 = 50$.

7.50 (8 б.) Пусть дробные числа (с плавающей точкой во внутреннем представлении), которые сильно различаются по величине, вводятся с клавиатуры или из файла. Вывести их на экран (или в текстовый файл по 80 символов в строке) столбиками, оставляя в числе по 2 цифры после точки, выровняв числа в столбце по правому краю так, чтобы в строке между числами было не менее 1 пробела; ширина столбца определяется самым большим числом в нем.

7.51 (7 б.) Утверждается, что разность любого натурального числа и суммы его цифр кратна 9. Проверить этот факт для всех чисел, лежащих между заданными m и n.

8. Геометрия и теория множеств

В задачах этого раздела нельзя использовать переменные типа множества в Паскале, так как они не поддаются вводу и выводу, ограничены по мощности; для задания множеств следует использовать массивы чисел или записей.

Геометрические задачи полезно сопровождать графической иллюстрацией, при этом они «повышаются в цене» (оцениваются большим числом баллов) при условии, что соблюдено *масштабирование* (то есть линейное преобразование пространства, при котором площадь экрана используется наиболее эффективно), вся картинка показана вместе с системой координат, независимо от размеров «мирового пространства» (порядка обрабатываемых чисел), и нет пустого места на экране.

Кардинальной мощностью или просто *мощностью* card(A) конечного множества A называют число элементов в нем (предполагается, что все элементы множества различны). Все рассматриваемые множества конечны и, как правило, неупорядочены.

Расстоянием между двумя множествами геометрических точек называется наименьшее расстояние между двумя точками, принадлежащими разным множествам. Расстояние между геометрическими фигурами — это расстояние между множествами точек, составляющими эти фигуры. Очевидно, что если фигуры пересекаются, расстояние между ними равно нулю.

Проекцией точки на множество A геометрических точек называется точка из A, наименее удаленная от заданной точки.

Геометрические точки задаются своими координатами. Для материальных точек указывается еще и масса каждой точки.

Координаты центра тяжести совокупности n материальных точек или фигур находятся по формулам:

$$x_ц = \frac{1}{M}\sum_{i=1}^{n} x_i m_i;\ \ y_ц = \frac{1}{M}\sum_{i=1}^{n} y_i m_i;\ \ M = \sum_{i=1}^{n} m_i,$$

где x_i, y_i — координаты i-й точки или центра тяжести i-й фигуры; m_i — ее масса (или площадь фигуры); $i = 1, 2, ..., n$. Центр тяжести треугольника находится в точке пересечения его медиан; для параллелограмма — в точке пересечения его диагоналей.

Площадь многоугольника на плоскости, образованного не самопересекающейся замкнутой ломаной $A_1A_2A_3 \ldots A_n$, может быть найдена по формуле:

$$s = \left|\sum_{i=2}^{n-1} \pm s_{1,i,i+1}\right|,$$

где $s_{1,i,i+1}$ — площадь треугольника $A_1A_iA_{i+1}$; знак $\pm$ есть знак внутреннего угла $A_1A_iA_{i+1}$.

Пример. Рассмотрим задачу 8.35: найти площадь и центр тяжести фигуры, заданной матричным изображением на клетчатом поле. Решение может выглядеть следующим образом. Исходные данные готовятся с помощью рандомизации. Программа снабжена графической иллюстрацией матричного изображения фигуры и ее центра тяжести.

```
program Center_8_35;
        uses Crt, Graph;
        const max=100;
        var A: Array[1..max,1..max] of Byte;
                i,j:Word;
                n,m:Word;
                k,massa:Word;
                x,y:Integer;
                x0,y0:Real;
                dr,dm:Integer;
begin
        ClrScr;
        Writeln('Сколько строк и столбцов?');
        ReadLn(m,n);
        Randomize;
        for i:=1 to m do
                for j:=1 to n do
                        A[i,j]:=Random(2);
```

```
        massa:=0;
        x0:=0;
        y0:=0;
        for i:=1 to m do
              for j:=1 to n do
                    if (A[i,j]=1) then
                          begin
                                massa:=massa+1;
                                x0:=x0+j;
                                y0:=y0+i;
                          end;
        x0:=x0/massa;
        y0:=y0/massa;
        WriteLn
              ('Координаты центра (',
              x0:4:2,';',y0:4:2,')');
        WriteLn('Площадь = ',massa);
        ReadKey;
        dr:=Detect;
        InitGraph(dr,dm,'d:\_langz\bp\bgi');
        if (GetMaxX/n > GetMaxY/m) then
              k:=Trunc(GetMaxY/m)
              else k:=Trunc(GetMaxX/n);
        SetFillStyle(1,7);
        for i:=1 to m do
              for j:=1 to n do
                    begin
                          x:=(j-1)*k+5;
                          y:=(i-1)*k+5;
                          RecTangle(x,y,x+k,y+k);
                          if A[i,j]=1 then
                                FloodFill(x+1,y+1,15);
                    end;
        SetFillStyle(1,12);
        FillEllipse
              (trunc((x0-0.5)*k)+5,
              trunc((y0-0.5)*k)+5,trunc(k/2),
              trunc(k/2));
        ReadKey;
  end.
```

8.1 (9 б.) Заяц, хаотично прыгая, оставил след в виде замкнутой самопересекающейся ломаной, охватывающей территорию его владения (отрезки ломаной заданы длиной прыжка и его направлением по азимуту). Найти площадь минимального по площади выпуклого многоугольника, описанного вокруг этой территории.

8.2 (8 б.) Задано множество точек на плоскости. Найти выпуклую оболочку этого множества, то есть выпуклый многоугольник с вершинами в некоторых точках этого множества, охватывающий все его точки.

8.3 (9 б.) Задано множество точек в трехмерном пространстве. Найти его выпуклую оболочку, то есть множество плоскостей – граней многогранника наименьшего объема, содержащего все заданные точки.

Указание. Каждая грань проходит как минимум через некоторые 3 точки заданного множества.

8.4 (9 б.) Из числового множества A мощности n выбрать все подмножества, сумма элементов каждого из которых лежит в пределах от a до b.

8.5 (8 б.) Найти расстояние от заданной точки до заданного треугольника на плоскости.

(12 б.) **Развитие задачи.** Расстояние между двумя треугольниками на плоскости. Взаимное расположение фигур произвольно (возможно пересечение).

8.6 (7 б.) **Перетекание массы.** На плоскости заданы n материальных точек. С некоторого момента точка с наименьшей массой исчезает, передавая свою массу ближайшей к ней точке. Так продолжается до тех пор, пока не останется одна точка. Реализовать этот процесс и найти оставшуюся точку.

8.7 (8 б.) Выпуклый многоугольник на плоскости задан своими вершинами, расположенными в произвольном порядке. Расположить вершины в порядке обхода по часовой стрелке.

8.8 (9 б.) Пусть n прямых на плоскости, заданных своими уравнениями, разбивают всю плоскость на многоугольники. Найти вершины наименьшего по площади (или по количеству вершин) многоугольника, содержащего заданную точку A.

8.9 (5 б.) Найти расстояние между двумя заданными множествами точек на плоскости.

8.10 (6 б.) Найти площадь многоугольника (не обязательно выпуклого), заданного координатами своих вершин на плоскости в порядке обхода по или против часовой стрелки.

8.11 (7 б.) Дано $3n$ точек на плоскости, причем никакие три из них не лежат на одной прямой. Построить множество n треугольников с вершинами в этих точках так, чтобы никакие два треугольника не пересекались и не содержали друг друга.

8.12 (11 б.) Задано n произвольных натуральных чисел. Найти все группы по k чисел, сумма которых равна заданному числу m.

8.13 (8 б.) Найти координаты вершин многоугольника, заданного системой неравенств $a_i x + b_i y \le c_i$, $i = 1, 2, ..., n$.

8.14 (8 б.) Выпуклый многоугольник на плоскости задан своими вершинами в порядке обхода по часовой стрелке. Изменит ли его вид неравенство $ax + by \le c$, и если да, найти вершины нового многоугольника.

8.15 (6 б.) Найти центр тяжести выпуклого многоугольника, заданного своими вершинами в порядке обхода по часовой стрелке.

8.16 (7 б.) У выпуклого $2n$-угольника, заданного координатами своих вершин на плоскости в порядке обхода по часовой стрелке, сократить вдвое число вершин, продолжив все нечетные стороны до пересечения.

8.17 (6 б.) Задано множество точек. Стянуть это множество к его центру тяжести, уменьшая расстояние между точками в $\alpha > 1$ раз. Центр тяжести должен оставаться неподвижным.

8.18 (7 б.) На плоскости задано n произвольно расположенных окружностей. Найти пару наименее удаленных из них.

8.19 (6 б.) Построить множество точек на плоскости, симметричных заданному множеству относительно заданной прямой $ax + by + c = 0$.

8.20 (6 б.) Из множества n-мерных векторов, заданных своими целочисленными координатами, найти пары ортогональных либо коллинеарных векторов.

8.21 (7 б.) Вершины многогранника в трехмерном пространстве пронумерованы в произвольном порядке. Каждая грань задана множеством номеров ее вершин. Найти пары вершин, образующие ребра.

8.22 (6 б.) Найти проекции каждой точки заданного множества в трех-мерном пространстве на заданную прямую.

8.23 (7 б.) Две строки матрицы назовем *похожими*, если совпадают множества чисел, встречающихся в этих строках. Найти все пары непохожих строк в заданной матрице.

8.24 (8 б.) Даны два массива числовых множеств: $A(m)$ и $B(n)$. Найти все пары множеств A_i и B_j таких, что $\mathrm{card}(A_i \cap B_j) \geq$ $\geq k$ (k задано).

8.25 (7 б.) *Медианой множества точек на плоскости* назовем прямую, которая делит множество на два подмножества одинаковой мощности. Найти горизонтальную и вертикальную медианы заданного множества, у которого никакие две точки не лежат на одной горизонтальной или вертикальной прямой.

8.26 (6 б.) Для заданного множества материальных точек найти прямую, которая делит это множество на два подмножества одинаковой мощности и относительно которой эти подмножества находятся в статическом равновесии. (Для равновесия достаточно прохождения прямой через центр тяжести всего множества.)

8.27 (8 б.) В числовом множестве A мощности n найти подмножество B мощности k такое, модуль суммы элементов которого минимален (числа в A имеют разные знаки).

(10 б.) **Развитие задачи.** Снять ограничение на мощность подмножества B.

8.28 (12 б.) Пусть n произвольных прямоугольников на плоскости заданы своими габаритами, то есть неравенствами:

$a_i \le x \le b_i$, $c_i \le y \le d_i$, $i = 1, 2, ..., n$. Найти площадь фигуры, полученной объединением прямоугольников.

Указание. Применение метода Монте-Карло для решения этой задачи или вычисление площади на мелкой равномерной сетке весьма трудоемко и не обеспечивает точного результата. Можно рекомендовать формулу: $s = s_1 - s_2 + s_3 - s_4 + ...$, где s_1 — сумма площадей заданных прямоугольников, s_2 — сумма площадей всех попарных пересечений, s_3 — сумма всех площадей пересечений по 3 прямоугольника и т. д. Каждое пересечение прямоугольников есть также прямоугольник.

Другой подход: разбить область на неравномерную сетку прямыми $x = a_i$, $x = b_i$, $y = c_i$, $y = d_i$, $i = \overline{1,n}$ и найти сумму всех «занятых» элементарных прямоугольников.

Тестирование: предусмотреть крайние случаи с очевидным результатом — непересекающиеся прямоугольники; наличие одного прямоугольника, включающего все остальные.

8.29. (7 б.) Вершины выпуклого многоугольника заданы в произвольном порядке своими координатами на плоскости. Убедиться в его выпуклости и найти периметр такого многоугольника.

8.30. (8 б.) Многоугольник на плоскости (не обязательно выпуклый) задан своими вершинами в порядке обхода по часовой стрелке. Проверить, лежит ли заданная точка A внутри или вне него.

Указание. Достаточно провести через точку A произвольную, например горизонтальную, прямую и найти точки ее пересечения с границей многоугольника. Если число таких точек, расположенных левее (или правее) точки A, четно, точка A лежит вне многоугольника (за исключением вырожденных случаев, когда прямая касается границы).

8.31. (9 б.) На клетчатом поле размером $m \times n$ задано мозаичное изображение нескольких фигур (не обязательно выпуклых). Клетки, принадлежащие разным фигурам, не соприкасаются. Найти количество фигур и площадь каждой из них.

8.32 (7 б.) Проверить тождество: $A \cap (B \cup C) = A \cap B \cup A \cap C$ на произвольных заданных числовых множествах A, B, C. Вывести полученное множество.

8.33 (7 б.) Для заданных множеств целых чисел A и B найти левую и правую части тождества и сравнить их: $(A \setminus B) \cup (B \setminus A) = (A \cup B) \setminus (A \cap B)$.

8.34 (7 б.) Найти объединение n заданных множеств целых чисел $A_1 \cup A_2 \cup \ldots \cup A_n$ и его мощность.

8.35 (7 б.) На клетчатом поле размером $m \times n$ задано матричное изображение некоторой геометрической области (не обязательно связной). Найти площадь и центр тяжести этой области.

8.36 (7 б.) **Черепашья графика.** Многоугольник (не обязательно выпуклый) задан следующим образом: длина очередной стороны, угол поворота к следующей стороне (положительный или отрицательный) и т. д. Убедиться в корректности данных (замкнутости ломаной, с учетом погрешности вычислений) и вычислить площадь многоугольника.

8.37 (6 б.) **Круглая оболочка.** Для заданного множества точек на плоскости найти круг минимального радиуса, содержащий все эти точки.

Примечание. Либо диаметр такого круга совпадает с диаметром множества, либо соответствующая окружность проходит через некоторые 3 точки множества.

8.38 (8 б.) **Треугольная оболочка.** Среди заданного множества точек на плоскости выбрать 3 таких, чтобы треугольник, вершинами которого эти точки являются, содержал бы наибольшее количество точек множества.

8.39 (7 б.) Найти расстояние между заданными на плоскости треугольником и окружностью (при их произвольном взаимном расположении).

9. Линейная алгебра и сжатие информации

При оцифровке (сканировании) графической, аудио-, видео- и другой информации, а также в некоторых экономико-математических задачах образуются массивы большой размерности, содержащие большие фрагменты из одинаковых элементов, то есть память используется неэффективно. Разработаны различные алгоритмы упаковки (сжатия) информации, например известный алгоритм Хаффмана, сжимающие без потерь информации занимаемый объем в несколько раз, а также еще более эффективные алгоритмы с несущественными потерями информации. В данном разделе предлагается опробовать простейшие алгоритмы сжатия.

Для хранения матриц, содержащих много нулевых элементов (разреженных матриц), используются различные способы упаковки, при которой хранятся только ненулевые элементы. Так, диагональная матрица $D(n,n)$ в упакованном виде хранится в одномерном массиве $C(n)$, содержащем только диагональные элементы. Верхнетреугольная или нижнетреугольная матрица упаковывается в одномерный массив, в котором последовательно записываются ненулевые части строк матрицы. Так же можно хранить симметричную матрицу.

Сильно разреженная матрица (содержащая много нулей, расположенных беспорядочно) может храниться в трех одномерных массивах $I(k)$, $J(k)$, $A(k)$, где i_l и j_l — соответственно номера строки и столбца, в которых стоит число a_l, то есть координаты этого числа в исходной матрице; $l = 1, 2, ..., k$.

Любая матрица $A(m,n)$ может храниться по строкам или по столбцам в одномерном массиве длиной $m \times n$ с соответствующим пересчетом индексов. Тогда элемент a_{ij} такой матрицы в одномерном массиве при построчном хранении стоит под номером $k = (i-1) \cdot n + j$; число k называют *приведенным индексом* этого элемента.

Двоичным массивом (одно- или многомерным) будем называть массив, состоящий из нулей и единиц. Такие массивы используются, например, для хранения двухтоновой графической информации. Задание целого типа (Integer) для хранения элементов такого массива вызовет как минимум 16-кратный перерасход памяти.

Один из способов сжатия двоичных массивов — представление их в виде двоичных кодов, где под каждый элемент массива выделяется 1 бит памяти. В другом способе хранятся только длины цепочек, целиком состоящих из нулей либо единиц. Так, например, последовательность 11110001100000111 представляется пятью числами: 4, 3, 2, 5, 3. Для их хранения используются переменные целого типа, либо байтовое или даже полубайтовое представление, при этом слишком длинные цепочки разбиваются на более короткие.

Сильно разреженную двоичную матрицу целесообразно хранить парами чисел $< i_l, j_l >$ — координатами единиц в матрице.

Метод Гаусса для решения системы линейных алгебраических уравнений $AX = B$, содержащей m уравнений и n неизвестных $(m \le n)$, сводится к последовательности элементарных преобразований над расширенной матрицей (A, B) системы, приводящих к эквивалентной системе. Следует заметить, что трудоемкость метода Гаусса имеет кубическую зависимость от размерности, в отличие от классических методов (с использованием алгебраических дополнений), имеющих факториальную зависимость. Формально метод может быть описан следующим алгоритмом.

1. *Прямой ход* — приведение матрицы к трапециевидной, для чего необходимо обнулить все элементы под главной диагональю левой квадратной части матрицы. Для этого можно воспользоваться следующей программой:

для всех $k = 1, 2, ..., m-1$ выполнить:
 для всех $i = k+1, ..., m$ выполнить:
  для всех $j = k+1, ..., n$ заменить:
   $a'_{ij} = a_{ij}a_{kk} - a_{ik}a_{kj}$;
  $a'_{ik} = 0$.

Если строки матрицы линейно зависимы, то в результате такого преобразования хотя бы одна строка матрицы станет нулевой.

2. *Обратный ход* — обнуление всех элементов над главной диагональю левой квадратной части матрицы.

Для всех $k = 2, 3, ..., m$ выполнить:
 для всех $i = 1, 2, ..., k-1$ выполнить:
  для всех $j = i, i+1, ..., n$ заменить:
   $a'_{ij} = a_{ij}a_{kk} - a_{ik}a_{kj}$.

Короче (и эффективнее) выглядит метод Гаусса—Жордана (метод жордановых исключений), используемый в следующей программе:

Для всех $k = 1, 2, .., m$ выполнить:
 для всех $i = 1, 2, .., m, i \neq k$ выполнить:
  для всех $j = 1, 2, ..., n$ заменить:
   $a'_{ij} = a_{ij} - \dfrac{a_{ik}a_{kj}}{a_{kk}}$;
 для всех $j = 1, 2, ..., n$ заменить: $a'_{kj} = \dfrac{a_{kj}}{a_{kk}}$.

Если же $a_{kk} = 0$, то в k-м столбце следует найти ненулевой элемент (его называют разрешающим) и поменять местами соответствующую строку с k-й.

В результате одного шага этого метода k-й столбец превращается в *единичный* с единицей на месте разрешающего элемента; по окончании алгоритма левая квадратная часть матрицы превращается в единичную.

Метод Гаусса с выбором разрешающего элемента предполагает на каждом шаге (для каждого значения k) выбор наибольшего по модулю элемента k-го столбца; это гарантирует устойчивость метода к накоплению погрешностей.

Для отыскания обратной A^{-1} для заданной квадратной матрицы A достаточно подвергнуть методу Гаусса—Жордана расширенную матрицу (A,E), где E — единичная матрица. В результате матрица A превращается в единичную, а на месте матрицы E получаем обратную.

Пример. Рассмотрим задачу 9.12: найти решение системы линейных алгебраических уравнений, матрица которой приведена к верхнетреугольному виду. Исходные данные задаются рандомизацией. Решение данной задачи может быть найдено с помощью следующей программы:

```
Program System;
   Uses Crt;
   var
         A:Array[1..500] of Real;
         X:Array[1..100] of Real;
         N:Integer;
         i,j:Integer;
         k:Integer;
         Sum:Real;
begin
   ClrScr;
   Write
  ('Введите размерность матрицы A[n,n]: ');
   ReadLn(N);
   for i:=1 to Sqr(N) do A[i]:=0;
   Randomize;
   Writeln
 ('Матрица в распакованном виде вместе с матрицей
 B');
   k:=1;
   for i:=1 to N do
         begin
               for j:=1 to i-1 do
                     Write(0:5);
               for j:=1 to N-i+2 do
                     begin
                           A[k]:=Random(10);
                           if Random(2)=0 then
                                 A[k]:=A[k]*(-1);
                                 {случайная смена знака}
                           Write(A[k]:5:0);
                           k:=k+1;
                     end;
                     WriteLn;
         end;
   WriteLn;
```

```
{Вычисление значений X[i]. не распаковывая A[i]}
for i:=1 to N do
      begin
            k:=k-1;
            Sum:=A[k];
            for j:=1 to i-1 do
                  begin
                        k:=k-1;
                        Sum:=Sum-X[j]*A[k];
                  end;
            k:=k-1;
            X[i]:=Sum/A[k];
      end;
   WriteLn;
   WriteLn('Массив решений');
   for i:=N downto 1 do
         Write(X[i]:8:4);
   ReadLn;
end.
```

Задачи по теме «Линейная алгебра и сжатие информации»

9.1 (5 б.) Выполнить операцию транспонирования прямоугольной матрицы $A(m,n)$, $m \neq n$, не выделяя дополнительного массива для хранения результата.

9.2 (6 б.) Некоторые из столбцов матрицы $K(m, n)$ являются собственными векторами матрицы $L(n, n)$. Найти эти столбцы и соответствующие им собственные числа.

9.3 (7 б.) Найти произведение матриц $A(m, n)$ и $B(n, k)$: $C(m,k) = A \times B$. Матрицы A, B и C описать как одномерные массивы, используя приведенные индексы.

9.4 (8 б.) В матрице A (m, n) выбран произвольный ненулевой минор k-го порядка $(k < m,\ k < n)$: $M_{IJ}^k = (a_{ij}, i \in I, j \in J)$. Для задания минора M_{IJ}^k достаточно указать множество I номеров строк и множество J номеров столбцов входящих в него элементов (оба множества мощности k). Распечатать (не вычисляя) все возможные миноры $(k+1)$-го порядка, окаймляющие данный минор.

9.5 (7 б.) Матрицу $A(n, n)$ разложить на слагаемые: $A = B + C + D$, где B — строго нижнетреугольная, C — диагональная, D — строго верхнетреугольная матрицы того же размера. Для экономии памяти матрицы B, C, D хранить в упакованном виде в одномерных массивах; распечатывать все треугольные матрицы в «треугольном» виде.

9.6 (8 б.) Решить задачу 9.5 для сильно разреженных упакованных матриц.

9.7 (7 б.) Найти определитель заданной матрицы n-го порядка методом Гаусса (в любой модификации).

9.8 (8 б.) Найти матрицу, обратную заданной $A(n, n)$, методом Гаусса (в любой модификации).

9.9 (7 б.) Привести матрицу $A(m, n)$к трапециевидной, используя метод Гаусса; при этом не обязательно $m \le n$.

9.10 (6 б.) Найти сумму двух сильно разреженных матриц $A(m, n)$ и $B(m, n)$, хранящихся в упакованном виде. Результат получить также в упакованном виде, а вывести — в обычном.

9.11 (7 б.) Как и в задаче 9.10, найти произведение двух сильно разреженных матриц, хранящихся в упакованном виде, не производя распаковку.

9.12 (7 б.) Матрица $A(n, n)$ системы линейных уравнений $AX = B$ приведена к верхнетреугольному виду и упакована в одномерный массив. Найти вектор решения X последовательной подстановкой, не распаковывая A.

9.13 (8 б.) Решить задачу 9.12 для сильно разреженной упакованной матрицы A (то есть матрица A системы линейных уравнений сильно разрежена, приведена к верхнетреугольному виду и упакована).

9.14 (8 б.) Решить систему уравнений $AX = B$ с квадратной матрицей A методом Гаусса. Предусмотреть возможность отсутствия и неединственности решения.

9.15 (7 б.) Найти ранг прямоугольной матрицы $A(m, n)$ методом Гаусса.

9.16 (7 б.) Методом Гаусса найти общее решение системы уравнений $AX = B$, если известно, что она совместна и имеет неединственное решение.

9.17 (7 б.) Задано m n-мерных векторов. Выбрать из них максимальное количество линейно независимых.

9.18 (7 б.) Найти произведение двух симметричных матриц A и B. Матрицы хранятся в одномерных массивах, где построчно записаны элементы, стоящие не ниже главной диагонали.

9.19 (7 б.) Найти произведение двух верхнетреугольных матриц, хранящихся в упакованном виде в одномерных массивах.

9.20 (8 б.) В упакованной сильно разреженной матрице удалить нулевые строки и столбцы, не производя распаковки. Результат напечатать в распакованном виде (в виде матрицы).

9.21 (6 б.) Известно, что строки матрицы $A(n, n)$, рассматриваемые как векторы, образуют базис в R^n. Проверить, является ли он ортогональным, и если это так, пронормировать его.

9.22 (8 б.) Решить задачу 9.21 для сильно разреженной упакованной матрицы.

9.23 (6 б.) Для заданной матрицы $A(m, n)$ найти ее произведение на транспонированную к ней AA' в упакованном виде (так как результат — симметричная матрица).

9.24 (4 б.) Для заданной сильно разреженной упакованной матрицы $A(m, n)$ найти след матрицы AA' (*следом* назовем сумму элементов главной диагонали).

9.25 (10 б.) Найти k-ю степень A^k заданной квадратной матрицы $A(n, n)$, выполнив минимальное число матричных умножений.

9.26 (8 б.) Для заданного вектора $X(n)$ и матрицы $A(n, n)$ вычислить значение *квадратичной формы* $y = X'AX$. Матрица A симметричная, поэтому хранится только ее верхнетреугольная часть в упакованном виде.

9.27 (8 б.) Каждое слагаемое алгебраического выражения второго порядка от n переменных $z(x)=\sum_{k=1}^{m}\alpha_k x_{i_k} x_{j_k}$, $i_k \le n,\ j_k \le n$ задано тройкой чисел $<i_k, j_k, \alpha_k>$, $k = 1, 2, \ldots, m$. Привести подобные (с учетом коммутативности умножения) и получить симметричную матрицу A квадратичной формы $z(x) = X'AX$.

9.28 (6 б.) Заданы коэффициенты многочленов: $P_n(x) = a_n x^n + \ldots + a_1 x + a_0$, $Q_m(x) = b_m x^m + \ldots + b_1 x + b_0$. Найти коэффициенты произведения этих многочленов $R_{n+m}(x) = P_n(x) \cdot Q_m(x) = c_{n+m} x^{n+m} + \ldots + c_1 x + c_0$.

9.29 (9 б.) Заданный неупакованный двоичный массив сжать, используя полубайтовое представление длин цепочек.

9.30 (9 б.) Двоичный массив, представленный в виде двоичных кодов, преобразовать, используя байтовое представление длин цепочек.

9.31 (9 б.) Решить задачу, обратную задаче 9.30, то есть массив в байтовом представлении длин цепочек нулей и единиц преобразовать в двоичный массив.

9.32 (9 б.) Для заданного массива (можно взять любой файл произвольной природы), рассматривая его как двоичное кодирование, вычислить коэффициенты экономии или перерасхода памяти (отношение использованных объемов при разных способах хранения) при хранении его в виде длин цепочек с полубайтовым, байтовым и двухбайтовым представлением длин цепочек.

9.33 (8 б.) Сильно разреженную упакованную единичную матрицу представить длинами цепочек нулей, предварительно развернув ее в одномерный массив.

9.34 (8 б.) Решить задачу, обратную задаче 9.33, то есть сильно разреженную единичную матрицу, представленную в одномерном массиве длинами цепочек, получить в упакованном виде.

9.35 (8 б.) Сильно разреженную двоичную матрицу, хранящуюся в виде двоичных кодов (число столбцов кратно восьми), представить в упакованном виде.

10. Комбинаторика и теория вероятностей

В некоторых задачах этого раздела предусматривается использование датчиков случайных чисел. Иногда такие датчики реализуются аппаратно, однако чаще используются программные датчики, в которых псевдослучайные числа вырабатываются детерминированным алгоритмом, но по своим характеристикам они «похожи» на случайные. В языках программирования такие датчики носят имя RANDOM, RND и т. п.

Если ни в ЭВМ, ни в языке нет датчика, можно воспользоваться простейшим алгоритмом, например: $\xi_{i+1} = \text{fl}(\xi_i / z_i + \pi)$; $z_{i+1} = z_i + 10^{-8}$. Здесь для $\xi_0 = 0$; $z_0 = 0{,}011$; ξ_{i+1} — очередное псевдослучайное число, равномерно распределенное на отрезке $[0;1]$; $\text{fl}(x)$ — дробная часть числа x.

Выражение $x = a + (b-a)\xi$ дает действительное число, равномерно распределенное на отрезке $[a;b]$, а выражение $k = \text{ent}[m + (n - m + 1 - 10^{-8})\xi]$ — целое число, равновероятно выбранное из множества $\{m, m+1, ..., n\}$. Здесь $\text{ent}(x)$ — целая часть числа x.

С помощью датчиков случайных чисел реализуется широко известный метод статистических испытаний, или метод Монте-Карло, для решения многих задач. Суть метода состоит в следующем. Пусть в прямоугольнике на плоскости: $a \le x \le b$; $c \le y \le d$ задана некоторая фигура M и требуется найти ее площадь. В прямоугольнике выбирается n случайных точек («бросков»); пусть m из них попало в область M. Площадь фигуры оценивается как $S_M \approx \frac{m}{n} S_p$, где $S_p = (b-a)(d-c)$ — площадь прямоугольника; причем точность оценки будет тем выше, чем больше число испытаний n (погрешность квадратично уменьшается с ростом числа испытаний).

Для реализации одного «броска» достаточно получить случайные x и y — координаты точки, лежащей в прямоугольнике, и проверить, не попала ли точка в исследуемую область.

Пример. Рассмотрим задачу 10.18 о случайном расположении заданного количества слонов на шахматной доске при условии, что ни один из них не угрожает другим. Решение может выглядеть следующим образом:

```
Program Slony_10_18;
   uses Crt;
   type Point=Record x,y:Byte; end;
   var A:Array[1..8] of Point;
          i,j,r: Byte;
          m:Byte;
   function  Proverka(i,j,r:Byte):Boolean;
          var f:Byte;
   begin
          Proverka:=True;
          for f:=1 to r do
                 if abs(A[f].x-i)=abs(A[f].y-j) then
                        Proverka:=False;
   end; {Proverka}
begin
   ClrScr;
   repeat
          Write('Сколько слонов? ');
          ReadLn(m);
          if not (m in [1..8]) then
                 WriteLn('Надо от 1 до 8');
   until m in [1..8];
   Randomize;
   for r:=1 to m do
          repeat
                 j:=Random(8)+1;
                 i:=Random(8)+1;
                 if Proverka(i,j,r-1) or (r=1) then
                        begin
                               A[r].x:=i;
                               A[r].y:=j;
                        end;
          until Proverka(i,j,r-1) or (r=1);
```

```
    for i:=1 to m do
    WriteLn('(',A[i].x,':',A[i].y,')');
    ReadKey;
end.
```

Задачи по теме «Комбинаторика и теория вероятностей»

10.1. (7 б.) Из массива $A(n)$ выбрать случайным образом m различных элементов и поместить их в массив $B(m)$. Рассмотреть случаи: а) $m << n$; б) $m \approx n$.

10.2. (8 б.) «Перемешать» массив $T(m)$, переставив его элементы случайным образом. В массиве $K(m)$ запомнить позиции элементов до перемешивания.

10.3. (7 б.) «Перемешать» матрицу $Z(m,n)$, переставив все ее элементы случайным образом.

10.4. (8 б.) Переставить все строки и столбцы матрицы $F(m, n)$ случайным образом. В массивах $K(m)$ и $L(n)$ запомнить места, которые каждая строка и каждый столбец занимали до перестановки.

10.5. (9 б.) Задан массив $K(m)$ попарно различных целых чисел. Получить все перестановки этих чисел.

10.6. (9 б.) Для заданного n получить все возможные перестановки чисел $1, 2, ..., n$.

10.7. (8 б.) Для выборки случайных чисел, содержащейся в массиве $A(n)$, построить гистограмму распределения, содержащую m интервалов.

Для справки. Гистограммой (или *диаграммой*) *распределения* называется массив из m элементов, значение каждого из которых равно количеству (или доле из общего числа) чисел из A, попавших в соответствующий интервал. Ширина интервала определяется как $(a_{max} - a_{min})/m$.

10.8. (7 б.) Массив $Z(n)$ разбить на m фрагментов случайной длины. Фрагменты записать в матрицу A, содержащую m строк.

10.9. (9 б.) Массив $X(n)$ разбит на m фрагментов. Длины фрагментов собраны в массиве $K(m)$. Переставить фрагменты случайным образом и преобразовать соответственно массивы X и K.

10.10 (8 б.) На плоскости заданы своими целочисленными координатами n точек. Найти всевозможные группы по 3, 4, ... точки, лежащие на одной прямой.

10.11 (7 б.) Задан список n абитуриентов. Сформировать из них случайным образом

$$m = \text{ent}\left(\frac{n}{25} + 0{,}999\right)$$

групп так, чтобы количество абитуриентов в разных группах отличалось не более чем на 1 и не превышало 25. Вместо фамилий можно использовать шифры.

10.12 (9 б.) **Детерминированная задача о раскрое.** Стержень длиной l_0 нужно разрезать на стержни длиной $l_1, l_2, ..., l_k$. Количество отрезков каждого типа не ограничено. Найти все способы раскроя и вычислить неиспользованный остаток в каждом способе.

10.13 (9 б.) **Вероятностная задача о раскрое.** Как и в задаче 10.12, найти m различных случайно выбранных способов раскроя.

10.14 (9 б.) Для заданных m и n найти все сочетания по m из чисел $1, 2, ..., n$.

10.15 (14 б.) **Задача о ранце.** Известен вес $p_1, p_2, ... p_n$ каждого из n предметов. Как разместить их в наименьшем числе, если грузоподъемность каждого рюкзака равна q? Например: рюкзаков $q = 9$; $P = (4; 4; 3; 3; 2; 2)$.

10.16 (6 б.) Матрицу $A(m,n)$ случайным образом заполнить разными целыми числами от 1 до $m \times n$.

10.17 (6 б.) **Латинский квадрат.** Заполнить квадратную матрицу $K(n,n)$ случайным образом числами $1, 2, ..., n$ так, чтобы в каждой строке и в каждом столбце не было одинаковых чисел.

10.18 (5 б.) Расположить на шахматной доске случайным образом $m \le 8$ слонов так, чтобы они не угрожали друг другу (слон бьет по диагонали).

10.19 (9 б.) **Морской бой.** Напечатать заготовку для игры в «морской бой», то есть расположить случайным образом 15 «линейных» кораблей (1 — 5-клеточный, 2 — 4-клеточных, 3 — 3-клеточных, 4 — 2-клеточных, 5 — 1-клеточных) на поле размером 10×10 так, чтобы они не касались друг друга.

10.20 (6 б.) Из массива $A(n)$ удалить m наудачу (случайно) выбранных элементов. Порядок следования оставшихся элементов сохранить.

10.21 (6 б.) Для заданной функции $y = f(x)$ с известной первообразной методом Монте-Карло вычислить определенный интеграл

$$I = \int_a^b f(x)dx.$$

Оценить зависимость статистической погрешности от числа испытаний n.

10.22 (7 б.) Утверждается, что отношение объемов гиперсферы в R^n и описанного вокруг нее гиперкуба быстро падает с ростом n (уже при $n = 6$ оно составляет менее 5%). Проверить это с помощью метода Монте-Карло.

10.23 (8 б.) Многогранник в R^3 задан системой линейных неравенств: $a_i x + b_i y + c_i z \le d_i$, $i = 1, 2, \ldots, m$. Известны также его «габариты»: $x_{\min} \le x \le x_{\max}$; $y_{\min} \le y \le y_{\max}$; $z_{\min} \le z \le z_{\max}$. Оценить объем многогранника методом Монте-Карло.

10.24 (7 б.) **Горе-синоптик.** В некоторой местности погода ведет себя непредсказуемо; известны лишь следующие постоянные вероятности событий: p_1 — в течение дня будет дождь; p_2 — будет снег; p_3 — без осадков; $p_1 + p_2 + p_3 = 1$. Синоптики в прогнозе на сегодня попросту повторяют то, что произошло в погоде вчера. Методом Монте-Карло оценить вероятность правильного прогноза для заданных p_1, p_2, p_3 на протяжении длительного периода.

10.25 (6 б.) На плоскости заданы несколько произвольных кругов (с различными случаями пересечения и наложения). Методом Монте-Карло найти площадь фигуры, полученной объединением кругов.

10.26 (6 б.) Трехмерное тело образовано объединением нескольких сфер произвольного размера и взаимного расположения. Найти объем этого тела, используя метод Монте-Карло.

10.27 (12 б.) **12 монет.** Известен способ, как за три взвешивания на весах без гирь среди 12 монет найти фальшивую (отличающуюся от остальных по весу в неизвестную сторону). Для этого при каждом взвешивании на чаши весов нужно класть по 4 монеты. Реализовать такую идентификацию: человек загадывает номер фальшивой монеты и ее особенность (легче или тяжелее других). ЭВМ организует взвешивание, спрашивает, куда отклонилась стрелка весов при каждом опыте, и находит номер фальшивой монеты.

(12 б.) **Развитие задачи.** Реализовать идентификацию за минимальное число взвешиваний для произвольного числа монет.

10.28 (8 б.) Перемешать массив $A(n)$ случайным образом так, чтобы ни один из элементов гарантированно не остался на своем прежнем месте и имел бы одинаковые шансы занять любое из остальных мест. Разрешается использовать рабочий массив такой же размерности, что и основной.

10.31 (10 б.) Имитировать перетасовку новой колоды игральных карт в 52 листа многократным применением операций сдвига и «врезки» так, чтобы никакие две рядом лежащие карты не сохранили бы свой первоначальный порядок.

Примечание. *Операция врезки* состоит в делении колоды на две неравные подколоды и слиянии частей случайного объема, взятых из обеих подколод.

11. Элементы численного анализа

В задачах этого раздела требуется реализовать заданные численные методы, экспериментально исследовать условия и скорость сходимости методов. В условиях задач, как правило, дается лишь основная идея каждого метода. Рекомендуется самостоятельно познакомиться с обоснованием и условиями применения метода по специальной литературе, например:

1. *Демидович Б. П., Марон И. А.* Основы вычислительной математики. — М.: Наука, 1970.
2. *Калиткин Н. Н.* Численные методы. — М.: Наука, 1978.
3. *Бахвалов Н. С.* Численные методы. — М.: Наука, 1973.
4. *Бронштейн И. М., Семендяев К. А.* Справочник по математике. — М.: Наука, 1978.

Вычисление функций и их производных, используемых в задачах, рекомендуется оформлять в виде подпрограмм, так чтобы можно было подставлять любую функцию, не меняя самой программы. Погрешность, начальные условия и параметры алгоритма задаются вводом. Там, где это возможно, рекомендуется тестировать алгоритм на примерах, для которых известно или может быть найдено аналитически точное аналитическое решение (например, результат численного интегрирования сравнить с известной первообразной). Кроме собственно программирования и тестирования алгоритма полезно провести исследование предложенного метода на сходимость и скорость сходимости при разных условиях, устойчивость для разных функций и разных начальных условий.

Пример. Рассмотрим решение задачи 11.6 — вычисление определенного интеграла по формуле прямоугольников. В качестве подынтегральной функции возьмем $y = e^{-x^2}$, для которой первообразная не выражается в элементарных функциях (это одна из разновидностей интеграла вероятности), поэтому завершение итерационного процесса — увеличение числа отрезков разбиения — запишем по условию близости соседних приближений, а не близости точного и приближенного результатов. Кроме того, встроенная функция Exp(x) не может быть вычислена при больших (по модулю) значениях аргумента, поэтому вводим ограничение на задаваемые пределы интегрирования – не более чем $[-5; 5]$, так как подынтегральная функция за пределами этого отрезка практически равна нулю. Решение может быть представлено с помощью следующей программы:

```
program Integral;
{вычисление определенного интеграла с заданой погрешностью
методом прямоугольников}
{Автор - А. Жариков}
   uses Crt;
   const
         n0=10; {начальное количество отрезков}
   var
         n1:LongInt;
         {требуемое количество отрезков
               разбиения}
         a,b,i1,i2,eps:Real;
         okey:Boolean;
   Function Fun(x:Real):Real;
         {подынтегральное выражение}
   begin
         Fun:=Exp(-Sqr(x));
   end; {Fun}
   Function Rec(n:LongInt;a,b:Real):Real;
         {вычисление интеграла}
   var
         i:LongInt;
         h,s:Real;
   begin
         s:=0;
         h:=Abs((b-a)/n);
         {длина отрезка разбиения}
         for i:=1 to n-1 do
         s:=s+Fun(a+i*h);
         {сумма значений функции на концах
               отрезков}
```

```
            Rec:=h*s; {значение интеграла}
    end; {Rec}
begin
    ClrScr;
    WriteLn
            ('Введите пределы интегрирования:');
    repeat
            ReadLn(a,b);
            okey:=(a<b) and (abs(a)<=5) and
                    (abs(b)<=5);
                    {функция Exp(x) невычислима при
                            больших значениях аргумента }
            if not okey then
                    Writeln('Пределы интегрирования',
                            ' неверны');
    until okey;
    WriteLn ('Введите погрешность',
                    ' (не менее 0.000001):');
    repeat
            ReadLn(eps);
            okey:=(eps>0) and (0.000001<=Abs(e));
            if not okey then
                    WriteLn('Повторите, пожалуйста!');
    until okey;
    Writeln('Задано: a=',a:7:3,' b= ',b:7:3,
            ' eps= ',e:7:6);
    i1:=0;
    n1:=n0;
    repeat {основной цикл}
            i2:=i1;
            i1:=Rec(n1,a,b);
            n1:=2*n1;
                    {увеличение количества отрезков}
    until Abs(i1-i2) <= eps;
                    {проверка на погрешность}
    Writeln('интеграл ',i1:13:7,
            ' найден на ',n1,' отрезках');
    ReadLn;
end.
```

Тестирование. Для испытаний возьмем данные с известным результатом. Так, известно, что

$$\int_{-\infty}^{+\infty} e^{-x^2}dx = \sqrt{\pi}, \text{ или } \int_{0}^{+\infty} e^{-x^2}dx = \frac{\sqrt{\pi}}{2} = 0{,}8862269.$$

Задачи по теме «Элементы численного анализа»

11.1 (5 б.) **Метод простой итерации.** Решить методом итераций уравнение вида $x = \varphi(x)$. Очередное приближение корня находится по формуле $x_n = \varphi(x_{n-1})$; начальное приближение x_0 задается произвольно. Обратите внимание, что метод сходится, если только $|\varphi'(x)| < 1$.

11.2 (5 б.) **Метод Ньютона.** Решить методом Ньютона уравнение $f(x) = 0$. Очередное приближение корня находится по формуле:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

11.3 (6 б.) **Метод Эйткена–Стеффенсона** (развитие метода простой итерации). Найти решение уравнения $x = \varphi(x)$ методом Эйткена–Стеффенсона, в котором от заданного начального x_0 три очередных приближения находятся по формулам: $x_{n+1} = \varphi(x_n)$; $x_{n+2} = \varphi(x_{n+1})$;

$$x_{n+3} = \frac{x_n x_{n+2} - x_{n+1}^2}{x_n - 2x_{n+1} + x_{n+2}}.$$

Процесс продолжается до достижения одного из условий: $|x_{n+3} - x_{n+2}| < \varepsilon$ или $x_n - 2x_{n+1} + x_{n+2} = 0$.

11.4 (5 б.) **Метод дихотомии.** Решить уравнение $f(x) = 0$ методом деления отрезка пополам (методом дихотомии). На каждой итерации отрезок $[a, b]$ делится пополам и выбирается та из половин, на концах которой функция $f(x)$ имеет значения разных знаков.

11.5 (5 б.) **Метод хорд.** Найти решение уравнения $f(x) = 0$ на заданном отрезке $a \le x \le b$ методом хорд, в котором очередное приближение находится по формуле:

$$x_{n+1} = x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n).$$

В задачах на численное интегрирование (задачи 11.6–11.8) определенный интеграл требуется найти с заданной точностью, для чего вычисление по формуле метода рекомендуется проводить многократно, каждый раз уменьшая шаг интегрирования в 2 раза, пока разница между соседними приближениями не станет меньше заданной погрешности.

11.6 (5 б.) **Метод прямоугольников.** Вычислить определенный интеграл $I = \int_a^b f(x)\,dx$ методом прямоугольников: $I \approx \frac{b-a}{n}(y_0 + y_1 + ... + y_n)$, где n — количество отрезков разбиения; $y_0, y_1, ..., y_n$ — значения функции на концах отрезков.

11.7. (5 б.) **Метод трапеций.** Вычислить определенный интеграл $I = \int_a^b f(x)\,dx$ методом трапеций: $I \approx \frac{b-a}{2n}(y_0 + 2y_1 + +...+ 2y_{n-1} + y_n)$, где n — количество отрезков разбиения; $y_0, y_1, ..., y_n$ — значения функции на концах отрезков.

11.8 (6 б.) **Метод Симпсона.** Вычислить определенный интеграл $I = \int_a^b f(x)\,dx$ по формуле Симпсона: $I \approx \frac{b-a}{6n}(y_0 + +4y_1 + 2y_2 + ... + 4y_{2n-1} + y_{2n})$, где $2n$ — количество отрезков разбиения; $y_0, y_1, ..., y_{2n}$ — значения функции $f(x)$ на концах отрезков.

11.9 (7 б.) **Метод Эйлера.** Найти приближенное решение обыкновенного дифференциального уравнения второго порядка $y'' = f(x, y, y')$ с заданными начальными условиями $y(a) = y_0$ и $y'(0) = p_0$ методом Эйлера на отрезке $[a, b]$ с постоянным шагом h. Значения функции $y(x)$ и ее производной $p(x) = y'$ в узловых точках вычисляются по формулам: $y_{i+1} = y_i + hp_i$; $p_{i+1} = p_i + hf(x_i, y_i, p_i)$, $i = 1, 2, ...$

11.10 (8 б.) **Метод Ньютона.** Составить таблицу значений функции $y(x)$, которая задана неявно уравнением $F(x, y) = 0$. При фиксированном x уравнение $F(x, y) = 0$ можно решить методом Ньютона (см. задачу 11.2).

11.11 (5 б.) **Линейная интерполяция.** Функция $y = f(x)$ задана таблично в массиве $Y(n)$ при соответствующих значениях аргумента, хранящихся в неупорядоченном массиве $X(n)$, не содержащем одинаковых значений. Используя формулу линейной интерполяции:

$$y = y_{i-1} + \frac{x - x_{i-1}}{x_i - x_{i-1}}(y_i - y_{i-1}),\ i = 2, 3, \ldots, n,$$

построить таблицу значений функции на отрезке, содержащем все заданные значения аргумента, с постоянным заданным шагом h. Можно упорядочить пару массивов $< X(n), Y(n) >$ по возрастанию x.

11.12 (6 б.) **Интерполяционный многочлен Лагранжа.** Значения функции $y = f(x)$ заданы таблично в массиве $Y(n)$ при соответствующих значениях аргумента в упорядоченном массиве $X(n)$. Найти значение функции в произвольной точке x по формуле Лагранжа:

$$y = L_n(x) = \sum_{i=1}^{n} y_i \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}.$$

11.13 (7 б.) **Метод Рунге–Кутта.** Найти приближенное решение обыкновенного дифференциального уравнения $y' = f(x, y)$, $y(a) = y_0$ методом Рунге–Кутта пятого порядка на отрезке $[a, b]$ с заданным постоянным шагом h. Значения функции $y(x)$ в узловых точках вычисляются

по формуле: $y_{i+1} = y_i + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$, $i = 0, 1, 2, \ldots$,

где $k_1 = f(x_i, y_i)$; $k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_1\right)$;

$k_3 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_2\right)$; $k_4 = f\left(x_i + h, y\ \ + hk_3\right)$.

11.14 (7 б.) **Метод золотого сечения.** Найти минимум заданной функции $y = f(x)$ на заданном отрезке $[a, b]$ методом золотого сечения. Для этого на отрезке $[a, b]$ находятся две точки: $x_1 = a + (1 - \tau)(b - a)$ и $x_2 = a + \tau(b - a)$,

где $\tau = \frac{-1 + \sqrt{5}}{2} = 0{,}6180339$.

Если $f(x_1) > f(x_2)$, отбрасывается интервал $[a, x_1]$, иначе — $[x_2, b]$; процесс продолжается до достижения заданной точности.

11.15 (6 б.) **Метод парабол.** Найти минимум заданной функции $y = f(x)$, двигаясь от заданной точки x_0 по методу парабол:

$$x_{i+1} = x_i - \frac{h}{2}\frac{f(x_i + h) - f(x_i - h)}{f(x_i + h) - 2f(x_i) + f(x_i - h)}, \; i = 0, 1, \dots,$$

пока не будет достигнута заданная точность.

11.16 (8 б.) Для заданной матрицы $A(n, n)$ найти обратную A^{-1}, используя итерационную формулу:

$$A_k^{-1} = A_{k-1}^{-1}\left(2E - A \cdot A_{k-1}^{-1}\right),$$

где E — единичная матрица; $A_0^{-1} = E$. Итерационный процесс заканчивается, если для заданной погрешности ε справедливо $\left|\det\left(A \cdot A_k^{-1}\right) - 1\right| \le \varepsilon$.

12. Алгоритмы обработки символьной информации

Текстом будем называть длинную последовательность символов, несущую информацию по любой отрасли знаний. Текст состоит из слов (в том числе сокращений, обозначений, чисел, формул и так далее); разделителем между словами служат: пробел, символы перевода строки, табуляции или сочетание этих символов. Считается, что вместо пробела может стоять несколько пробелов. В тексте можно выделить фразы. Признаком конца фразы будем считать один из следующих символов: точка, восклицательный или вопросительный знак, многоточие, хотя реально это и не всегда так.

Текст может быть записан одной длинной строкой либо разбит на строки фиксированной или неопределенной (но ограниченной) длины. В последнем случае перенос осуществляется целыми словами. Строки группируются в абзацы; признаком начала абзаца служит красная строка, содержащая в начале, как правило, 5 пробелов или знак табуляции. В задачах на орфографию прописные и строчные буквы считать неразличимыми.

Для испытания программ следует подготовить (или найти готовый) файл, содержащий текст или словарь со всеми характерными случаями. Результат работы программы, как правило, рекомендуется помещать в текстовый файл с параллельным просмотром на экране.

Пример. Рассмотрим задачу 12.30 — процедура угадывания слова по его шаблону, в котором высвечиваются некоторые буквы, как в игре «Поле чудес». Одно из решений выглядит следующим образом:

```
program Pole_12_30;
    uses Crt;
    var st,temp: String;
        ch: Char;
        i: Byte;
        a: array[1..256] of Byte;
        name: String;
        f: Text;
        kol: Word;
    function Ask(i:Word):String;
        var temp:String;
            j:Byte;
        begin
            Assign(F,name);
            Reset(f);
            j:=0;
            while j<i do
                begin
                    j:=j+1;
                    ReadLn(F,temp);
                end;
            Ask:=temp;
            Close(F);
        end; {Ask}
    procedure Shablon;
        var c: Byte;
    begin
        c:=trunc(40-Length(st)/2);
        GotoXY(c+1,15);
        for i:=1 to Length(st) do
            if (st[i]<>' ') then
                Write('*') else
                Write(' ');
    end; {Shablon}
    procedure Win;
    begin
        GotoXY(10,10);
        WriteLn('Поздравляю, Вы выиграли!');
        ch:=#27;
        ReadKey;
    end; {Win}
    function Summa:Byte;
        var x,i:Byte;
    begin
```

```
        x:=0;
        for i:=1 to Length(st) do
              x:=x+a[i];
        Summa:=x;
    end; {Summa}
    procedure Open(i:Byte);
        var c:Byte;
    begin
        a[i]:=1;
        c:=Trunc(40-Length(st)/2);
        GotoXY(c+i,15);
        Write(st[i]);
    end; {Open}
begin
    ClrScr;
    name:='u:\t\f.txt';
    Assign(F,name);
    Reset(F);
    while not EOF(f) do
        begin
              ReadLn(f,st);
              kol:=kol+1;
        end;
    Close(f);
    Randomize;
    st:=ask(Random(Kol)+1);
    Shablon;
    temp:='';
    repeat
        ch:=ReadKey;
        if (ch<>#13) then temp:=temp+ch;
        GotoXY(5,25);
        Write(' ');
        GotoXY(5,25);
        Write(temp);
        if ch=#13 then
              begin
                    GotoXY(5,25);
                    ClrEOL;
              end;
        if (ch=#13) and (Length(temp)=1) then
        begin
              ch:=temp[1];
              for i:=1 to Length(st) do
                    if (ch=st[i]) and (ch<>' ') then
                          open(i);
              temp:='';
```

```
        end;
        if (ch=#13) and (Length(temp)>1) and
            (temp<>st) then temp:='';
        if (ch=#13) and (temp=st) then
            for i:=1 to Length(st) do open(i);
        if summa=Length(st) then Win;
    until ch=#27;
end.
```

Задачи по теме «Алгоритмы обработки символьной информации»

12.1 (7 б.) Текст записан одной длинной строкой. Признаком красной строки служит символ $. Переформатировать текст в 60-символьные строки, формируя абзацы.

12.2 (8 б.) Текст, сформированный построчно, выровнять по правому краю так, чтобы каждая строка заканчивалась знаком препинания или одним пробелом. Выравнивание осуществить, вставляя дополнительные пробелы между словами (равномерно по всей строке).

12.3 (8 б.) Дан текст программы на каком-либо алгоритмическом языке и словарь зарезервированных слов этого языка (в английской транскрипции). Преобразовать текст, записав все зарезервированные слова прописными буквами, а остальные конструкции (имена и так далее) — строчными. Русские буквы (имена, литералы) не заменять.

12.4 (7 б.) Программа, записанная 80-байтовыми строками, в последних 8 байтах содержит номер строки. Строки упорядочены по номерам, но не обязательно с шагом 1. Поступает изменение к программе в таком же виде. Вставить новые строки взамен имеющихся старых или между ними.

12.5 (8 б.) Текст записан 60-символьными строками, содержит следующие знаки корректуры: $ — сделать красную строку; # — удалить следующее слово; @ — удалить следующую фразу. Произвести указанную корректировку, переформатируя строки в пределах абзаца.

12.6 (8 б.) Часто встречающаяся ошибка начинающих наборщиков — дважды записанное слово. Обнаружить и исправить такие ошибки в тексте, записанном 80-символьными строками; переформатировать строки в пределах абзаца.

12.7 (6 б.) Стихотворный текст (в строке не более 80 символов) имеет четырехстрочную строфу. Записать его «лесенкой» (по одному слову в строке), вставляя пустую строку после каждого четверостишья.

12.8 (7 б.) **Поздравления.** По заданному списку фамилий напечатать каждому упомянутому в списке поздравление к определенному празднику. Чтобы избежать шаблона, перечень желаемых благ выбирать как случайное подмножество из заготовленного списка (например, здоровья, счастья, продвижения по службе, долголетия и т. д.). Можно сделать переменным и название праздника — для универсальности программы.

12.9 (7 б.) Имеется список членов коллектива с указанием принадлежности каждого к различным общественным организациям (профком, ученый совет, общество книголюбов, федерация пентикса и т. д.). Напечатать приглашение всем членам на очередное заседание указанной организации. Задается только вид организации, место и время сбора.

12.10 (7 б.) Заданный список русских фамилий (вместе с именами и отчествами) упорядочить по алфавиту. Проверить (и исправить, если нужно) написание собственных имен с прописных букв.

12.11 (8 б.) В заданном тексте подсчитать частоту использования каждого буквосочетания, слова и словосочетания из заданного списка.

12.12 (10 б.) В заданном тексте найти самое длинное слово и самую длинную фразу.

12.13 (7 б.) **Морзянка.** Вводимый с клавиатуры или из файла текст перевести в последовательность точек и тире с помощью азбуки Морзе. Результат можно иллюстрировать звуком.

Для справки — азбука Морзе:

А, A	.-	Б, B	-...	В, W	.--	Г, G	--.	Д, D	-..
Е, Ё, E	.	Ж, V	...-	З, Z	--..	И, I	..	Й, J	.---
К, K	-.-	Л, L	.-..	М, M	--	Н, N	-.	О, O	---
П, P	.--.	Р, R	.-.	С, S	...	Т, T	-	У, U	..-
Ф, F	..-.	Х, H	....	Ц, C	-.-.	Ч	---.	Ш	----
Щ, Q	--.-	Ъ	.--.-.	Ы, Y	-.--	Ь, X	-..-	Э	..-..
Ю	..--	Я	.-.-	1	.----	2	..---	3	...--
4	....-	5	.....	6	-....	7	--...	8	---..
9	----.	0	-----	.	.-.-.-	,	--..--	:	---...
?	..--..	‘, ’	.----.	-	-....-	/	-..-.	,	-.--.-

12.14 (6 б.) Обнаружено, что в тексте пропущены некоторые слова и словосочетания. Эти слова и словосочетания представлены отдельным списком в том порядке, в каком должны быть вставлены. Места вставки отмечены в тексте символом $. Откорректировать текст.

12.15 (6 б.) Текст не содержит собственных имен и сокращений, набран с использованием прописных и строчных русских букв. Проверить то, что все фразы (и только они) начинаются с прописной буквы; при необходимости откорректировать текст.

12.16 (6 б.) Текст, не содержащий собственных имен и сокращений, набран полностью прописными русскими буквами. Заменить все прописные буквы, кроме букв, стоящих после точки, строчными буквами.

12.17 (6 б.) Имеется большой словарь русских слов. Найти в нем *слова-палиндромы* («перевертыши»), одинаково читающиеся как слева направо, так и справа налево, например, АННА, ШАЛАШ и так далее.

(10 б.) **Развитие задачи.** В словаре встречаются не только слова-палиндромы, но и фразы-палиндромы, например: «А роза упала на лапу Азора». Найти и их.

12.18 (8 б.) В имеющемся словаре найти слова, которые могут быть полностью составлены из других слов с помощью конкатенации, например: «БАЛКОН» = «БАЛ» + «КОН»; «БАРСУК» = «БАР» + «СУК».

12.19 (8 б.) В имеющемся словаре найти группы слов, записанных одними и теми же буквами и отличающиеся только их порядком, то есть перестановкой, например, (КОМАР, КОРМА).

12.20 (5 б.) В имеющемся словаре найти пары слов (*анаграммы*), при прочтении каждого из которых в обратном направлении образуется другое слово пары, например, (ПОЛК, КЛОП); (БАР, РАБ).

12.21 (6 б.) Задан набор ключевых слов, а также текст, в котором хранится длинный список названий книг и научных работ. Выбрать названия, содержащие хотя бы одно из заданных ключевых слов.

12.22 (7 б.) **Орнамент.** На небольшом поле (например, размером 6×8 позиций) случайным образом или преднамеренно с помощью нескольких выбранных символов, обозначающих цвета, сформировать элемент орнамента (лучше осе- или центросимметричный). Повторяя этот элемент многократно по горизонтали и (или) вертикали, получить весь орнамент. Если элемент формируется случайно, а вывод осуществляется на дисплей, предусмотреть визуальную оценку гармоничности результата.

12.23 (8 б.) Один из способов идентификации автора литературного произведения — подсчет частоты вхождения отдельных слов. В заданном тексте найти 20 наиболее часто встречающихся слов с указанием количества использования каждого из них.

12.24 (7 б.) В русском языке, как правило, после букв Ж, Ч, Ш, Щ пишется И, А, У, а не Ы, Я, Ю. Проверить заданный текст на соблюдение этого правила и исправить ошибки (с учетом исключений: ЖЮРИ, БРОШЮРА, ПАРАШЮТ).

12.25 (6 б.) Каждую 80-байтовую строку заданного текста (например, стихотворного произведения) *отцентрировать*, то есть обеспечить осевую симметрию текста на экране добавлением пробелов слева.

12.26 (5 б.) Как в предыдущей задаче, отцентрировать вводимый с клавиатуры текст: первый символ помещается в 40-ю позицию; второй — в 41-ю; появление третьего и каждого последующего нечетного символа вызывает удаление одного пробела слева. Так продолжается до конца строки; ввод следующих строк — аналогично.

12.27 (9 б.) Перенос. Примем следующие правила переноса русских слов:

- в каждой из разделяемых частей должно быть более одной буквы, из которых хотя бы одна — гласная;
- нельзя разделять согласную и следующую за ней гласную;
- буквы Й, Ь, Ъ считать согласными, но перенос после них допустим.

В каждом из вводимых слов поставить все возможные знаки переноса, например: СЕ-ЛЬ-С-КО-ХО-ЗЯЙ-С-Т-ВЕ-Н-НАЯ. Строчные и прописные буквы считать неразличимыми.

12.28 (8 б.) По правилам пунктуации пробел может стоять после, а не перед каждым из следующих знаков: . , ; : ! ?)] } ...; перед, а не после знаков: ([{. Заданный текст проверить на соблюдение этих правил и при необходимости исправить. Вместо пробела может быть перевод строки или знак табуляции.

12.29 (7 б.) **Скобки.** Текст (например, арифметическое выражение) содержит многократно вложенные круглые скобки. Исправить его, оставив скобки первого уровня круглыми, второго — заменить на квадратные, третьего и последующих — на фигурные. Убедиться в корректности использования скобок.

Пример. $\ln(\sin((a+b)c-d))$ заменить выражением $\ln\left\{\sin\left[\left(a+b\right)c-d\right]\right\}$.

12.30 (6 б.) **Поле чудес.** ЭВМ загадывает (случайно выбирает из имеющегося словаря) слово или словосочетание и выводит рамку-шаблон для его отгадывания. Человек отгадывает букву или называет все слово; ЭВМ анализирует результаты отгадывания. Реализовать этот процесс.

12.31 (8 б.) Строка содержит арифметическое выражение, состоящее из целых чисел и знаков операций: +, -, *, / (без скобок). Проверить корректность выражения (в смысле последовательности чисел и знаков операций).

(12 б.) **Развитие задачи.** В случае корректности вычислить значение выражения (с учетом приоритетов операций).

12.32 (7 б.) Любителям кроссвордов. В слове угаданы некоторые буквы, надо рассмотреть все возможные варианты. Для этого пользователь вводит «шаблон» слова, заменяя неизвестные буквы пробелом или знаком подчеркивания; компьютер из файла-словаря выбирает все слова, удовлетворяющие этому шаблону.

12.33 (9 б.) В файле дан исходный текст программы на каком-либо языке программирования (Паскаль, Фортран, С и другие). Уровнем комментированности текста будем считать отношение объема комментариев к объему всего текста (в байтах). Определить уровень комментированности данного текста (с указанием языка программирования).

12.34 (7 б.) **«Балда».** Для заданного достаточно длинного слова найти в имеющемся словаре все слова, в которых использованы только буквы, имеющиеся в заданном слове (с учетом кратности вхождения).

12.35 (8 б.) Для заданного номера месяца вывести все приходящиеся на этот месяц праздничные дни, например: 1 января — Новый год, 7 января — Рождество и т. д. Справочную информацию — список праздников — хранить в текстовом файле.

12.36 (7 б.) **Разнобуквица.** Из имеющегося словаря выбрать наиболее длинное слово, в котором все буквы разные, например: ЛЕЙКОПЛАСТЫРЬ, НЕРЯШЛИВОСТЬ, ЧЕТЫРЁХДЮЙМОВКА.

12.37 (9 б.) В текстовом файле представлены в виде таблицы результаты соревнований по прыжкам в длину. В каждой строке файла записаны фамилия с инициалами спортсмена и через пробелы — его результаты в трех попытках. Найти трех призеров соревнования, не выделяя массива для хранения всех результатов.

Пример.

```
Иванов И.И. 720 732 735
Петров П.П. 722 727 730
Сидоров С.С. 721 733 738
```

12.38 (9 б.) В текстовом файле представлены названия лекарств, срок их годности, количество стандартов и стоимость одного стандарта (через пробелы), например:

```
Аспирин 15.04.99 127 1.35
Настойка валерьяны 27.09.01 23 5.40
```

Вывести на экран данные о лекарствах с истекшим сроком годности на сегодняшний день (с учетом перехода через столетие) и подсчитать общую стоимость таких медикаментов.

12.39 (7 б.) Для заданного текста построить гистограмму (см. задачу 10.7) распределения длин слов.

12.40 (6 б.) **«Латинизатор кириллицы».** При интернет-общении с русской диаспорой в других странах часто возникают проблемы отсутствия кириллицы у зарубежных респондентов, а также слабое знание иностранных языков у соотечественников. Один из выходов — набор русских слов похожими по начертанию буквами латинского алфавита. Среди прописных русских букв таких насчитывается одиннадцать: А, В, Е, К, М, Н, О, Р, С, Т, Х. В заданном русском тексте выбрать те слова, которые без искажения могут быть написаны латинскими буквами, например:

```
СВЕТА РОЕТ РОВ.
ВОВКА СЕЕТ ОВЕС.
```

12.41 (8 б.) Текстовый файл содержит маленькую записную книжку, в которой в беспорядке записаны данные о знакомых: фамилия, имя, дата рождения в формате `ДД.ММ.ГГ`. Поля в строке можно разделить пробелами или использовать структуру типа Record в Паскале. Для заданной сегодняшней даты: ДД.ММ найти знакомого, к дню рождения которого надо незамедлительно готовиться.

12.42 (6 б.) **Гороскоп.** В древнеяпонском календаре был принят 60-летний цикл из пяти 12-летних циклов, обозначенных цветами: зеленый, красный, желтый, белый и черный. В подциклах каждый год имел название одного из животных: крысы, быка, тигра, кролика, дракона, змеи, лошади, овцы, обезьяны, петуха, собаки и кабана. Началом очередного цикла был 4-й год нашей эры — год зеленой крысы. Для заданного номера года найти его название по древнеяпонскому календарю.

12.43 (10 б.) Для двух заданных строк символов найти самую длинную общую подстроку. Пробелы и знаки препинания игнорировать, строчные и прописные буквы считать неразличимыми. Например, строки: «Дай вилку! Бок севрюжий кончается» и «Чемпионский кубок достался не нам» содержат общую подстроку «кубок».

13. Элементарная машинная графика

В задачах этого раздела требуется построить предложенное статическое (неподвижное) изображение на экране дисплея. В необходимых случаях предусмотреть масштабирование для эффективного использования площади экрана. Графики функций строить в декартовой системе координат с обоснованным выбором масштаба изображения. Полезно вместо точечного изображения использовать кусочно-линейную аппроксимацию с достаточно мелким шагом.

При защите программ оценивается не только качество и надежность программы, но и качество постановки (конкретизация, детализация проработки) и дизайн полученного изображения.

Пример. Рассмотрим задачу 13.33: заполнение экрана полуокружностями заданного диаметра, напоминающими рисунок рыбной чешуи. В приведенной ниже программе расчет количества полуокружностей по горизонтали и вертикали ведется в зависимости от разрешения дисплея — из требования мобильности программы.

```
program Fish;
{Автор - В. Жариков}
{Заполнить экран "рыбьей чешуёй"}
uses Graph, Crt;
const a=180; b=270; c=360;
   {углы для рисования дуг}
var  gd,gm,r,x,y,n,m,i,j,s,q,mx,my:
   Integer;
        okey:Boolean;
```

```
begin
   ClrScr:
   Writeln
         ('Введите размер (радиус) чешуи от 5 до 100:'):
   repeat
         Readln(r);
         okey:=(r>=5) and (r<=100);
         if not okey then
               Writeln('Повторите, пожайлуста!'):
   until okey:
   gd:=Detect:
   InitGraph(gd.gm, 'c:\bp\bgi\'):
   SetColor(6);
   mx:=GetMaxX;
   my:=GetMaxY: {рзрешение экрана}
   n:=mx div (2*r):
         {количество "чешуек", входящих на экран по
               горизонтали}
   m:=my div r:
         {количество "чешуек", входящих на
               экран по вертикали}
   s:=(mx-n*2*r) div 2:
         {отступ от края по горизонтали}
   x:=s:
   q:=(my-m*r) div 2:
         {отступ от края по вертикали}
   y:=q:
   RecTangle(x,y,mx-x,my-y): {рамка}
   my:=-1: {индикатор чётности номера
         строки}
   for i:=1 to m do
         begin
               my:=-my:
               for j:=1 to n do
                     begin
                           Arc(x+my*r,y,b,c,r):
                           Arc(x+r,y,a,b,r):
                           x:=x+2*r;
                     end;
               x:=r*(i mod 2)+s:
                y:=y+r;
         end:
   Readln:
   CloseGraph:
end.
```

Тестирование сводится к испытанию программы при различных (допустимых) размерах элементарных чешуек.

Задачи по теме «Элементарная машинная графика»

13.1 (6 б.) **Почтовый индекс.** Заданное шестиразрядное десятичное натуральное число изобразить цифрами по 9-сегментному шаблону, используемому при почтовой индексации.

13.2 (6 б.) Изобразить произвольный домик (деревянный или кирпичный), отобразив материал стен (бревна, кирпичи) и крыши (шифер, черепица и т. п.). Размеры домика, кирпичей, окон и т. д. задаются вводом.

13.3 (5 б.) Распределение скорости ветра по каждому из восьми направлений задано массивом из восьми чисел. Построить «розу ветров» с указанием направлений.

13.4 (6 б.) Для некоторой функции $y = f(x, a)$ построить семейство графиков для различных значений параметра a.

13.5 (6 б.) В декартовых координатах построить семейство кривых по параметрическим уравнениям вида $x = \varphi(t, a)$; $y = \psi(t, a)$ при различных значениях параметра a.

13.6 (6 б.) Построить семейство кривых, заданных следующим уравнением в полярных координатах: $\rho = f(\varphi, a)$, при различных значениях параметра a.

13.7 (9 б.) Уравнение кривой $F(x, y) = 0$ не удалось представить в явном виде. Построить такую кривую.

13.8 (5 б.) Изобразить на экране шахматную доску (вместе с буквенно-цифровым обозначением горизонталей и вертикалей) и случайным образом расставленные на ней шашки.

13.9 (7 б.) Разработать и получить на экране рисунок обложки какого-либо учебника вместе с названием, фамилиями авторов, рисунками, отражающими суть предмета, и так далее.

13.10 (7 б.) **Паркет.** Заполнить экран рисунком паркета «елочка» из прямоугольных дощечек заданного размера.

13.11 (6 б.) Построить *фигуру Лиссажу*: $x = \alpha \sin(\omega_x t + \varphi_x)$; $y = \beta \sin(\omega_y t + \varphi_y)$. Параметры $\alpha, \beta, \omega_x, \omega_y, \varphi_x, \varphi_y$ задаются вводом.

13.12 (6 б.) По правилам черчения изобразить проекцию какого-либо тела и проставить размеры.

13.13 (8 б.) Для заданного n построить (разместить на экране) все n-конечные звезды. Например, для $n = 13$ насчитывается 5 различных звезд.

13.14 (6 б.) Функция $y = f(x)$ задана таблично в виде массивов $X(n)$ и $Y(n)$. Шаг по x непостоянен; массив X упорядочен по возрастанию. Построить график функции, используя линейную интерполяцию между точками.

13.15 (7 б.) Задан массив $K(n)$ распределения населения по профессиям (название профессии и соответствующе ей количество человек). Изобразить распределение в виде круговой (секторной) диаграммы с необходимыми надписями.

13.16 (8 б.) **Кубик Рубика.** Получить в аксонометрии или диметрии кубик Рубика в любом разобранном виде. Цвета элементарных кубиков можно изобразить разной штриховкой или полутонами.

13.17 (8 б.) **Перспектива.** Изобразить уходящую вдаль улицу, состоящую из двух рядов однотипных домов. Учесть невидимые части зданий.

13.18 (7 б.) Заполнить экран кругами заданного радиуса, расположив их как можно плотнее, симметрично относительно границ экрана.

13.19 (8 б.) **Алгебра и гармония.** Разработать орнамент на основе каких-либо математических кривых и заполнить ими экран.

13.20 (7 б.) **Паутина.** Получить на экране рисунок паутины с центром в произвольной (заданной) точке, с произвольным числом лучей. Паутина образована лучами и многоугольниками.

13.21 (8 б.) Множество точек на плоскости задано своими координатами. Построить в декартовых координатах эти точки и выпуклую оболочку множества, то есть многоугольник минимальной площади, охватывающий все точки.

13.22 (10 б.) **Графический редактор.** Используя курсор-перекрестие, реализовать возможность ручного построения фигур с помощью операций: задать цвет (если он есть), передвинуться, начертить отрезок, «резиновый прямоугольник», стереть отрезок, заштриховать (залить) область. Код операции задается с клавиатуры.

13.23 (6 б.) **Графическое интегрирование.** Для заданного дифференциального уравнения вида $y' = F(x, y)$ и начального условия $y(0) = y_0$ построить кусочно-линейный график решения $y = f(x)$, вычисляя y с заданным постоянным шагом по x: $y_i = y_{i-1} + h \cdot F(x_{i-1}, y_{i-1})$.

13.24 (4 б.) **Медленное печатание.** Заданный текст печатать крупными буквами с некоторыми паузами между буквами, сопровождая каждую букву звуковым щелчком. Перенос осуществлять только целыми словами.

13.25 (7 б.) Построить мозаику из правильных шестиугольников заданного размера, закрасив их разными цветами или применив разные типы штриховок (полутона).

13.26 (8 б.) **Счеты.** Заданное число (не обязательно целое) отложить на бухгалтерских счетах, изображенных на экране.

13.27 (8 б.) **Микрокалькулятор.** Заданное число изобразить как на индикаторе микрокалькулятора, используя для цифр 7-сегментный шаблон.

13.28 (7 б.) Изобразить «рог изобилия», являющийся стилизацией закрученного бараньего рога. При построении полезно использовать математические кривые (спирали, эллипсы и т. д.).

13.29 (7 б.) **Подсолнух.** Рисунок на шляпке подсолнуха представляет собой семейство логарифмических спиралей, закрученных в разные стороны. Получить такой рисунок. Между прочим, количество «правых» и «левых» спиралей есть два соседних числа Фибоначчи.

13.30 (8 б.) Изобразить на экране достаточно сложный цветок (георгин, ромашка со случайным числом лепестков, калина, василек и так далее).

13.31 (7 б.) **Дерево.** Для заданного *n* построить двоичное дерево, содержащее *n* уровней (например, генеалогическое).

13.32 (6 б.) **Чешуя.** Заполнить экран рисунком рыбьей чешуи с заданным размером элементарных чешуек.

13.33 (7 б.) **Пирамида.** Одинаковые трубы в количестве *n* штук уложены возможно более компактно — пирамидой. Получить на экране вид пирамиды с торца для произвольного числа *n*.

13.34 (7 б.) **Всходы.** Изобразить на экране всходы какого-либо растения, посаженного квадратно-гнездовым способом на грядке или на поле. Учесть перспективу.

13.35 (8 б.) Изобразить в зацеплении две шестерни (зубчатых колеса) какого-либо механизма; диаметры шестерен и количество зубьев задаются.

(10 б.) **Развитие задачи.** Изобразить последовательно зацепленными *n* шестерен с заданными параметрами; при необходимости смасштабировать чертеж до размеров экрана.

5 Зак. № 767

14. Элементы компьютерной мультипликации

В задачах этого раздела требуется построить динамическое изображение, получающееся последовательным формированием кадров мультипликации. Следует позаботиться о масштабировании времени, то есть по возможности форсировать вывод или задержать его. Рекомендуется использовать спрайты или другой подобный механизм для переноса фрагментов изображения по фону.

Пример. Рассмотрим задачу 14.20 (кривая дракона). Задается единственное исходное данное — порядок кривой, на экране формируется любопытная кривая из серии фракталов. Рекомендуемый порядок кривой — не более 20.

```
Program Dragon2;{Автор - Дудник}
    uses Graph, Crt;
    var horfact, vertfact, x_max, y_max: Real;
            x__max, y__max, x, y, dx, dy, i, p, xmin,
            xmax, ymin, ymax, ixC, iyC: Integer;
            fx, fy, f, xC, yC: Real;
            GrDriver, GrMode: Integer;
            n: LongInt;
    function  IX(x:Real):Integer;
            begin
                    IX:=Round(x*horfact+0.5);
            end; {IX}
    function  IY(Y:Real):Integer;
            begin
                    IY:=Round(y*vertfact+0.5);
            end; {IY}
    procedure  GMove(x,y:Real);
```

```
        begin
                MoveTo(IX(x),IY(y));
        end; {GMove}
procedure  Draw(x,y:Real);
        begin
                LineTo(IX(x),IY(y));
        end; {Draw}
function  Xreal(x:Integer):Real;
        begin
                Xreal:=xC+f*(x-ixC);
        end; {Xreal}
function  Yreal(y:Integer):Real;
        begin
                Yreal:=yC+f*(y-iyC);
        end; {Yreal}
procedure  Step(r:Integer;prescan:Boolean);
        var
                t,dx1,dy1:Integer;
        begin
                if prescan then
                        begin
                                x:=x+dx;y:=y+dy;
                                if(x<xmin)then  xmin:=x;
                                if(x>xmax)then  xmax:=x;
                                if(y<ymin)then  ymin:=y;
                                if(y>ymax)then  ymax:=y;
                        end
                else
                        begin
                                dx1:=dx div 4;
                                dy1:=dy div 4;
                                Draw(xreal(x+dx1),yreal(y+dy1));
                                x:=x+dx;
                                y:=y+dy;
                                Draw(xreal(x-dx1),yreal(y-dy1));
                        end;
                if(r=1)then
                        begin
                                t:=dx;dx:=-dy;dy:=t;
                        end
                else
                        begin
                                t:=dx;dx:=dy;dy:=-t;
                        end;
        end; {Step}
procedure  Curve(prescan:boolean);
        var  i,j,r:LongInt;
```

```
begin
      i:=1;
      while(i<=n)do
            begin
                  i:=i+1;
                  j:=i;
                  while((j and 1)=0) do
                        j:=j shr 1;
                        r:=j and 3;
                        Step(r,prescan);
            end;
end; {Curve}
begin
      WriteLn('Порядок кривой?');
      ReadLn(p);
      n:=1;
      for i:=1 to p do n:=n*2;
      GrDriver := Detect;
      InitGraph(GrDriver, GrMode, 'P:\bp\BGI');
      x_max:=10;
      y_max:=7;
      x__max:=GetMaxX;
      y__max:=GetMaxY;
      horfact:=x__max/x_max;
      vertfact:=y__max/y_max;
      x:=0;
      y:=0;
      dx:=4;
      dy:=0;
      xmin:=10;
      xmax:=-10;
      ymin:=10;
      ymax:=-10;
      Curve(true);
      fx:=x_max/(xmax-xmin);
      fy:=y_max/(ymax-ymin);
      if(fx<fy)then f:=fx*0.7 else f:=fy*0.7;
      ixC:=(xmin+xmax)div 2;
      iyC:=(ymin+ymax)div 2;
      xC:=x_max/2;
      yC:=y_max/2;
      x:=0;
      y:=0;
      dx:=4;
      dy:=0;
      GMove(xreal(x+dx div 4),yreal(y));
      Curve(false);
```

```
        ReadKey;
        CloseGraph;
    end;
end.
```

Задачи по теме «Элементы компьютерной мультипликации»

14.1 (9 б.) **Круги на воде.** Экран изображает бассейн с водой, в который бросили камень (в заданных координатах). От камня пошли круги, которые, дойдя до стенок бассейна, отражаются от них. Реализовать эту динамическую картину.

(13 б.) **Развитие задачи.** Эффект «блинчиков» — отскоков камня от поверхности воды с последующими падениями.

14.2 (8 б.) **Открытка.** Разработать и реализовать на экране поздравительную открытку к празднику (Новый год, день рождения и так далее) с динамическими элементами (мигающая гирлянда, салют, горящие свечи, мигающие надписи и так далее).

14.3 (8 б.) **Плакат.** Разработать и реализовать на экране политический или экологический плакат с динамическими (движущимися или мигающими) эффектами.

14.4 (8 б.) **Телезаставка.** Разработать и реализовать на экране заставку к одной из популярных телепрограмм («Поле чудес», «КВН» и так далее).

14.5 (8 б.) **Ветряная мельница.** Изобразить на экране работающую ветряную мельницу. Плоскость вращения крыльев не обязательно параллельна плоскости экрана.

14.6 (8 б.) **Электронные часы.** Изобразить на экране работающие электронные часы с цифровым индикатором (каждая цифра изображается на 7-сегментном шаблоне). При недоступности встроенного таймера ЭВМ реализовать его с помощью настроенных циклов, задавая стартовое время при запуске программы.

(10 б.) **Развитие задачи.** Индикация даты, дня недели.

14.7 (7 б.) **Стрелочные часы.** Изобразить на экране работающие часы со стрелочным индикатором (3 стрелки). При отсутствии таймера — имитировать его, как в задаче 14.6.

(9 б.) **Развитие задачи.** Индикация в окне даты и дня недели.

14.8 (8 б.) **Песочные часы.** Изобразить на экране действующие песочные часы. Учесть законы физики: количество песка, вытекающего из верхней колбы в единицу времени, постоянно и равно количеству песка, притекающего в нижнюю колбу. Установка времени перетекания песка производится при запуске программы.

14.9 (8 б.) **Бильярд.** На экране — изображение бильярдного стола с одним шаром. Задается направление и начальная скорость шара после удара кием. Реализовать движение шара после удара до попадания его в лузу или до прерывания с клавиатуры.

(12 б.) **Развитие задачи.** Учет сил трения. Можно высвечивать след (траекторию) шара.

14.10 (8 б.) **Футбольный мяч.** Изобразить движение футбольного мяча после удара (задается начальное положение мяча и вектор скорости). В процессе полета мяч ударяется о пол, потолок и стены, теряя при каждом ударе часть энергии.

(12 б.) **Развитие задачи.** Учет сопротивления воздуха. Можно высвечивать траекторию мяча.

14.11 (6 б.) **Броуновское движение.** Частица (от заданной начальной точки) совершает хаотичное движение, двигаясь в случайном направлении на случайное расстояние (в пределах экрана). Получить на экране траекторию движения частицы до прерывания с клавиатуры.

(10 б.) **Развитие задачи.** *n* частиц совершают хаотичное движение в пространстве, ограниченном размерами экрана (на каждом шаге — в случайном направлении на

случайное расстояние). Удары частиц друг о друга (при пересечении траекторий) и о стенки экрана считать абсолютно упругими. Построить траектории движения частиц (для каждой частицы — свой цвет).

14.12 (6 б.) **Пьяница.** В случайных точках местности расположены несколько столбов, некоторые из них соединены заборами. Пьяница с равной вероятностью делает шаг вперед, назад, вперед — вправо или вперед — влево (под 45°). Натолкнувшись на столб или забор, он падает (на некоторое время). Изобразить траекторию его движения.

14.13 (6 б.) **Морзянка.** Заданный текст (вводимый с клавиатуры или из файла) представить последовательностью точек и тире с помощью азбуки Морзе (азбука Морзе приведена в задаче 12.13). Если возможно, сопроводить вывод звуковой индикацией.

14.14 (7 б.) Изобразить в действии кривошипно-шатунный механизм парового двигателя или двигателя внутреннего сгорания.

14.15 (7 б.) **Орнамент из квадратов.** Построить квадрат заданного размера. Каждую сторону разделить в заданном отношении $m : n$; полученные точки суть вершины нового квадрата. И так далее до заполнения внутренности квадрата. Заполнить такими квадратами весь экран.

14.16 (7 б.) **Орнамент из треугольников.** Как в задаче 14.15, заполнить экран орнаментом из правильных треугольников.

14.17 (6 б.) **Конвейер.** Изобразить действующий конвейер, транспортирующий какие-либо однотипные предметы.

14.18 (8 б.) **Кипящая жидкость.** Экран — сосуд с кипящей жидкостью. На дне в случайной точке образуется пузырек; при движении вверх он растет, а дойдя до поверхности — лопается. Если два пузырька соприкасаются, они сливаются в один. Реализовать этот процесс.

14.19 (8 б.) **Брошенная палка.** Известна угловая скорость вращения и вектор начальной линейной скорости брошенной палки. Изобразить ее в движении до падения.

(13 б.) **Развитие задачи.** Учесть сопротивление воздуха и отскоки от границ экрана.

14.20 (9 б.) **Кривая дракона.** Для заданного n построить кривую дракона порядка n. Описание кривой можно найти, например, в книге: *Абрамов С. А.* и др. «Задачи по программированию». — М.: Наука, 1988, с. 174. Кривая состоит из единичных отрезков, соединенных под прямым углом по следующему правилу. Каждой кривой порядка n ставится в соответствие последовательность из нулей и единиц (назовем ее двоичной формулой), где единица соответствует повороту налево, а нуль — повороту направо. Кривая первого порядка имеет формулу «1». Для получения формулы кривой каждого следующего порядка следует к формуле предыдущего порядка справа приписать единицу, после чего справа приписать формулу предыдущего порядка, в которой средняя единица заменена нулем. Получаем для второго порядка формулу «110», для третьего — «1101100», для четвертого — «110110011100100» и т. д.

14.21 (8 б.) **Вращающийся кубик.** Изобразить в движении кубик заданного размера, равномерно вращающийся вокруг вертикальной оси.

(12 б.) **Развитие задачи.** Кубик, вращаясь, удаляется в бесконечность.

14.22 (7 б.) **Затмение луны.** Изобразить на экране звездное небо, полную луну и медленно надвигающуюся на нее тень Земли; затем — медленное открытие диска Луны.

(10 б.) **Развитие задачи.** На звездном небе — Млечный путь, падающие звезды, искусственные спутники.

14.23 (7 б.) **Сумерки.** Изобразить на экране произвольный пейзаж, натюрморт или интерьер. Затем случайными точками или прямыми заполнять экран до полного исчезновения картины (удобнее реализовать эффект управлением палитрой).

(10 б.) **Развитие задачи.** Обратный процесс: рассвет или проявление фотоизображения.

14.24 (8 б.) **Калейдоскоп.** Построить в центре экрана треугольник заданного размера и заполнить его произвольным (жестко заданным, случайным или задаваемым с клавиатуры) изображением. Произвести многократное зеркальное отражение изображения от каждой стороны треугольника до заполнения всего экрана.

14.25 (8 б.) **Паровоз.** Получить на экране картину, которую видит машинист движущегося поезда: рельсы, шпалы, столбы, придорожные строения и так далее.

(12 б.) **Развитие задачи.** Учесть повороты, стрелки, изменение скорости поезда, встречные составы и так далее.

14.26 (7 б.) **Заиавес.** Изобразить финальную сцену какого-либо театрального представления: на экране: произвольное изображение; справа и слева на него надвигается занавес. На занавесе — надпись: «КОНЕЦ».

14.27 **Атом.** Изобразить модель атома произвольного химического элемента: ядро и электроны, вращающиеся по своим орбитам. Распределение электронов по орбитам задается.

(13 б.) **Развитие задачи.** В подготовленном файле хранится распределение электронов по орбитам для всей системы Менделеева; пользователь задает только номер или обозначение химического элемента.

14.28 (6 б.) **Маятник.** Получить изображение движущегося математического маятника. Длина маятника и начальное положение задаются.

(9 б.) **Развитие задачи.** Учесть сопротивление воздуха; вместо маятника изобразить качели.

14.29 (7 б.) **Флаг.** Изобразить развевающийся на ветру флаг (например, российский трехцветный).

14.30 (7 б.) **Реклама.** Разработать и реализовать динамическую рекламу какого-либо товара или услуги.

14.31 (9 б.) **Счеты.** Изобразить бухгалтерские счеты и реализовать на них тренажер, демонстрирующий операции сложения и вычитания. Числа и знак операции вводятся с клавиатуры. Предусмотреть задержку для наглядности поразрядных операций.

14.32 (9 б.) **Салют.** Реализовать на экране картину праздничного салюта: взлет, разрывы, падение пиротехнических ракет и их осколков (из нескольких стволов). Световые эффекты желательно сопроводить звуковыми.

14.33 (8 б.) **Месяц-Месяцович.** Цикл движения Луны по небосводу составляет 28 суток; орбиту движения можно считать постоянной. Реализовать в ускоренном темпе процессы движения, рождения, роста и «старения» Луны на фоне звездного неба.

14.34 (8 б.) На экране — мишень (спортивная или армейская), содержащая 10 концентрических колец с соответствующими очками. В произвольный момент или по нажатию клавиши происходит «выстрел», в результате которого на мишени появляется пробоина. Процесс продолжается до исчерпания запаса патронов или до нажатия клавиши; подсчитывается число очков. Пользователь может задать параметры случайного распределения.

(12 б.) **Развитие задачи.** Точка прицеливания (перекрестие прицела) задается с клавиатуры или мышью; прицел изначально «сбит»; можно дать несколько пристрелочных патронов.

14.35 (7 б.) **Наперсточник.** Компьютер — в роли ведущего; пользователь — «лоха». На экране — 3 наперстка, под одним из них — шарик (на старте он виден). Ведущий в заданном темпе меняет местами 2 наудачу выбранных наперстка; по окончании процесса пользователь должен угадать, где шарик.

14.36 (10-20 б.) **Игра «Жизнь».** На бесконечном клетчатом поле произвольно расположены несколько «особей»; клетки 8-связные, то есть каждая клетка имеет 8 соседних. Шаг – это «смена поколения», при котором действуют следующие правила:

- выживание: особь, у которой есть 2 или 3 соседа, выживает;
- гибель: особь, имеющая более 3 соседей, погибает от перенаселения; особь, имеющая менее 2 соседей, гибнет от одиночества;
- рождение: если у пустой клетки ровно 3 соседних особи, в ней рождается особь.

При хорошем разрешении экрана и быстродействии компьютера можно наблюдать интересные эффекты, описанные, например, в книге Гарднера М. «Крестики-нолики» (М.: Мир, 1988): устойчивые конфигурации, пульсары, движение, столкновение, поглощение конфигураций и пр. Исходное состояние популяции (количество и расположение особей) можно задать случайно или из заранее заготовленного файла.

15. Сортировка и слияние массивов

Сортировкой информационной структуры (массива, списка, файла) называется преобразование исходной структуры путем перестановки ее элементов для достижения упорядоченности по заданному признаку порядка.

Ниже приводятся различные признаки порядка, которые мы будем использовать в задачах этого раздела. Одномерный массив $A(n)$ называется *упорядоченным* по:

$$\left\{\begin{matrix}\text{возрастанию}\\ \text{неубыванию}\\ \text{убыванию}\\ \text{невозрастанию}\end{matrix}\right\}, \text{ если для любого } i = 2, 3, \ldots, n \left\{\begin{matrix}a_i > a_{i-1}\\ a_i \ge a_{i-1}\\ a_i < a_{i-1}\\ a_i \le a_{i-1}\end{matrix}\right\}.$$

Массив $A(n)$ лексикографически $\left\{\begin{matrix}\text{больше}\\ \text{меньше}\end{matrix}\right\}$ массива $B(n)$, если существует $k = 1, 2, \ldots, n$ такое, что $a_i = b_i$ для всех $i < k$ и, кроме того, $\left\{\begin{matrix}a_k > b_k\\ a_k < b_k\end{matrix}\right\}$. Пример — сравнение двух слов по алфавиту входящих в них букв.

Матрица называется *упорядоченной по одному из признаков порядка*, если ее строки попарно лексикографически удовлетворяют этому признаку.

Сортировкой массива (матрицы) по заданному признаку порядка называется процесс достижения этого признака порядка с помощью перестановки элементов массива (строк матрицы). Если в задаче не указан метод сортировки, нужно выбрать наиболее эффективный.

Слиянием двух массивов (из которых хотя бы один упорядочен) называется объединение их в один массив с сохранением упорядоченности. Слияние гораздо менее трудоемко, чем сортировка, так как упорядоченность уже частично достигнута.

Линейным односвязным списком, или просто списком, будем называть совокупность элементов вида пар <link, info>, где info — информационная часть, хранящая собственно элемент списка; link — ссылочная часть — указатель на следующий по порядку элемент списка. Ссылки реализуются посредством указателей (например в Паскале), а если такой возможности нет, список можно реализовать двумя массивами $X(n)$ и $L(n)$, где x_i содержит информационную часть любого типа, а l_i — ссылка в виде индекса следующего элемента списка.

Изменение порядка следования элементов, а также операции вставки, удаления и замены не требуют реорганизации всего списка; достаточно локального изменения ссылок.

В одной паре массивов $< X, L >$ могут храниться два и более списков; для доступа к каждому из них надо знать индекс или ссылку на его вершину — первый элемент. Последний элемент имеет ссылку «в никуда»: например, достаточно положить $l_n = 0$.

Пример. Рассмотрим задачу 15.14 — упорядочение строк заданной матрицы лексикографически по убыванию. Одно из решений (на примере рандомизированных исходных данных) выглядит следующим образом:

```
program Z_15_14_:
   const
   n = 100:
   type
         Line = Array [1..n] of Real:
         Matrix = Array [1..n] of        line:
   procedure Swap (var a.b :Line:g:Byte):
   var
         i:Byte:
         l:Line:
         bb:Boolean:
```

```
    begin
          bb:=false;
          i:=1;
          while not bb and (i<=g) do
                begin
                      if a[i]>b[i] then
                            bb:=true
                      else
                            if a[i]<b[i] then
                                  begin
                                        l:=a;
                                        a:=b;
                                        b:=l;
                                        bb:=true
                                  end
                            else
                                  inc(i);
                end;
    end; {Swap}
    var
          i,j,v,g:Byte;
          m:Matrix;
          s_:String[3];
          code_:Integer;
begin
    Randomize;
    repeat
          Write('вертикаль:');
          ReadLn(S_);
          val(S_,V,Code_);
          if Code_ <> 0 then
                writeln('не то, повтори'#7);
    until Code_=0;
    repeat
          Write('горизонталь:');
          ReadLn(S_);
          val(S_,g,Code_);
          if Code_ <> 0 then
                writeln('не то, повтори'#7);
    until Code_=0;
    WriteLn('что дали');
    for i:=1 to v do
          for j:=1 to g do
                m[i][j] := random(1000);
    for i:=1 to v do
          begin
```

```
                for j:=1 to g do
                    Write(m[i][j]:5:3,'  ');
                WriteLn;
            end;
    for i:=1 to v-1 do
            for j:=v downto i do
                Swap(m[i],m[j],g);
            WriteLn('что получили');
    for i:=1 to v do
            begin
                for j:=1 to g do
                    Write(m[i][j]:8:3);
                Writeln;
            end;
end.
```

Задачи по теме «Сортировка и слияние массивов»

15.1 (4 б.) **Пузырьковая сортировка.** Упорядочить заданный массив $A(n)$ по неубыванию, многократно переставляя каждые два соседних элемента, нарушающие порядок. Процесс завершается по достижении упорядоченности массива.

15.2 (6 б.) **Сортировка выбором.** Упорядочить массив $P(m)$ по неубыванию, меняя местами каждый элемент с минимальным среди следующих за ним, то есть используя метод:

$$p_i' = \min_j \{p_j,\ j = i, i+1, \ldots, m\},\ i = 1, \ldots, m-1.$$

15.3 (6 б.) **Сортировка простыми вставками.** Упорядочить массив $R(l)$ по невозрастанию, используя следующий подход: для $i = 2, 3, \ldots, l$ каждый элемент r_i вставлять в нужное место среди упорядоченных ранее элементов $r_1, r_2, \ldots, r_{i-1}$, раздвигая их за счет удаления r_i.

15.4 (7 б.) **Сортировка бинарными вставками.** Как в предыдущей задаче, каждый из элементов r_i вставлять в нужное место ранее упорядоченной цепочки $r_1, r_2, \ldots, r_{i-1}$, используя для поиска места метод дихотомии (метод деления отрезка пополам).

15.5 (9 б.) **Сортировка слиянием.** Упорядочить по неубыванию массив $A(n)$, разбивая его на группы по 1, 2, 4, 8, … элементов и проводя попарное слияние соседних групп (число n не обязательно равно степени 2). Разрешается выделить дополнительный рабочий массив $B(n)$ и проводить слияние поочередно из A в B, затем — из B в A и так далее.

15.6 (8 б.) В неупорядоченном массиве $K(m)$ есть совпадающие элементы. Из каждой группы одинаковых элементов оставить только один, удалив остальные и поджав массив к его началу.

15.7 (6 б.) Турнирная таблица представлена квадратной матрицей $A(n, n)$, каждый элемент a_{ij} которой есть число голов, забитых i-й командой в ворота j-й команды. По диагонали расположить место каждой команды (по числу побед за вычетом числа поражений; в случае равенства — по суммарной разности забитых и пропущенных голов).

15.8 (7 б.) В турнирной таблице, полученной в предыдущей задаче, переставить команды (то есть соответствующие строки и столбцы таблицы) в соответствии с занятым местом, указанным на диагонали.

15.9 (8 б.) Упорядоченный по невозрастанию массив $B(n)$ преобразовать в упорядоченный по возрастанию, оставив по одному в каждой группе совпадающих элементов.

15.10 (6 б.) Даны два упорядоченных по возрастанию массива $A(m)$ и $B(n)$. Получить из них путем слияния упорядоченный по возрастанию массив C; совпадающие элементы вставлять единожды. Подсчитать количество элементов в массиве C.

15.11 (7 б.) Из двух упорядоченных по невозрастанию массивов $A(m)$ и $B(n)$ получить путем слияния упорядоченный по убыванию массив C; удаляемые элементы собрать в массиве D. Подсчитать количество элементов в массивах C и D.

15.12 (7 б.) Произвести слияние упорядоченного по возрастанию $A(m)$ и неупорядоченного $B(n)$ массивов ($n << m$) в упорядоченный по неубыванию массив C.

15.13 (7 б.) Путем слияния из возрастающего $A(m)$ и невозрастающего $B(n)$ массивов получить возрастающий массив C (с удалением совпадающих элементов). Подсчитать количество элементов в массиве C.

15.14 (8 б.) Упорядочить строки матрицы $A(m, n)$ лексикографически по убыванию.

15.15 (7 б.) Упорядочить строки матрицы $B(m, n)$ по неубыванию элементов ее k-го столбца.

15.16 (8 б.) Упорядочить строки матрицы $D(m, n)$ лексикографически по неубыванию первых k элементов строки.

15.17 (8 б.) Провести построчное слияние двух матриц $A(m, n)$ и $B(k, n)$, строки которых лексикографически упорядочены по неубыванию первых l элементов каждой строки.

15.18 (7 б.) Произвести построчное слияние двух матриц $A(m, n)$ и $B(k, n)$, упорядоченных по неубыванию элементов первого столбца.

15.19 (7 б.) Матрица $K(m, n)$ упорядочена по возрастанию элементов первого столбца. С внешнего устройства (с клавиатуры, из файла и так далее) поступают дополнительные строки. Вставлять их в нужное место, раздвигая остальные, с сохранением упорядоченности. При невозможности такой вставки (первые элементы строк совпадают, а остальные — нет) вводимую строку вывести на печать с сообщением о невозможности вставки. Результат – пополненная матрица.

15.20 (8 б.) Упорядочить по неубыванию каждую строку матрицы $A(m, n)$, а после этого перестановкой строк упорядочить всю матрицу по неубыванию элементов первого столбца.

15.21 (7 б.) Возрастающий массив $A(n)$ хранится в памяти. С устройства ввода (из файла или с клавиатуры) поступают элементы неупорядоченного массива B. По мере ввода элементов массива B производить слияние этих массивов в массив A, не выделяя дополнительного места под массив B. При совпадении элементов (невозможности вставки без нарушения признака порядка) выводить их на печать с сообщением. Работу прекратить при исчерпании входных данных или при заполнении всей памяти, отведенной под массив A.

15.22 (8 б.) Строки матрицы $A(m, n)$ в произвольном порядке поступают с устройства ввода (из файла или с клавиатуры). Располагать их по мере поступления в выделенном под нее массиве лексикографически по возрастанию; совпадающие строки вставлять единожды.

15.23 (7 б.) Элементы массива $D(n)$ случайным образом перемешаны. Элементы k_i массива $K(n)$ указывают номера позиций, которые занимали соответствующие элементы d_i до перемешивания. Восстановить исходное состояние массива D.

15.24 (8 б.) Матрица $L(m, n)$ состоит из нулей и единиц. Удалить из нее совпадающие строки, а оставшиеся упорядочить по возрастанию двоичных чисел, образуемых строками. Число n больше разрядности компьютера.

15.25 (8 б.) В начале каждой строки частично заполненной матрицы $A(m, n)$ сгруппированы элементы, упорядоченные по возрастанию. В массиве $K(m)$ указано количество элементов в каждой строке. Слить все строки матрицы A в одномерный неубывающий массив B.

15.26 (9 б.) Массив $X(n)$ разбит на m фрагментов. В целочисленном массиве $K(m)$ хранятся длины соответствующих фрагментов (все k_i различны, их сумма равна n). Упорядочить массив K по возрастанию, переставив соответствующие фрагменты в массиве X.

15.27 (7 б.) Строки и столбцы верхнетреугольной матрицы, не содержащей нулей на диагонали и над ней, случайным образом перемешаны, первоначальные номера строк и столбцов не сохранены. Перестановкой строк и столбцов восстановить исходный вид матрицы.

15,28 (7 б.) В результате эксперимента получены наборы значений аргумента x и соответствующих значений функции y. Сформировать и напечатать таблицу значений функции, упорядочив их по возрастанию x. Если одному значению x соответствует несколько значений y, взять их среднее значение.

15.29 (8 б.) В результате эксперимента получено n пар значений величин x и y в массивах $X(n)$ и $Y(n)$. Массив X не упорядочен, но не содержит одинаковых элементов. Построить таблицу значений функции $y(x)$ с постоянным шагом $h=(x_{\max}-x_{\min})/(n-1)$, где $x_{\max}$ и $x_{\min}$ — соответственно наибольшее и наименьшее значения в массиве X. Значения y вычислять по формуле линейной интерполяции соседних наблюдений (после упорядочения по x): $y = y_{i-1} + \dfrac{x - x_{i-1}}{h}(y_i - y_{i-1})$.

15.30 (10 б.) В результате m экспериментов разной продолжительности получены наборы значений функций $y^{(k)} = y^{(k)}(x)$, $k = 1, 2, \ldots, m$ с соответствующими значениями аргумента – массивы $X^{(1)}(n_1)$, $Y^{(1)}(n_1)$; $X^{(2)}(n_2)$; $Y^{(2)}(n_2)$; ...; $X^{(m)}(n_m)$, $Y^{(m)}(n_m)$. Многие значения x в разных экспериментах совпадают. Получить совокупную таблицу значений всех функций, слив упорядоченные массивы $X^{(1)}, X^{(2)}, \ldots, X^{(m)}$. Отсутствующие наблюдения функций $y^{(k)}$ печатать пробелами или другими символами.

15.31 (7 б.) Перенести в начало массива $A(n)$ все его отрицательные элементы, затем — нулевые и в конце — все положительные (с сохранением порядка следования в каждой группе).

15.32 (9 б.) К подготовленной статье автор приложил список использованной литературы, но расположил издания в порядке появления ссылок на них в тексте. Редактор потребовал расположить источники по алфавиту. Упорядочить список литературы по требованию редактора (каждое название – с нового абзаца: номер п/п, автор(-ы), название работы и т. д.

(12 б.) **Развитие задачи.** Текст статьи со ссылками прилагается; после сортировки внести соответствующие исправления в текст. Для справки: каждая ссылка — это (в квадратных скобках) номер из списка или несколько номеров. Проверить корректность ссылок (наличие в списке источника с соответствующим номером).

15.33 (8 б.) Из неупорядоченного числового массива $A(m)$ получить упорядоченный по возрастанию массив $D(n)$ $(n \le m)$ следующим образом. Выбрать в A наименьший элемент и поместить его в D, затем — наименьший из оставшихся и т. д. Повторяющиеся элементы включать единожды; массив A сохранить.

15.34 (10 б.) Переставить элементы матрицы $A(n,n)$ так, чтобы на главной диагонали и линиях, параллельных ей, стояли неубывающие последовательности (перестановку элементов вести только в пределах этих линий).

15.35 (10 б.) Массив чисел, заданных в шестнадцатеричной системе счисления в текстовом файле (по одному числу в строке, цифры разделены пробелами), упорядочить по неубыванию, не переводя числа в другую систему.

15.36 (8 б.) **Богатый урожай.** Собранную пшеницу фермер затарил в n мешков, пронумеровав их и измерив засоренность в каждом мешке, он получил массив значений засоренности $S(n)$. Ему необходимо k мешков оставить на семена со средней засоренностью не более z_1, как можно больше выставить на продажу со средней засоренностью не более z_2, остальные мешки отправить на корм скоту. Как ему поступить?

15.37 (9 б.) Упорядочить строки матрицы $K(m,n)$, содержащей натуральные числа, по возрастанию суммы цифр в десятичной системе счисления, используемых для записи элементов строки.

15.38 (9 б.) **Горе-летописец.** Первое слово каждой строки текстового файла (первое поле каждой записи в массиве записей) содержит дату события в формате Д/М/ГГГГ, где месяц представлен римскими цифрами, например 13/IV/2001; остальные слова или поля — описание события. Упорядочить события хронологически (по возрастанию дат).

15.39 (8 б.) Несколько арифметических прогрессий заданы своими параметрами: $a^{(i)}$ — первый член; $d^{(i)}$ — разность; $n^{(i)}$ — количество членов; $d^{(i)} > 0$, $i = 1, ..., m$. Не находя самих прогрессий (в виде массивов или файлов), произвести их слияние в один неубывающий массив.

15.40 (10 б.) **Клуб однофамильцев.** В массиве (файле) хранятся данные о членах некоторого коллектива: фамилия, имя (как мужского, так и женского пола), телефон. Среди представленных персон немало однофамильцев. Упорядочить список по невозрастанию количества однофамильцев. Наверняка на первом месте окажутся все Ивановы, на одном из последних — Ленин.

15.41 (9 б.) Переставить строки и столбцы квадратной матрицы $A(n,n)$ так, чтобы элементы главной диагонали образовали неубывающую последовательность.

15.42 (10 б.) **Задача о двух станках.** Имеется n деталей, каждая из которых проходит обработку сначала на одном станке, затем на другом (например, токарный и шлифовальный). На каждом станке одновременно обрабатывается только одна деталь; время на переналадку не требуется. Известно время обработки детали на каждом станке. Упорядочить детали так, чтобы суммарное время обработки партии деталей было минимально.

В задачах 15.43–15.46 предусмотрена работа со списками.

15.43 (7 б.) Произвести слияние двух заданных упорядоченных по возрастанию списков в один неубывающий список.

15.44 (7 б.) Провести сортировку элементов заданного списка по неубыванию.

15.45 (9 б.) В заданном неупорядоченном списке оставить по одному в каждой группе совпадающих элементов, сохранив порядок следования остальных.

15.46 (8 б.) Пополнить упорядоченный по возрастанию список A элементами неупорядоченного списка B, сохранив упорядоченность (совпадающие элементы включать единожды).

16. Поиск с возвратом. Задачи на графах

Задачи этой темы носят в основном олимпиадный характер и не являются обязательными, тем не менее могут существенно пополнить «копилку» накопленных баллов по другим задачам. Приведем лишь минимально необходимые теоретические сведения и понятия, а за подробностями алгоритмизации отошлем к специальной литературе и консультантам.

Граф, содержащий n вершин u_1, u_2, ..., u_n, задается множеством ребер — неупорядоченных пар вершин $<u_i, u_j>, i \neq j$.

Путь в графе — последовательность вершин, попарно связанных ребрами. Граф называется связным, если между любыми двумя его вершинами существует путь (в противном случае граф разбивается на ряд связных подграфов).

Орграф, или *ориентированный граф*, задается множеством дуг — упорядоченных пар вершин $<u_i, u_j>$. *Петля в орграфе* — это дуга вида $<u_i, u_i>$. *Путь в орграфе* — последовательность вершин, попарно связанных дугами в одном направлении.

Цикл — это замкнутый путь, в котором последняя вершина совпадает с первой.

Дерево — связный граф или орграф без циклов.

Любой граф может быть задан *матрицей смежности* $G(n, n)$. Так, для орграфа

$$g_{ij} = \begin{cases} 1, \text{ если есть дуга } < u_i, u_j >; \\ -1, \text{ если есть дуга } < u_j, u_i >; \\ 0 \text{ в противном случае.} \end{cases}$$

Для неориентированного графа все g_{ij} неотрицательны, матрица смежности симметрична.

Вместо этого граф можно задать множеством ребер или дуг $<u_i, u_j>$ – пар $<i, j>$ номеров их вершин.

Дугам или ребрам могут быть приписаны *численные веса* (например, длина дуги или ребра, пропускная способность, продолжительность работы и т. д.). Эти веса записываются в матрицу смежности (со знаком для орграфа) вместо ее ненулевых элементов.

Для неориентированного графа матрица смежности симметрична и может быть задана в верхнетреугольном виде.

Пример. Рассмотрим задачу 16.22 – склеивание треугольников в графе. Одно из возможных решений ее имеет следующий вид:

```
program Graf_16_22;
     uses crt;
     const max=200;
     type gr=Array [1..max,1..2] of Word;
     var i,j,k,n: Word;
        u:gr;
        p:Array[1..max] of Word;
        a:Boolean;
     procedure Repet(var s:gr;l:Word);
        var i,w,e,m:Word;
           c:Boolean;
           g:gr;
     begin
        m:=1;
        l:=0;
        for i:=1 to n-1 do
           begin
               for w:=1 to m do
                  begin
                     if s[w,1]>s[w,2] then
                        begin
                           e:=s[w,1];
                           s[w,1]:=s[w,2];
                           s[w,2]:=e;
                        end;
                     if (s[w,1]=i) and (i<>s[w,2])
                        then
                        begin
                           c:=true;
```

```
                    for e:=p[i-1]+1 to l do
                    if g[e,2]=s[w,2] then
                       c:=false;
                    if c then
                          begin
                             l:=l+1;
                             g[l,1]:=i;
                             g[l,2]:=s[w,2];
                          end;
           end;
     end;
     p[i]:=l;
     WriteLn(p[i]);
   end;
   for e:=1 to l do
   writeln('  ',g[e,1],' - ',g[e,2]);
   for e:=i+1 to max do
     begin
        g[e,1]:=0;
        g[e,2]:=0;
     end;
   s:=g;
   k:=l;
end;
procedure look(var d:gr);
   var s,z,i,j,t:word;
     c:Boolean;
     g:gr;
begin
   t:=0;
   c:=true;
   while (t<n) and (c) do
     begin
        t:=t+1;
        if t=1 then s:=0 else s:=p[t-1];
        while (s<p[t]) and (c) do
           begin
              s:=s+1;
              if d[t,2]<d[s,2] then
                 begin
                    i:=d[t,2];
                    j:=d[s,2];
                 end
              else
                 begin
                    i:=d[s,2];
                    j:=d[t,2];
```

```
                end;
          for z:=p[i-1]+1 to p[i] do
          if d[z,2]=j then c:=false;
      end;
    end;
    if not (c) then
      begin
        for z:=1 to k do
          for s:=1 to 2 do
            if (d[z,s]=i) or (d[z,s]=j)
              then d[z,s]:=t
            else if d[z,s]>j then
              d[z,s]:=d[z,s]-2
              else if d[z,s]>i then
                d[z,s]:=d[z,s]-1;
        n:=n-2;
        WriteLn
          ('Склеиваем треугольник',
            t,'-',i,'-',j,' в ',t);
        ReadLn;
      end
    else a:=true;
  end;
begin
  ClrScr;
  a:=false;
  WriteLn
    ('Введите количество вершин графа');
  ReadLn(n);
  WriteLn
    ('Вводите ребра. окончание - 0-0');
  k:=0;
  repeat
    Write('>');
    Read(i);
    Read(j);
    if (i>0) and (j>0) and (i<=n)
       and (j<=n) then
      begin
        k:=k+1;
        u[k,1]:=i;
        u[k,2]:=j;
      end
    else if (i<>0) or (j<>0) then
      writeln('Вы что-то перепутали');
  until (i=0) and (j=0);
  repeat
```

```
      Repet(u,k);
      ReadLn;
      Look(u);
    until a;
    WriteLn('Это граф без треугольников');
    ReadLn;
  end.
```

Задачи по теме «Поиск с возвратом. Задачи на графах»

16.1 (15 б.) Граф, заданный матрицей смежности, проверить на связность; в случае несвязности вывести подмножества вершин каждого связного подграфа.

16.2 (15 б.) Заданный орграф проверить на наличие циклов и при их наличии вывести каждый цикл в виде последовательности вершин циклического пути.

16.3 (20 б.) В орграфе без циклов, заданном матрицей смежности с весами (длинами) дуг, найти *критический путь*, то есть путь наибольшей длины, идущий из первой вершины до последней.

16.4 (7 б.) В орграфе найти все *стоки*, то есть вершины, в которые только входят дуги, и *истоки*, из которых только выходят дуги.

16.5 (15 б.) В заданном неориентированном графе найти кратчайший путь, связывающий две заданные вершины.

16.6 (12 б.) Матрицу смежности несвязного графа *перенумерацией вершин* (перестановкой строк и столбцов) превратить в блочную, состоящую из нескольких подматриц, расположенных на главной диагонали. При этом каждая подматрица будет задавать связный подграф.

16.7 (10 б.) Известно, что плоская фигура может быть обведена за один прием, «не отрывая карандаша от бумаги», если она содержит две или ни одной точки, в которой сходится нечетное число линий. Фигура задана множеством пар номеров вершин $<i, j>$, соединенных линиями. Найти путь обхода фигуры или показать, что его не существует.

16.8 (15 б.) **Касса.** Массив $K(n)$ содержит значения (номиналы) денежных знаков (купюр и монет) некоторой валютной системы; $L(n)$ — количество знаков каждого достоинства в кассе. Массив $S(m)$ — ведомость выдачи зарплаты; известно, что $\sum_{i=1}^{m} s_i \le \sum_{j=1}^{n} k_j l_j$, то есть касса платежеспособна. Реализовать выдачу зарплаты, то есть найти количество знаков каждого достоинства для каждого работника или показать, что без сдачи это сделать невозможно.

16.9 (20 б.) **Задача о ранце.** Из n предметов, обладающих каждый весом v_i и стоимостью p_i, $i = 1, 2, \ldots, n$, выбрать такие, что при суммарном весе не более V их суммарная стоимость максимальна.

16.10 (15 б.) Пусть n красных и n синих точек на плоскости заданы своими координатами. Построить n отрезков с разноцветными концами, суммарная длина которых минимальна (каждая точка является концом только одного отрезка).

16.11 (12 б.) **Минимальное дерево-остов.** На плоскости своими координатами задано n точек. Построить связный граф с вершинами во всех этих точках так, чтобы суммарная длина его ребер была наименьшей.

Указание. Для решения задачи достаточно начиная с любой точки на каждом шаге присоединять к связной части графа ближайшую к ней несвязную точку.

16.12 (15 б.) В массиве $Z(n)$ найти наибольшую по количеству элементов арифметическую прогрессию (элементы прогрессии стоят в массиве в произвольном порядке).

16.13 (18 б.) **Задача о раскрое.** Из прямоугольника размером $a \times b$ требуется вырезать возможно большее число прямоугольников размером $c \times d$. Найти оптимальный вариант раскроя (возможно, таких вариантов несколько).

16.14 (15 б.) Из элементов массива $A(2n)$ сформировать два «почти ортогональных» массива $B(n)$ и $C(n)$, то есть таких, чтобы модуль скалярного произведения их был минимален.

16.15 (17 б.) Для каждого жителя города задано множество (возможно, пустое) имен его детей; каждый житель города имеет уникальное имя. Жители x и y называются родственниками, если либо x — ребенок y, либо y — ребенок x, либо существует некий z, такой, что x является родственником z, а z — родственником y. Получить все подмножества родственников. Вместо имен можно использовать шифры (номера) жителей.

16.16 (19 б.) В условиях предыдущей задачи найти жителя, имеющего наибольшее количество потомков (детей, внуков и так далее). В терминах теории графов это значит, что нужно в несвязном графе найти связный подграф с наибольшим числом вершин. Вывести также весь найденный родовой клан.

16.17 (15 б.) Найти все вершины графа, к которым существует путь заданной длины (не обязательно кратчайший) от его первой вершины.

16.18 (18 б.) **Крестики-нолики.** Клеточное поле размером $m \times n$ есть результат игры в крестики-нолики на «бесконечном» (неограниченном) поле. Проверить, является ли конфигурация предвыигрышной для одного из игроков, то есть нельзя ли за один ход достичь победы. Считается, что, например, «крестики» выиграли, если на поле найдется по горизонтали, вертикали или диагонали цепочка, состоящая из пяти крестиков.

16.19 (15 б.) В заданном орграфе перенумеровать вершины так, чтобы всякая дуга вела от вершины с меньшим номером к вершине с большим номером или показать, что это невозможно.

16.20 (15 б.) Определить, является ли заданный граф двудольным, то есть можно ли разбить множество его вершин так, чтобы каждое ребро соединяло вершины двух разных подмножеств.

16.21 (12 б.) *Треугольником* в графе называется всякая тройка различных и попарно смежных вершин этого графа. *Склеиванием* треугольника называется замена треугольника одной вершиной с сохранением связности его с остальным графом. Последовательно применяя склеивание, преобразовать данный граф в граф, в котором нет треугольников.

16.22 (20 б.) Имеется n костей домино (не обязательно из одного комплекта). Выстроить из них правильную последовательность максимальной длины.

16.23 (20 б.) Проверить, можно ли заданную прямоугольную целочисленную матрицу составить из костей одного комплекта домино.

16.24 (18 б.) Некоторые пары городов связаны прямыми дорогами, длина которых известна. Найти кратчайший путь между двумя заданными городами, или показать, что эти города не связаны дорогами.

16.25 (15 б.) **Электронная сваха.** Известен возраст каждого из n женихов и каждой из m невест. Сформировать из них возможно большее число пар так, чтобы разница в возрасте между молодоженами в каждой паре была не больше заданного k.

16.26 (18 б.) Латинским квадратом порядка n называется таблица размером $n \times n$, каждая строка и каждый столбец которой содержит числа 1, 2, ..., n. Проверить, не найдется ли среди всевозможных миноров заданной прямоугольной целочисленной матрицы латинского квадрата порядка более 1.

16.27 (18 б.) **Горячие клавиши.** Во многих программах в пунктах меню выбираются уникальные символы, которые связываются с «горячими» клавишами (например, New, Open, Save, Save as, Save all), так что для выбора нужного пункта достаточно нажать соответствующую клавишу (ввести соответствующий символ). Для заданного набора пунктов меню, представленных словами и словосочетаниями, выбрать неповторяющиеся горячие символы или показать, что это невозможно.

16.28 (16 б.) Дано множество из m n-мерных векторов. Удалить из него минимальное количество векторов так, чтобы среди оставшихся не было ортогональных (например, большинство векторов компланарны, а несколько векторов, коллинеарных между собой, ортогональны им).

16.29 (18 б.) Удалить из заданного числового массива наименьшее число элементов так, чтобы оставшиеся составили возрастающую последовательность (не меняя их порядок следования).

16.30 (20 б.) Среди множества целых чисел, хранящихся в массиве $K(n)$, найти подмножество, образующее самую длинную арифметическую прогрессию (в исходном массиве элементы прогрессии расположены беспорядочно).

Тесты. Сгенерировать достаточно длинную прогрессию, перемешать ее элементы случайным образом, добавить «вкрапления» случайных чисел; сформировать несколько прогрессий и перемешать их элементы случайным образом.

16.31 (20 б.) Для заданной матрицы $K(m, n)$ известно p_i — количество единиц в i-й строке и q_j — количество единиц в j-м столбце; все остальные элементы — нулевые. Найти один из способов расстановки нулей и единиц в матрице K или показать, что это невозможно.

Тестирование при отладке — на любой случайно сгенерированной матрице из нулей и единиц: подсчитать количество единиц по строкам и столбцам и передать результаты испытуемой программе.

32.32 (18 б.) **Черный квадрат.** В матрице $A(m, n)$, состоящей из нулей и единиц, найти самый большой квадрат (квадратную подматрицу), состоящий целиком из нулей.

32.33 (17 б.) Произведение матриц не коммутативно: $AB \neq BA$, но ассоциативно: $(AB)C = A(BC)$. Для вычисления произведения матриц $A_1 A_2 \ldots A_k$ известных размеров (m_1, n_1), (m_2, n_2), ..., (m_k, n_k) расставить скобки в произведении так, чтобы потребовалось минимальное количество умножений чисел — элементов матриц. Например, если $m_1 = 100$, $n_1 = 1$; $m_2 = 1$, $n_2 = 100$; $m_3 = 100$, $n_3 = 1$, вычисление $(A_1 A_2) A_3$ потребует 20 000 умножений, в то время как $A_1 (A_2 A_3)$ — всего 200 умножений.

17. Разработка простейших АРМ и ИПС

Задачи этого раздела состоят из двух этапов. Каждый из этапов рассматривается как самостоятельное задание и защищается отдельно. На первом предлагается «окунуться» в информационную и терминологическую среды потенциального пользователя, выяснить, какую должность этот пользователь может занимать (кому нужно это информационное обеспечение), выяснить, какая информация и для чего используется. После этого — преобразовать недостаточно структурированные документы, содержащие основную информацию, в таблицы типа .dbf или .mdb, спроектировать и защитить структуру *базы данных* в виде связанных таблиц (.dbf-файлов). Полезно (если это возможно) непосредственно познакомиться с потенциальными пользователями информационного обеспечения, с терминологической и информационной средой, с информационными потоками и документооборотом в предложенной отрасли или области знаний, выяснить, какие запросы приходится отрабатывать в профессиональной деятельности пользователя. Кроме этого предлагается продемонстрировать умение выполнять и обучить потенциального пользователя выполнению простейших оперативных запросов (из командного окна, без создания командных файлов и услуг профессиональных программистов) по спроектированной и частично заполненной (20–30 записей) базе данных.

Здесь же — на первом этапе — должно созреть «ядро» алгоритма будущего программного обеспечения — автоматизиро-

6 Зак. № 767

ванного рабочего места (АРМ) специалиста или информационно-поисковой системы (ИПС), то есть выполняется постановка задачи. Предлагаемая формулировка задач представляет собой лишь поверхностные наброски постановки и не претендует на полноту и завершенность; конкретизация и уточнение возлагается на разработчика. При этом может оказаться, что *функциональная полнота* (обеспечение всех потребностей пользователя) не может быть достигнута за ограниченное время, выделенное на разработку. В этом случае необходимо согласовать с преподавателем подмножество функций, составляющее ядро реальной системы.

Второй этап — непосредственная разработка программного обеспечения, в результате которого должен получиться законченный программный продукт, пригодный для опытной эксплуатации пользователем без участия разработчика, пусть и не обладающий функциональной полнотой. Полезно провести его апробацию силами потенциального пользователя или независимого эксперта, а в идеале — найти заказчика и заключить договорные отношения не на модельную, а на практическую разработку с последующим внедрением.

Требования к программам. Независимо от конкретности проблемы, можно выделить некоторые формальные требования, на которых защищается квалификация разработчика.

1. **Устойчивость программы.** Программа не должна терять работоспособности ни при каких, даже некорректных, действиях пользователя. Всякие действия, грозящие потерей информации, выполняются только после повторного подтверждения. Вводимая информация там, где возможно, подвергается логическому контролю.

2. **Обеспечение целостности баз данных.** При любых действиях пользователя базы не должны терять целостности (некорректность индексов, потеря ссылок в связях после удаления-добавления записей и т. д.).

3. **Функциональная полнота.** В рамках согласованного с преподавателем или заказчиком подмножества функций все они должны быть реализованы.

4. **Терминологическая среда и интерфейс.** В диалоговых средствах используются только термины, понятные пользователю, и не используются термины разработчика («запись», «индексация» и т. д.). Появление служебных англоязычных сообщений СУБД недопустимо. Язык диалога — с соблюдением норм вежливости, цветовая гамма — по общепринятым рекомендациям.

5. **Использование клавиатуры.** На любом этапе нажатие любой клавиши (особенно функциональных) должно игнорироваться или вызывать предусмотренные действия (описанные в средствах помощи). Привязка действий к клавишам должна быть общепринятой: F1 — помощь; Enter — согласие, завершение ввода; Esc — отказ, возврат к предыдущему узлу ветви алгоритма (с восстановлением экранной формы); Tab — переход к следующему полю, окну и т. д.; Shift-Tab — возврат к предыдущему полю и т. д.

6. **Порядок движения.** Движение по дереву алгоритма «сверху вниз» сопровождается заголовками всех пройденных вершин; возврат возможен только на предыдущий уровень с сохранением введенной информации, выбранных пунктов меню и указателей записей.

7. **Средства помощи и реклама.** При запуске программы появляется рекламная заставка, отражающая суть и возможности программного средства, а также сведения об авторе. В любой точке алгоритма в строке подсказки должны высвечиваться все активные в данный момент горячие клавиши; в любой момент при нажатии клавиши F1 должен выдаваться контекстно-зависимый (зависящий от ситуации) текст помощи.

8. **Входные и выходные документы.** Экранные формы для ввода и корректировки должны быть максимально «похожими» на привычные для пользователя документы; результаты работы не только отображаются на экране, но и выводятся в текстовый файл в привычной для пользователя форме с возможностью корректировки и распечатки.

9. **Средства документации.** Программы снабжаются внутренней документацией в виде комментариев, средств меню и диалога, средств помощи, а также (по требованию преподавателя или заказчика) внешней документацией в виде отчета, содержащего, как минимум, постановку задачи, структуру баз данных, инструкцию для пользователя с элементами рекламы, описанием алгоритма, экранных форм, примеры входных и выходных документов.

Кроме того, можно перечислить формальные требования к организации и обслуживанию баз данных, соблюдение которых защищает квалификацию разработчика. Программный продукт должен обеспечивать выполнение перечисленных ниже операций над базами.

1. База состоит из нескольких связанных таблиц с использованием (по необходимости) связей «один к одному», «один ко многим», «много к одному», «много ко многим».

2. Для облегчения просмотра и поиска используется упорядоченность с помощью индексации или (в крайнем случае) сортировки.

3. Вся информация поддается просмотру и редактированию. Удаляемую информацию полезно переносить в архивные базы для последующего просмотра и восстановления. Записи, содержащие много полей, можно просматривать как в табличном, так и в постраничном виде (в виде карточек).

4. Информация, поддающаяся стандартизации, хранится в отдельных таблицах-справочниках, также поддающихся корректировке (с соблюдением целостности).

5. Программа позволяет проводить поиск или выборку информации по произвольному запросу (фильтру). При неудачном задании условия запроса возможна его корректировка (с целью сужения или расширения зоны поиска).

Задачи по теме «Разработка простейших АРМ и ИПС»

17.1 **Личная библиотека.** Картотека домашней библиотеки: выходные данные книги (авторы, название, издательство и так далее), раздел библиотеки (специальная литература, хобби, домашнее хозяйство, беллетристика и так далее), происхождение и наличие книги в данный момент, субъективная оценка книги. Выбор книг по произвольному запросу; инвентаризация библиотеки.

17.2 **Картотека Интерпола.** Данные по каждому зарегистрированному преступнику: фамилия, имя, кличка, рост, цвет волос и глаз, особые приметы, гражданство, место и дата рождения, последнее место жительства, знание языков, преступная профессия, последнее дело и так далее. Преступные и мафиозные группировки (данные о подельщиках). Выборка по любому подмножеству признаков. Перенос «завязавших» в архив; удаление — только после смерти.

17.3 **Бюро знакомств.** База потенциальных женихов и невест: пол, регистрационный номер, дата регистрации, сведения о себе, требования к партнеру. Выбор подмножества подходящих кандидатур, подготовка встреч (формирование приглашения для знакомства). Перенос в архив пар, решивших свои семейные проблемы, удаление клиентов, отказавшихся от услуг.

17.4 **Биржа труда.** База безработных: анкетные данные, профессия, образование, место и должность последней работы, причина увольнения, семейное положение, жилищные условия, контактные координаты, требования к будущей работе. База вакансий: фирма, должность, условия труда и оплаты, жилищные условия, требования к специалисту. Поиск и регистрация вариантов с той и другой стороны; формирование объявлений для печати, удаление в архив после трудоустройства, полное удаление при отказе от услуг.

17.5 **Записная книжка.** Анкетные данные, адреса, телефоны, место работы или учебы, должность знакомых, коллег и родственников, характер знакомства, деловые качества и так далее. Автоматическое формирование поздравления с днем рождения (по текущей дате). Упорядочение по алфавиту и по дате последней корректировки. Поиск по произвольному шаблону.

17.6 **Касса аэрофлота.** Расписание: номер рейса, маршрут, пункты промежуточной посадки, время отправления, дни полета. Количество свободных мест на каждом рейсе. Выбор ближайшего рейса до заданного пункта (при наличии свободных мест), оформление заданного числа билетов по согласованию с пассажиром (с уменьшением числа свободных мест), оформление посадочной ведомости.

17.7 **Справочник потребителя (служба быта).** База предприятий бытового обслуживания города: название, разряд, адрес и телефоны, специализация, перечень оказываемых услуг, форма собственности, часы и дни работы. Поиск предприятий по заданной услуге и другим признакам.

17.8 **Справочник покупателя.** База торговых точек города: название, адрес и телефоны, специализация, форма собственности, время работы. Выбор магазинов по произвольному шаблону.

17.9 **Магазин с одним продавцом.** Компьютер вместо кассового аппарата. База наличия товаров: наименование, единица измерения, цена единицы, количество, дата последнего завоза. Регистрация поступления товара (как старых, так и новых наименований). Оформление покупки: выписка чека, корректировка базы. Проблема уценки и списания. Инвентаризация остатков товара с вычислением суммарной стоимости.

17.10 **Отдел кадров.** База данных о сотрудниках фирмы: паспортные данные, образование, специальность, подразделение, должность, оклад, даты поступления в фирму и последнего назначения и т. д. Выбор по произвольному шаблону. Сокращение штатов: выбор для увольнения лиц пенсионного и предпенсионного возраста, подготовка приказа.

17.11 **Генеалогическое дерево.** Паспортные данные членов некоторого родового клана; ссылки на детей (или на родителей). Поиск всех потомков или всех предков для указанного лица.

17.12 **Склад.** База товаров, хранящихся на складе: наименование, единица измерения, цена единицы, количество, дата последнего завоза. Регистрация поступления товара (формирование приходной накладной) и отгрузки (расходная накладная). Вывод инвентарной ведомости.

17.13 **Касса автовокзала.** Расписание автобусов: номер рейса, конечный и промежуточный пункты, время отправления. Количество свободных мест на каждом рейсе. Выбор ближайшего рейса до заданного пункта (при наличии свободных мест), оформление билетов, оформление посадочной ведомости. Предварительная продажа, возврат билетов.

17.14 **Администратор гостиницы.** Список номеров: класс, число мест. Список гостей: паспортные данные, даты приезда и отъезда, номер. Поселение гостей: выбор подходящего номера (при наличии свободных мест), регистрация, оформление квитанции. Отъезд: выбор всех постояльцев, отъезжающих сегодня, освобождение места или оформление задержки с выпиской дополнительной квитанции. Возможность досрочного отъезда с перерасчетом. Поиск гостя по произвольному признаку.

17.15 **Справочник меломана.** База групп и исполнителей; база песен; база дисков с перечнем песен (в виде ссылок). Выбор всех песен заданной группы; всех дисков, где встречается заданная песня.

17.16 **Ежедневник.** База намечаемых мероприятий — дата, время и протяженность, место проведения. Автоматическое напоминание ближайшего дела: по текущей дате и времени; удаление вчерашних дел либо перенос на будущее. Анализ «накладок» — пересечений планируемых дел. Просмотр дел на завтра, послезавтра и так далее.

17.17 **Терминология.** База определений какой-либо науки: вводимый термин, его толкование (определение), ссылки на используемые термины. Возможность просмотра всей цепочки от заданного термина до первичных понятий.

17.18 **Шеф-повар.** База рецептур блюд: раскладка, рецепт приготовления. База продуктов на складе: наименование, цена, количество. Формирование меню на день (на заданное число персон); званый ужин. Проверка достаточности запасов; формирование расходной накладной на склад, корректировка запасов.

17.19 **Справочник лекаря.** База болезней: название, симптомы, процедуры, перечень рекомендуемых лекарств с указанием требуемого количества. База медикаментов на складе: название, количество, взаимозаменяемость. Формирование рецепта после осмотра больного, проверка наличия лекарств, корректировка запасов.

17.20 **Зачисление абитуриентов.** База абитуриентов: анкетные данные, совокупность оценок на вступительных экзаменах, готовность учиться на договорной основе. Выбор для зачисления заданного количества абитуриентов; формирование для собеседования списка тех, кто набрал предельный проходной балл, но не может платить за образование.

17.21 **Обмен жилья.** База предложений по обмену: район, площадь, планировка и т. д.; требования к вариантам обмена. Регистрация клиентов, выбор подходящих вариантов, удаление при состоявшемся обмене или отказе.

Развитие задачи. Возможность съезда или разъезда, в том числе «несколько на несколько»; «возможны варианты».

17.22 **Справочник абитуриента.** База вузов: наименование, адрес, перечень специальностей, конкурс прошлого года по каждой специальности (дневной, вечерней, заочной форм), размер оплаты при договорном обучении. Выбор по разным критериям: все о данном вузе; все о данной специальности, поиск минимального конкурса по данной специальности или вообще.

17.23 **Справочник почтовой индексации.** Республика, область (край), район, населенный пункт, почтовый индекс. Поиск по любой совокупности полей (кроме последнего); иерархическая связь между полями (обратите внимание, что, например, Павловск есть в Алтайском крае, Воронежской и Ленинградской областях).

17.24 **Сбербанк.** Сведения о вкладчиках банка: номер лицевого счета, категория вклада, паспортные данные, текущая сумма вклада, дата последней операции. Операции приема и выдачи любой суммы, автоматическое начисление процентов.

17.25 **Ломбард.** База хранимых товаров и недвижимости: анкетные данные клиента, наименование товара, оценочная стоимость; сумма, выданная под залог, дата сдачи, срок хранения. Операции приема товара, возврата, продажи по истечении срока хранения.

17.26 **Справочник селекционера.** Наименование сорта какой-либо культуры, автор, родительские сорта, урожайность, характеристики плодов, морозоустойчивость, устойчивость к вредителям и болезням, наличие в том или ином селекционном фонде. Выбор сортов, обладающих заданными свойствами.

17.27 **Справочник работника ГИБДД.** Марка, цвет, заводской и бортовой номера, дата выпуска, особенности конструкции и окраски, дата последнего техосмотра транспортного средства (автомобиля, мотоцикла, прицепа и т. д.), паспортные данные владельца. Выбор транспортных средств по произвольному шаблону. Формирование приглашений на техосмотр в соответствии со сроком.

17.28 **Справочник владельца видеотеки.** База видеофильмов: название, студия, жанр, год выпуска, режиссер, исполнители главных ролей, краткое содержание, субъективная оценка фильма. Факт наличия фильма в видеотеке. Оформление выдачи и возврата кассеты.

17.29 **Купи-продай.** База продавцов: наименование товара, объем партии при оптовой продаже, цена, условия продажи-отгрузки, форма оплаты, контактный адрес или телефон, примечание (например, «посредников прошу не беспокоиться»). База покупателей: наименование товара, объем покупки, приемлемая цена и форма оплаты, контактный адрес или телефон, примечание. Поиск и регистрация вариантов с той и другой стороны; формирование объявлений для печати, удаление в архив после купли-продажи (возможно, один из клиентов остается неудовлетворенным), полное удаление при отказе от услуг.

17.30 **Справочник фаната.** База спортсменов: анкетные и антропологические данные, гражданство, происхождение, вид спорта, клуб или команда, данные о личном рекорде или победах и так далее. Выбор по произвольному признаку. Поиск рекордсмена в заданном виде спорта.

17.31 **Справочник радиолюбителя.** Базы паспортных данных транзисторов, диодов, тиристоров и так далее: марка, характеристики, предельно допустимые условия эксплуатации, цена, учет взаимозаменяемости и т. д. Подборка по заданным требованиям.

17.32 **Справочник коммерческих банков.** Наименование, адрес, статус (форма собственности), условия хранения средств на лицевом счете (годовые проценты на различных видах вкладов). Выбор банка с наибольшим процентом для заданного типа вклада.

17.33 **Справочник начальника тюрьмы.** Анкетные данные заключенных, статья, срок, дата заключения под стражу, место в тюремной иерархии, камера, сведения о родственниках, особенности характера. Формирование статистических сводок о составе, выбор по произвольному признаку.

17.34 **Справочник командира.** Список подчиненных военнослужащих: анкетные данные, адрес родителей, гражданская профессия, образование, звание и дата его получения, должность, подразделение, форма службы (срочная, кадровая, контрактная и так далее), период службы (для срочнослужащих), особенности характера и отношение к службе. Формирование списков: заданного подразделения, офицерского состава, новобранцев и т. д.

17.35 **Риэлтерская контора (купля-продажа жилья).** База предложений: район и адрес, характеристика дома и квартиры, запрашиваемая стоимость, координаты заявителя. База спроса: требования покупателя к жилью (возможно несколько вариантов, допустимые диапазоны), финансовые возможности, координаты заявителя. Подбор вариантов для той и другой стороны, автоматизированный поиск взаимоприемлемых вариантов. Пример запроса покупателя: однокомнатная, до 200 тыс. р., Поток и Новосиликатный не предлагать.

17.36 **Очередь на жилье.** Список очередников на получение и улучшение жилья: дата поступления в фирму, дата подачи заявления, состав семьи, жилищные условия на дату подачи заявления, льготы на дополнительную площадь, внеочередное и первоочередное улучшение, пожелания на район и другие. Реализация распределения получаемого муниципального жилья: удовлетворение подходящих очередников, переселение следующих в освободившиеся квартиры. Добавление заявителей и удаление выбывших и удовлетворенных.

17.37 **Автосалон.** База новых и подержанных отечественных и иностранных автомобилей: марка, год выпуска, технические характеристики, особенности исполнения, техническое состояние, запрашиваемая цена. База покупателей:

контактные координаты, требования к марке, техническим характеристикам и техническому состоянию, финансовые возможности Автоматизация подбора вариантов для покупателя, формирование заявки для поставщиков и перегонщиков.

17.38 **Справочник туриста.** Турагенства и предлагаемые услуги: страна, город (или маршрут круиза), условия проживания и проезда, экскурсионное обслуживание, сервис принимающей стороны, стоимость путевки.

17.39 **«Купи-продай».** База объявлений (радио-, газетных, в бегущей строке) по всем рубрикам. Поиск для любого спроса или предложения, включая контекстный поиск в МЕМО-полях (например «репетиторство & математика»).

17.40 **Крылатые фразы.** Справочник пословиц, поговорок, афоризмов, каламбуров, других словесных курьезов. Классификация по авторам и источникам, поиск по темам и ключевым словам.

17.41 **Каталог запчастей автомобиля.** В автомобиле насчитывается несколько тысяч деталей; некоторые используются в разных марках. Таблицы: страна, фирма-изготовитель, марка автомобиля, агрегат, узел, деталь. Учет взаимозаменяемости. Пользователи: работники автосервиса, магазинов запчастей; поставщики-оптовики.

17.42 **Каталог радиодеталей (справочник радиомастера).** Модели бытовой аудио- и видеотехники; для каждой модели — каталог радиодеталей, использованных в ней (резисторы, конденсаторы, катушки индуктивности, чипы и т. д.). Многие детали используются в разных моделях; некоторые детали односторонне (например, вместо резистора 10К 0,125Вт можно поставить 10К 0,25Вт, но не наоборот) или двусторонне (транзисторы с близкими характеристиками) заменяемы. Учет наличия деталей на складе, поиск подходящих деталей для ремонта.

17.43 **Справочник нумизмата.** Монеты: страна, номинал, год выпуска, количество выпущенных монет, особенности. Коллекционеры: страна, имя, контактные координаты, наличие редких монет в коллекции. Собственная коллекция.

17.44 **Справочник филателиста.** Марки: страна, нарицательная стоимость, год выпуска, тираж, особенности. Филателисты: страна, имя, контактные координаты, наличие редких марок в коллекции. Собственная коллекция.

17.45 **Справочник любителя живописи.** Художники с анкетными данными и стилями. Картины со ссылкой на художников, датой создания, жанром. Коллекционеры и музеи: наличие оригиналов картин и копий. Аукционы и комиссионки: дата проведения, список выставленных шедевров и цены на них. Собственная коллекция.

17.46 **Телепрограмма.** Программа телепередач нескольких телекомпаний (на неделю по дням, часам). Разные жанры телепередач: новости, спорт, художественные фильмы, сериалы и т. д. Выбор совокупной программы по определенному запросу (вкусу). Программирование видеомагнитофона при временных «накладках» передач.

17.47 **Справочник астронома.** Для каждой из зарегистрированных звезд известны: название, созвездие, видимая звездная величина, расстояние, координаты на небосклоне. Поиск звезд заданного созвездия, самых ярких звезд, видимых звезд и созвездий в заданной точке земного шара в заданное время.

17.48 **Справочник географа.** Города (географические координаты, численность населения), регионы (области, провинции, штаты и т. д. — принадлежность стране, столица, численность населения), страны (площадь, численность населения, форма государственного правления, столица), материки. Поиск городов-«тезок», подсчет доли городского населения в странах и регионах, населенность материков и т. д.

17.49 **Справочник гидролога.** Реки мира: протяженность (от истока до впадения), куда впадает (в другую реку, море или озеро), годовой сток (км^3), площадь бассейна (без крупных притоков). Вычисление годового стока и площади бассейна для заданной крупной реки (вместе с ее притоками), моря или озера.

17.50 **Справочник астронома.** Видимые звезды: название, созвездие, видимая звездная величина, расстояние от Земли, координаты на небосклоне: прямое восхождение (ч, мин) и склонение (град., мин). Поиск звезд, входящих в данное созвездие, самой яркой звезды созвездия, видимых созвездий и звезд в заданной точке земного шара в заданное время.

18. Электронные таблицы

Задачи этого раздела предлагается решить средствами электронных таблиц (SuperCalc, Excel или других). Результаты рекомендуется получить не только на экране, но и в текстовом файле (в виде документа) и, если возможно, в виде твердой копии (распечатки), а там, где это имеет смысл, дать графическое представление в виде графика (диаграммы) и сравнить с точным решением, полученным аналитически.

18.1 **Турнирная таблица.** В клетки квадратной матрицы — турнирной таблицы — вносится счет очередного матча. На диагонали матрицы рассчитывается разность числа забитых и пропущенных голов.

Развитие задачи. Подсчитывать текущее место каждой команды в чемпионате или даже пересортировывать таблицу в соответствии с местами.

18.2 **Расписание автобусов.** Известны следующие параметры: список остановок некоторого городского или пригородного автобусного маршрута, время проезда автобусом каждого перегона между остановками и время на посадку. Задаются моменты отправления каждого рейса с конечной остановки. Рассчитать маршрутную таблицу для всех заданных рейсов (в прямом и обратном направлениях).

18.3 **Расписание электричек.** Известны список остановок некоторого направления движения электропоездов, время проезда каждого перегона между остановками и время на остановку. Задаются моменты отправления каждого поезда с конечной станции. Рассчитать таблицу суточного расписания движения электропоездов в прямом и обратном направлениях.

Развитие задачи. Протяженность маршрутов различна (конечная станция у каждого поезда своя).

18.4 **Жордановы исключения.** Реализовать вычисление обратной матрицы методом Гаусса (в любой модификации; удобнее всего использовать жордановы исключения) с разрешающим элементом по диагонали.

Развитие задачи. Выбор разрешающего элемента вручную или автоматически.

18.5 **Ранг матрицы.** Используя метод Гаусса (в любой модификации), определить ранг заданной прямоугольной матрицы.

18.6 **Стипендиальная ведомость.** Реализовать расчет стипендии на студенческую группу. Учесть дифференцированный характер стипендии, компенсацию на детей, плату за общежитие, профсоюзные взносы и другие доплаты и удержания.

18.7 **Сбербанк.** Реализовать расчет карточки лицевого счета «до востребования». Годовой банковский процент фиксирован; начисления производятся при каждой операции пропорционально времени между операциями (с использованием «сложных процентов»). Реализовать операции поступления и снятия произвольной суммы.

18.8 **Экзаменационная ведомость.** В сводную ведомость с заготовленными фамилиями студентов вносятся результаты сессии. По результатам определить размер дифференцированной стипендии каждого студента и итоговую сумму, количество оценок каждого вида по группе по каждому экзамену и по всем экзаменам. При пересдаче экзамена новая оценка вносится прямо в ведомость.

18.9 **Званый ужин.** Справочная информация содержит закладку продуктов на 1 порцию (рецептуру) для приготовления определенного набора блюд и стоимость единицы каждого продукта. Из них требуется сформировать меню званого ужина на заданное число персон с подсчетом требуемого количества продуктов и общей стоимости с учетом затрат на приготовление.

Развитие задачи. Ограниченные запасы некоторых продуктов; ограниченные финансовые возможности заказчика.

18.10 **План производства.** Справочная информация содержит расходные коэффициенты различных сырьевых компонентов на выпуск единицы продукции каждого наименования (например количество деревоплиты, древесины, шпона, лака на 1 тумбочку); стоимость единицы каждого вида сырья, трудоемкость, энергоемкость каждого изделия и его отпускная цена. Требуется рассчитать производственный план (например, месячный) с подсчетом затрат каждого вида сырья, общей трудоемкости, прибыли за каждый вид продукции и итого.

18.11 **День здоровья.** Известны данные о результатах лыжного забега: фамилии и инициалы участников, возраст, время старта, время финиша. По возрасту выделены 3 возрастных категории, заданные диапазонами. Найти чемпиона по каждой возрастной категории.

18.12 **Штанга.** Результаты чемпионата тяжеловесов (например в толчке) представлены следующими данными: фамилия и инициалы, команда, собственный вес, результат в каждой из трех попыток. Найти чемпиона в каждой весовой категории, выделяемой с шагом 5 кг.

18.13 **Метод Эйлера.** Найти приближенное решение обыкновенного дифференциального уравнения второго порядка $y'' = f(x, y, y')$ с заданными начальными условиями $y(a) = y_0$ и $y'(a) = p_0$ на отрезке $[a, b]$ методом Эйлера с постоянным шагом h. Значения функции $y(x)$ и ее производной $p(x) = y'$ в узловых точках вычисляются по формулам: $y_{i+1} = y_i + hp_i$; $p_{i+1} = p_i + hf(x_i, y_i, p_i)$, $i = 1, 2, \ldots$

18.14 **Метод Ньютона.** Составить таблицу значений функции $y(x)$, заданной неявно уравнением $F(x, y) = 0$. Уравнение $F(x, y) = 0$ при фиксированном x решать методом Ньютона (см. задачу 11.2).

18.15 **Метод Рунге–Кутта.** Найти приближенное решение обыкновенного дифференциального уравнения $y' = f(x, y)$, $y(a) = y_0$ методом Рунге–Кутта пятого порядка на отрезке $[a, b]$ с заданным постоянным шагом h. Значения функции $y(x)$ в узловых точках вычисляются по формуле:

$$y_{i+1} = y_i + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), \quad i = 0, 1, 2, \ldots,$$

где $k_1 = f(x_i, y_i)$; $k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_1\right)$;

$$k_3 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_2\right); \; k_4 = f\left(x_i + h, y \; + hk_3\right).$$

В задачах 18.16–18.20 конкретные числовые данные вместе с известным результатом можно найти в сборниках задач по численным методам, методам оптимизации и исследованию операций.

18.16 Решить систему нелинейных (алгебраических или трансцендентных) уравнений: $f_i(x) = a_i$, $i = 1, \ldots, n$, где $x = (x_1, x_2, \ldots, x_n)$.

18.17 Решить задачу линейного программирования:

$$\begin{cases} z = \sum_{j=1}^{n} c_j x_j \to \max_x; \\ \sum_{j=1}^{n} a_{ij} x_j \le b_i, \; i = 1, \ldots, m; \\ x_j \ge 0, \; j = 1, \ldots, n. \end{cases}$$

18.18 Решить задачу выпуклого программирования:

$$\begin{cases} z = f(x) \to \max\limits_{x}; \\ \varphi_i(x) \le 0,\ i = 1, \ldots, m. \end{cases}$$

Здесь $x = (x_1, x_2, \ldots, x_n)$ — векторный аргумент.

18.19 Решить транспортную задачу:

$$\begin{cases} z = \sum\limits_{i=1}^{m} \sum\limits_{j=1}^{n} c_{ij} x_{ij} \to \min\limits_{x}; \\ \sum\limits_{j=1}^{n} x_{ij} = a_i,\ i = 1, \ldots, m; \\ \sum\limits_{i=1}^{m} x_{ij} = b_j,\ j = 1, \ldots, n; \\ x_{ij} \ge 0,\ i = 1, \ldots, m,\ j = 1, \ldots, n. \end{cases}$$

18.20 Решить задачу о назначениях (в Булевых — двоичных — переменных):

$$\begin{cases} z = \sum\limits_{i=1}^{m} \sum\limits_{j=1}^{n} x_{ij} c_{ij} \to \min\limits_{x}; \\ \sum\limits_{i=1}^{m} x_{ij} = 1;\ j = 1, \ldots, n; \\ \sum\limits_{j=1}^{n} x_{ij} = 1;\ i = 1, \ldots, m; \\ x_{ij} \in \{0;1\},\ i = 1, \ldots, m,\ j = 1, \ldots, n. \end{cases}$$

Литература

1. *Абрамов С. А., Антипов И. Н.* Основы программирования на алголе. 2-е изд., перераб. — М.: Наука, 1982. 172 с.
2. *Абрамов С. А., Гнездилова Г. Г., Капустина Е. Н., Селюн М. И.* Задачи по программированию. — М.: Наука, 1988. 224 с.
3. *Абрамов С. А., Зима Е. В.* Начала информатики. — М.: Наука, 1989. 256 с.
4. *Брудно А. Л., Каплан Л. И.* Олимпиады по программированию для школьников/Под ред. *Б. И. Наумова*. — М.: Наука, 1985. 96 с.
5. *Бухтияров А. М., Фролов Г. Д.* Практикум по программированию на Фортране (ОС ЕС ЭВМ). — М.: Наука, 1979. 304 с.
6. *Бухтияров А. М., Фролов Г. Д.* Сборник задач по программированию на алгоритмических языках. — М.: Наука, 1974. 240 с.
7. *Ван Тассел Д.* Стиль, разработка, эффективность и испытание программ/Пер. с англ. — М.: Мир, 1981. 320 с.
8. *Гарднер М.* Математические досуги/Пер. с англ. — М.: Мир, 1972. 496 с.
9. *Гарднер М.* Математические новеллы/Пер. с англ. — М.: Мир, 1974. 456 с.
10. *Григас Г.* Начала программирования. — М.: Просвещение, 1987. 112 с.

11. *Гудман С., Хидетниеми С.* Введение в разработку и анализ алгоритмов/Пер. с англ. — М.: Мир, 1981. 368 с.

12. *Двайт Г. Б.* Таблицы интегралов и другие математические формулы. — М.: Наука, 1966.

13. *Дрейфус М., Ганглоер К.* Практика программирования на Фортране: Упражнения с комментариями. — М.: Мир, 1978. 224 с.

14. *Джонс Ж., Харроу К.* Решение задач в системе Турбо Паскаль. — М.: Финансы и статистика, 1991. 720 с.

15. *Джонстон Г.* Учитесь программировать. — М.: Финансы и статистика, 1989. 368 с.

16. *Дробушевич Г. А.* Сборник задач по программированию: Учеб. пособие. — Мн.: Высш. школа, 1983. 324 с.

17. *Дьяконов В. П.* Справочник по алгоритмам и программам на языке бейсик для персональных ЭВМ. — М.: Наука, 1987. 240 с.

18. *Золотова С. И.* Практикум по Access. — М.: Финансы и статистика, 2000. 144 с.

19. *Касьянов В. Н., Сабельфельд В. К.* Сборник упражнений по практикуму на ЭВМ: Учебное пособие для вузов. — М.: Наука, 1986. 272 с.

20. *Кнут Д.* Искусство программирования на ЭВМ. Т.1. Основные алгоритмы. — М.: Мир, 1976. 736 с.

21. *Кнут Д.* Искусство программирования на ЭВМ. Т. 3. Сортировка и поиск. — М.: Мир, 1978. 846 с.

22. *Котов Ю. В.* Как рисует машина. — М.: Наука, 1988. 224 с.

23. *Кушниренко А. Г., Лебедев Г. В.* Программирование для математиков. — М.: Наука, 1988. 384 с.

24. *Лепин-Дмитрюков Г. А., Овчаренко Е. К.* Сборник задач по программированию на языке ПЛ/1. — М.: Сов. радио, 1980. 304 с.

25. Основы информатики и вычислительной техники. Ч. 2/ Под ред. *А. П. Ершова, В. М. Монахова.* — М.: Просвещение, 1986. 143 с.

26. *Офицеров Д. В., Долгий А. Б., Старых В. А.* Программирование на персональных ЭВМ: Практикум: Учеб. пособие. – Мн.: Высш. шк., 1993. 256 с.

27. *Пильщиков В. Н.* Сборник упражнений по языку Паскаль. – М.: Наука, 1989. 160 с.

28. Сборник задач по базовой компьютерной подготовке: Учеб. пособ. *В. С. Зубов, И. Н. Котарова, О. Г. Архипов* и др.; Под редакцией *И. Н. Котаровой.* – М.: Издательство МЭИ, 1998. 178 с.

29. *Светозарова Г. И., Мельников А. А., Козловский А. В.* Практикум по программированию на языке Бейсик: Учеб. пособие для вузов. – М.: Наука, 1988. 368 с.

30. *Светозарова Г. И., Сигитов Е. В., Козловский А. В.* Практикум по программированию на алгоритмических языках. – М.: Наука, 1980. 320 с.

31. *Ставровский А. Б.* Турбо Паскаль 7.0: Учебник. – К.: Издательская группа BHV, 2000. 400 с.

32. *Трифонов Н. П.* Сборник упражнений по алголу. 2-е изд., перераб. и доп. – М.: Наука, 1978. 208 с.

33. *Уэзерелл Ч.* Этюды для программистов. – М.: Мир, 1982. 288 с.

34. *Шураков В. В., Морозов В. П.* Задачник по основам алгоритмизации, алгоязыкам и машинной обработке: Учебное пособие. – М.: Статистика, 1978. 198 с.

35. *Юркин А. Г.* Практикум по программированию: Учебное пособие для студентов естественнонаучных специальностей вузов. 2-е изд. переработанное и дополненное. Изд-во Алт. ун-та: Барнаул, 1999. 127 с.

А. Г. Юркин

ЗАДАЧНИК
ПО ПРОГРАММИРОВАНИЮ

ПРАКТИКУМ

Для студентов и школьников старших классов, изучающих программирование.

В этой книге собраны задачи различной степени сложности по всем основным разделам программирования.
В ней сделана попытка максимально охватить многообразие приемов алгоритмизации и программирования, дать возможность поупражняться в поиске эффективных методов решения.
Задачи, как правило, не требуют математических или других специальных знаний: для решения большинства из них достаточно школьного курса и лишь в некоторых используются элементы аналитической геометрии, векторной алгебры и математического анализа (необходимые для этого сведения приводятся в дополнениях к условию задачи либо в ссылках на литературу).

В книге представлены задачи по следующим темам:

- линейные алгоритмы
- разветвляющиеся алгоритмы
- циклические алгоритмы
- векторы и матрицы
- линейный поиск
- арифметика
- множества
- сжатие информации
- комбинаторика
- численный анализ
- обработка символьной информации
- машинная графика
- электронные таблицы

ISBN 5-318-00399-0

9 785318 003998

ПИТЕР®
WWW.PITER.COM