Загогулины

для начинающего программиста

Редакция 1.5 от 2014-07-21

Количество задач: 285

Сборник составил Деревенец О.В.

(http://oleg-derevenets.narod.ru)

Аннотация

Это сборник относительно простых задач для начинающих программистов. Служит дополнением к учебнику «Песни о Паскале»; сборник может применяться при изучении других языков (Си, Java, Python).

Содержание

О сборнике

Загогулина — это линия или предмет сложной, витиеватой формы, — таковы извилины опытного программиста. Опыт приходит лишь с практикой в ходе решения задач и работы над проектами нарастающей сложности. В Сети найдётся немало сайтов с олимпиадными задачами для программистов. К сожалению, большинство этих задач слишком трудны для новичка, а битьё головой о неприступную стену, как говорит опыт, охлаждает его энтузиазм. Сначала лучше потренироваться «на кошечках», нужен промежуточный этап, где решаются задачи простые или средней сложности, но найти такие задачи непросто.

Настоящий сборник преследует именно эту цель: дать новичку начальный опыт решения относительно несложных задачек. Обычно начинают с задач, решаемых по шаблону, так называемых «паттернов». Затем следует очередь несложных, но творческих задач. Особо нетерпеливые найдут здесь и несколько относительно сложных, но очень интересных задачек.

«Песни о Паскале»

Сборник задуман как дополнение к учебнику «Песни о Паскале», содержащем около сотни задач. Сотня — это кое-что, но не достаточно. Чтобы не раздувать учебник, я вынес дополнительные задачи в отдельный файл и сгруппировал их по плану книги. Таким образом, вы можете приступать к задачам, прочитав соответствующие главы «песен».

Вместе с тем, очевидно, что изучать Паскаль, да и другие языки, кто-то будет по иным источникам: книгам, лекциям, урокам. Хотелось, чтобы задачник был полезен и для них. Поэтому ряд задач из «песен» тоже включен в сборник, а каждая глава сборника открывается небольшим перечнем теоретических вопросов, которыми надо овладеть. Разумеется, что от главы к главе теория накапливается по нарастающей.

О процедурах и функциях

От школьников редко требуют создания собственных процедур и функций: их программы обычно «монолитны». Однако будущему программисту такой опыт крайне необходим, поэтому я рекомендую, а в некоторых задачах даже требую создавать в программе отдельные процедуры и функции.

Об ответах и решениях

Любой ученик мечтает заполучить решения задач, но у меня пока есть не все решения (они выложены отдельным файлом). Зато ко многим задачам даны контрольные примеры. А там, где их нет, предполагается самостоятельная проверка, — программист должен уметь создавать свои тесты.

Об источниках и авторах

Я старался соблюдать то, что с некоторой натяжкой можно назвать авторским правом на задачи. С натяжкой, поскольку многие типовые задачи с небольшими вариациями кочуют из книги в книгу, с сайта на сайт. Их истинные авторы обычно не известны, как говорится, музыка и слова здесь народные. И всё же для многих задач я указал источники, откуда они мною почерпнуты. Следующая ниже таблица содержит источники и другие ресурсы с

задачами. Некоторые задачи придуманы мною, но я не покушаюсь на авторскую монополию. Свежую редакцию этого сборника ищите на моём сайте:

http://oleg-derevenets.narod.ru

Источники задач и другие полезные ресурсы

№	Источник	Примечание
1	http://ptaskbook.com/ru/tasks/index.php	Простые задачи
2	http://tpxexe.narod.ru/problems.html	Задачи с решениями
3	http://pascal-for-all.ucoz.ru/publ/	Задачи с решениями
4	http://www.intuit.ru/department/school/olympr/	Ларина Э.С.
5	http://ips.ifmo.ru/courses/course1/index.html	Учебный сайт ИТМО
6	http://algolist.ru/	Алгоритмы
7	http://www.lotos-khv.narod.ru/zadachi/index.html	Потопахин В.В.
8	Занимательное программирование	Мозговой М.
9	Олимпиадные задачи по программированию	Скиена
10	Решение сложных и олимпиадных задач по программированию	Долинский М.С.
11	Олимпиадные задачи по программированию	Меньшиков Ф.
12	Збірник задач підвищеної складності з інформатики	Беркунський Є.Ю.
13	Алгоритмы на TP	Шилд

1 Главы 1–17
Целые и булевы

Теоретическая база

- Целочисленный тип данных
- Ввод и вывод чисел
- Арифметические операции с числами, сравнение чисел
- Булев тип данных
- Ввод и вывод булевых данных
- Логические операции
- Условные операторы IF-THEN-ELSE и CASE-OF
- Цикл с проверкой в конце
- Генерация случайных чисел

1.1 Стоимость учебников

Источник: --

Стоимость одного учебника составляет **N** рублей и **M** копеек. Для класса приобретено **K** учебников. Пользователь вводит три упомянутых числа, а программа вычисляет стоимость покупки в рублях и копейках.

Контрольный пример:

Входные данные	Результат
N=3 M=17 K=25	79 руб. 25 коп.

1.2 Одинаковые цифры

Источник: 1

Вводится число, программа должна напечатать **TRUE** (или **FALSE** в противном случае), если это число двузначное и обе его цифры одинаковы.

Контрольные примеры:

Входные данные	Результат
77	TRUE
333	FALSE
5	FALSE

1.3 Чётность цифр

Источник: 1

Вводится число, программа должна напечатать **TRUE** (или **FALSE** в противном случае), если запись этого числа содержит только чётные цифры.

Контрольные примеры:

Входные данные	Результат
4206	TRUE
212	FALE

1.4 Количество цифр в числе

Источник: 1

Программа принимает два целых числа и печатает, сколько цифр содержит их сумма.

Контрольные примеры:

Входные данные	Результат
11 + 88	2
60 + 40	3
5 + 2	1

1.5 Размещение палаток (1)

Источник: 1

Площадка для кемпинга имеет размеры **N** x **M** метров, а палатка имеет форму квадрата со стороной **K** метров. Надо определить количество палаток, которые можно разместить на площадке.

Контрольный пример:

Входные данные	Результат
N=20 M=50 K=3	96

1.6 Размещение палаток (2)

Источник: 1

Площадка для кемпинга имеет размеры **N** x **M** метров, а палатка — **K** x **L** метров. Надо определить наибольшее количество палаток, которые можно разместить на площадке.

Контрольный пример:

Входные данные	Результат
N=20 M=50 K=3 L=4	80

1.7 Укладка коробок

Источник: 1

Упаковочный ящик имеет размеры **L** x **M** x **N** см, а коробка с товаром — размеры **P** x **Q** x **R** см. Определить наибольшее количество коробок, вмещаемое в ящик.

П о д с к а з к а . Тут возможны 6 вариантов упаковки.

Контрольный пример:

Входные данные	Результат
L=180 M=120 N=150 P=15 Q=24 R=35	250= (180 div 35)*(120 div 24)*(150 div 15)

1.8 Логические выражения для чисел

Источник: 1

Вводятся три целых положительных числа, программа печатает **TRUE** или **FALSE** для следующих булевых выражений:

- все числа совпадают;
- все числа разные;
- два из трёх чисел совпадают;
- все числа нечётные.

Контрольный пример:

Входные данные	Результат
3 3 5	FALSE
	FALSE
	TRUE
	TRUE

1.9 Три монеты

Источник: "Песни о Паскале" 13-Г

В переменные **M1**, **M2** и **M3** вводится итог подбрасывания трех монет так, чтобы **TRUE** соответствовал «орел», а **FALSE** – «решка». Составить пять выражений таких, чтобы они выдавали **TRUE** для следующих случаев:

- у всех монет выпал «орел»;
- у всех монет выпала «решка»;
- все монеты упали одинаково;
- у первой – «решка», у прочих – «орел»;
- у первой – «орел», а две остальные упали одинаково.

Подсказка: логические данные можно сравнивать; сравнение обладает самым низким приоритетом, и потому внутри выражений заключается в скобки, например: **M1 and (M2=M3)**.

1.10 Большее из трёх

Источник: "Песни о Паскале" 14-В

Пусть ваша программа запросит три числа: **A**, **B** и **C**, а затем напечатает большее из трех.

Подсказка: примените булевы выражения вкупе с операциями сравнения, которые в булевых выражениях надо заключать в скобки, например:

```
if (A>=B) and (A>=C) then . . .
```

1.11 Кирпич

Источник: "Песни о Паскале" 14-Г

В стене прорублено прямоугольное отверстие со сторонами **A** и **B**. Пусть ваша программа определит, пройдет ли в него кирпич с ребрами **X**, **Y**, **Z**.

1.12 Сравнение площадей

Источник: "Песни о Паскале" 14-Д

Площадь земельного участка вычисляется как произведение его сторон: ширины **A** и длины **B**. Введите в программу ширину и длину двух участков (**A1**, **B1** и **A2**, **B2**), и пусть программа напечатает ширину и длину большего из них.

1.13 Цвет клетки

Источник: 1

Шахматная доска имеет размер 8x8 клеток, причём клетка с координатами (1,1) — чёрная. Программа принимает два числа **N=1...8** и **M=1...8**, а затем печатает цвет клетки с координатами (N,M). Если принятые числа выходят за пределы 1...8, выводится сообщение об ошибке.

Контрольные примеры:

Входные данные	Результат
2, 2	чёрная
3, 6	белая

1.14 Преобразование числа

Источник: 1

Пользователь вводит однозначное или двузначное число **N**, а программа печатает число, образуемое перестановкой цифр его десятков единиц.

Контрольные примеры:

Входные данные	Результат
75	57
10	1
2	20

1.15 Обмен значений (1)

Источник: 12

Обменять значения числовых переменных **A** и **B**, воспользовавшись третьей переменной.

1.16 Обмен значений (2)

Источник: 12

Обменять значения числовых переменных **A** и **B**, не применяя третьей переменной.

1.17 Упорядочение чисел

Источник: 1

Пользователь вводит два числа, а программа печатает сначала меньшее, затем большее.

Контрольные примеры:

Входные данные	Результат
5 2	2 5
7 7	7 7
2 9	2 9

1.18 Случайные булевы данные

Источник: "Песни о Паскале" 15-Г

Найдите способ сформировать ряд случайных булевых значений (**False**, **True**), напечатайте 20 из них.

П о д с к а з к а : булевы значения можно получить, сравнивая два случайных числа.

1.19 Генерация случайных чисел

Источник: "Песни о Паскале" 15-Д

Сгенерируйте два случайных числа (в диапазоне от 1 до 10) так, чтобы они гарантировано не совпадали. Сделайте то же самое для трех чисел.

1.20 Среднее из трёх

Источник: 1

Пользователь вводит три числа, а программа печатает среднее из них.

Контрольные примеры:

Входные данные	Результат
10 5 40	10
5 3 3	3

1.21 Номер несовпадающего числа

Источник: 1

Пользователь вводит три числа, два из которых совпадают. Программа печатает то из них, что отличается от прочих.

Контрольные примеры:

Входные данные	Результат
3 5 3	5
7 4 4	7

1.22 Два ближайших числа

Источник: 1

Программа принимает три числа и печатает пару тех из них, что меньше отличаются друг от друга.

Контрольные примеры:

Входные данные	Результат
40 21 19	19 21
20 20 20	20 20

1.23 Стороны треугольника

Источник: --

Программа принимает три числа и печатает «Да», если они могут выражать длины сторон треугольника, и «Нет» в противном случае.

П о д с к а з к а . Длина любой стороны треугольника не больше суммы длин двух других сторон.

1.24 Проверка високосного года

Источник: 1

Високосным является год, который делится на 4, за исключением тех годов, которые делятся на 100 и не делятся на 400.

Программа принимает год и печатает «ДА», если год високосный, и «НЕТ» в противном случае.

Контрольные примеры:

Входные данные	Результат
2000	ДА
2100	НЕТ

1.25 Определение квартала по месяцам

Источник: --

Год делится на четыре квартала: январь-март — это первый квартал и т.д. Программа принимает целое число — номер месяца, — а затем печатает номер соответствующего квартала или слово «Ошибка».

Контрольные примеры:

Входные данные	Результат
5	2
13	Ошибка

1.26 Определение месяцев по кварталу

Источник: --

Год делится на четыре квартала: январь-март — это первый квартал и т.д. Программа принимает от пользователя целое число — номер квартала, — а затем печатает через тире два числа: номера соответствующих месяцев или слово «Ошибка».

Контрольные примеры:

Входные данные	Результат
2	4 - 6
5	Ошибка

1.27 Название дней недели

Источник: "Песни о Паскале" 16-Б

Напишите программу, которая бы запрашивала номер дня недели, и в ответ печатала бы название этого дня («понедельник», «вторник» и так далее).

1.28 Время года

Источник: "Песни о Паскале" 16-Г

Пользователь вводит число – номер месяца от 1 до 12, а программа должна сообщить соответствующее ему время года: зима, весна, лето, осень.

П о д с к а з к а : в одной ветви можно применить несколько меток.

1.29 Сообщение возраста

Источник: "Песни о Паскале" 16-В

Пусть пользователь введет число – свой возраст в годах. Ваша программа должна напечатать фразу: «Вам столько-то лет» с правильным окончанием, например: «Вам 20 *лет*», или «Вам 34 *года*», или «Вам 41 *год*». Подсказка: надо определить последнюю цифру года операцией **`MOD 10`**. Некоторые числа выпадают из общего правила, их надо проверить особо (например, 11, 12, 13, 14).

1.30 Названия карт

Источник: 1

Старшинство карт пронумеровано цифрами: 1 — шестёрка, 2 — семёрка, и т.д.., 9 — туз. Мастям карт даны следующие номера: 1 — пики, 2 — трефы, 3 — бубны, 4 — червы. Программа принимает двузначное число, где старшая цифра указывает масть, а младшая — старшинство. Надо напечатать словами либо название карты, например, «семёрка бубен», либо «ошибка», если число не соответствует никакой карте.

Контрольные примеры:

Входные данные	Результат
21	шестёрка трефы
30	Ошибка

1.31 Управление танком

Источник: "Песни о Паскале" 16-Д

Танк в компьютерной игрушке может двигаться в одном из четырех направлений, обозначим их числами: 1 – север, 2 – восток, 3 – юг, 4 – запад. Направление движения изменяется тремя командами: 1 – направо, 2 – налево, 3 – кругом. Пользователь вводит начальное направление движения, а затем ряд команд. Программа должна определять и печатать всякий раз новое направление. Выход из цикла – команда 0.

1.32 Шахматные фигуры

Источник: "Песни о Паскале" 16-Е

Исходное состояние шахматных фигур известно всякому (если вы исключение из правила, ознакомьтесь с основами шахмат). Пользователь в цикле вводит число, по которому программа печатает название фигуры, стоящей на соответствующей вертикали шахматной доски (от 1 до 8). Ноль служит для выхода из цикла, а на все прочие числа программа сообщает об ошибке.

1.33 Платный участок (1)

Источник: "Песни о Паскале" 17-И

Платный участок трассы протянулся с **P1** до **P2** километра (**P1**<**P2**). А пост ГАИ размещен на километре **M**. Расположен ли этот пост на платном участке трассы? Пусть ваша программа разберется с этим.

1.34 Платный участок (2)

Источник: "Песни о Паскале" 17-К

Дорожная служба запланировала ремонт трассы на участке с **R1** по **R2** (**R1**<**R2**). В сочетании с условием предыдущей задачи ваша программа должна определить:

- Будут ли ремонтировать весь платный участок **P1-P2** ?
- Будут ли ремонтировать хотя бы часть платного участка **P1-P2** ? Если да, то определить длину ремонтируемой платной части.
- Будут ли ремонтировать хотя бы часть бесплатного участка? Если да, то определить длину ремонтируемой бесплатной части.

1.35 Наибольший общий делитель (НОД)

Источник: --

Пусть даны два целых числа **N** и **M**, надо найти их наибольший общий делитель.

П о д с к а з к а . Пока числа не равны, вычитайте из большего числа меньшее.

Контрольные примеры:

Входные данные	Результат
12, 18	6
45, 75	15

1.36 Викторина (1)

Источник: --

В телевизионной викторине десять этапов. В каждом следующем этапе участник выигрывает вдвое больше, чем в предыдущем, а общий выигрыш составляет сумму выигрышей на отдельных этапах. Пусть даны два целых числа: **N** — выигрыш на первом этапе и **M** — количество преодолённых этапов (от 1 до 10). Программа должна напечатать общий выигрыш участника.

Контрольные примеры:

Входные данные	Результат
1, 3	7
5, 5	155

1.37 Викторина (2)

Источник: --

В телевизионной викторине семь этапов. В первом этапе выигрыш составляет 1 рубль, а в каждом следующем этапе в **N** раз больше, чем в предыдущем. Общий выигрыш составляет сумму выигрышей на отдельных этапах. Пусть даны два целых числа: **N** — умножающий коэффициент (от 2 до 5) и **M** — количество преодолённых этапов (от 1 до 7). Программа должна напечатать общий выигрыш участника.

Контрольные примеры:

Входные данные	Результат
4, 3	21

1.38 Числа Фибоначчи

Источник: --

Числа Фибоначчи — это последовательность, где первые два числа равны единице, а каждое следующее равно сумме двух предыдущих:

1, 1, 2, 3, 5, 8, 13, 21, 34, 55 . . .

Программа должна запросить порядковый номер числа в этом ряду (N) и вывести N-е число Фибоначчи (N = 1 ... 20).

Контрольные примеры:

Входные данные	Результат
1	1
2	1
7	13
10	55

1.39 Проверка на Фибоначчи

Источник: 1

Дано число, программа должна напечатать «ДА», если оно принадлежит ряду Фибоначчи, и «НЕТ» в противном случае.

Контрольные примеры:

Входные данные	Результат
14	НЕТ
21	ДА

1.40 Проверка числа на простоту

Источник: 4

Ввести целое положительное число и напечатать «ДА», если это простое число и «НЕТ» в противном случае. Простым называется число, делящееся без остатка лишь на 1 и само на себя.

Контрольные примеры:

Входные данные	Результат
15	НЕТ
17	ДА
2	ДА
1	ДА

1.41 Печать простых чисел

Источник: 4

Ввести два целых положительных числа **A** и **B** и напечатать все простые числа, лежащие на этом интервале (включая крайние). Сделать два варианта программы:

1) Когда известно, что **A** < **B**

2) Когда соотношение между **A** и **B** неизвестно.

Контрольные примеры:

Входные данные	Результат
10 20	11 13 17 19
13 5	5 7 11 13
14 16	Пусто

2 Главы 18–28 Символы и строки

Теоретическая база

- Символьный тип данных
- Ввод и вывод символов
- Сравнение символов, кодировка символов
- Строковый тип данных
- Ввод и вывод строк
- Сравнение строк
- Доступ к символам строки
- Цикл со счётчиком
- Организация процедур и функций
- Ввод данных из текстовых файлов
- Вывод данных в текстовые файлы
- Цикл с проверкой в начале

2.1 Распечатка цифр

Источник: --

Пользователь вводит строку, а программа печатает только содержащиеся в ней цифры.

2.2 Подсчёт латинских букв

Источник: 1

Пользователь вводит строку, а программа печатает количество содержащихся в ней прописных и строчных латинских букв.

2.3 Преобразование букв

Источник: 1

Преобразовать во введённой строке все строчные латинские буквы в прописные.

2.4 Смешивание строк

Источник: --

Пользователь вводит две строки, а программа объединяет их, чередуя символы, например:

Входные данные	Результат
qwerty ABCD	qAwBeCrDty

2.5 Разделение символов

Источник: --

Дана строка символов, состоящая из латинских букв и цифр. Составить другую строку, где те же символы следуют так: сначала латинские прописные, затем цифры, а потом латинские строчные (внутри групп порядок прежний).

2.6 Разбор строки на отдельные слова

Источник: 4

Дана строка, состоящая из одного или нескольких слов, разделённых пробелами. Напечатать каждое слово в отдельной строке.

Контрольные примеры:

Входные данные	Результат
Один в один	Один в один
Два	Два

2.7 Удаление избыточных пробелов

Источник: 1

Дана строка из нескольких слов, разделённых одним или несколькими пробелами. Напечатать эту строку без избыточных пробелов.

2.8 Телефонный номер

Источник: "Песни о Паскале" 18-Г

Записи телефонных номеров обычно содержат дополнительные символы (скобки, черточки, пробелы), - для лучшего восприятия. Например: 8 (123) 45-67-89. Предположим, что пользователь их так и вводит. Пусть ваша программа преобразит строку с номером, удалив из нее все символы, кроме цифр. Например, после ввода указанного выше номера она должна напечатать: 8123456789.

2.9 Печать вразрядку

Источник: "Песни о Паскале" 18-Д

Пусть ваша программа напечатает введенную пользователем строку вразрядку, добавляя пробел после каждого символа, например: **'Pascal' → 'P a s c a l'**.

2.10 Печать строки

Источник: "Песни о Паскале" 19-Б

Напишите и испытайте процедуру (назовем её **Line** – «линия»), печатающую строку заданной длины, составленную из звездочек, например.

```
Line(3);    { печатает «***» }
Line(7);    { печатает «*******» }
```

П о д с к а з к а : внутри процедуры надо организовать цикл.

2.11 Очистка экрана

Источник: "Песни о Паскале" 19-В

Напишите процедуру для очистки экрана.

П о д с к а з к а : можно напечатать несколько десятков пустых строк (не менее 25, что зависит от настройки консольного окна).

2.12 Форматирование телефонного номера

Источник: "Песни о Паскале" 19-Д

Пользователь вводит строку с телефонным номером (только цифры), количество цифр заранее неизвестно. Ваша программа должна дополнить номер дефисами, разбивающими его на триады, т.е. по три цифры двумя способами:

- начиная с первых цифр, например: 112-345-1;
- начиная с последних цифр, например: 1-123-451.

2.13 Почтальон

Источник: "Песни о Паскале" 19-Е

Почтальон разносит газеты по улице, состоящей из **N** домов. Четные и нечетные номера расположены по разные стороны улицы. В здравом уме почтальон не рискует лишний раз переходить её. Ваша программа должна напечатать последовательность номеров, по которым будут разнесена почта, когда почтальон начинает работу:

- с первого дома;
- со второго дома;
- с N-го (последнего) дома.

2.14 Процедура вывода делителей числа

Источник: "Песни о Паскале" 20-Б

Создайте процедуру, печатающую все числа, кроме единицы, на которые без остатка делится число **N**, где **N** – параметр процедуры. Напишите программу для проверки этой процедуры.

2.15 Проверка наличия буквы в строке

Источник: "Песни о Паскале" 23-А

Напишите функцию для поиска буквы в заданной строке. Она должна возвращать **TRUE**, если в строке есть хоть одна эта буква, и **FALSE** в противном случае. Напишите программу для проверки функции.

2.16 Поиск буквы в строке

Источник: "Песни о Паскале" 23-Б

Напишите функцию для определения позиции буквы в заданной строке. Функция должна вернуть позицию первой такой буквы или ноль, если буквы в строке нет. Напишите программу для проверки функции.

2.17 Печать позиций букв в алфавитном порядке

Источник: "Песни о Паскале" 24-В

Для введенной пользователем строки напечатать позиции всех встречающихся в ней символов, кроме пробелов, в алфавитном порядке. Для символов, которые встречаются несколько раз, напечатать позиции в одной строке. Например, для слова «PASCAL»:

```
A – 2 5
C – 4
L – 6
p – 1
s – 3
```

2.18 Печать позиций букв в порядке следования

Источник: "Песни о Паскале" 24-Г

Для введенной пользователем строки напечатать позиции всех встречающихся в ней символов, кроме пробелов, в порядке их следования в строке. Например, для слова «PASCAL»:

```
P – 1
A – 2 5
S – 3
C – 4
L – 6
```

2.19 Подсчёт символов в файле

Источник: "Песни о Паскале" 25-В

Напишите три функции для подсчета:

- количества строк в файле;
- количества видимых символов в файле;
- количества всех символов файла (объем файла).

Функции принимают один параметр – ссылку на файловую переменную. Напишите программу, подсчитывающую упомянутые выше характеристики файла.

2.20 Вывод строк файла

Источник: "Песни о Паскале" 25-Е

Напишите процедуру для вывода на экран **N**-й строки файла, где **N** – параметр процедуры. Воспользовавшись этой процедурой, напишите программу для распечатки строк файла в обратном порядке.

П о д с к а з к а : предварительно посчитайте количество строк в файле.

2.21 Нумерация строк файла

Источник: "Песни о Паскале" 26-Б

Напишите программу для нумерации строк файла. Строки исходного файла должны копироваться в конечный файл с добавлением в начале каждой строки её номера.

2.22 Разделение файла

Источник: "Песни о Паскале" 26-Г

Для передачи по интернету секретного текстового файла создать из него два других: в первый записать нечетные строки исходного файла, а во второй – четные.

2.23 Объединение файлов

Источник: "Песни о Паскале" 26-Д

Создать программу для объединения двух файлов: из первого составить нечётные строки конечного файла, а из второго – чётные (см. условие предыдущей задачи).

2.24 Дребезг контактов

Источник: "Песни о Паскале" 27-Г

Дребезг контактов – с этим явлением борются специалисты по электронике. Дребезг возникает при замыкании-размыкании кнопок, тумблеров, реле и других подобных устройств. Сигнал от контактов поступает в микропроцессор с некоторой периодичностью, скажем, 100 раз в секунду. Если контакт разомкнут, микропроцессор принимает «0», а если замкнут – «1». В ходе замыкания-размыкания контакт неустойчив, и процессор получает несколько чередующихся нулей и единиц, – программа процессора должна отфильтровать эти ложные срабатывания.

Ваша программа будет моделировать поведение микропроцессора. Входной файл содержит последовательность нулей и единиц (по одному символу в строке). Из первой строки берется исходное значение сигнала, а дальше сигнал на выходе программы должен формироваться так: если три подряд идущие значения совпадают, то берется это новое значение, а иначе сохраняется текущее, например:

```
На входе  На выходе
    0         0
    1         0
    0         0
    1         0
    1         0
    1         1
    0         1
```

Выходной файл должен содержать две колонки: входной и выходной сигналы.

3 Главы 29–34
Разные задачи (простые типы данных)

Теоретическая база

- Ввод данных из текстовых файлов
- Вывод данных в текстовые файлы
- Цикл с проверкой в начале
- Организация процедур и функций
- Способы передачи параметров в процедуры и функции

3.1 Распределение груза (1)

Источник: --

В состав из **N** вагонов надо загрузить **M** одинаковых ящиков так, чтобы их количество в соседних вагонах отличалось не более чем на единицу. Программа должна вывести количество ящиков в каждом из **N** вагонов.

Контрольные примеры:

Входные данные	Результат
N=7 M=20	2 3 3 3 3 3 3

3.2 Распределение груза (2)

Источник: --

В состав из **N** вагонов надо загрузить **M** одинаковых ящиков так, чтобы их количество в соседних вагонах отличалось не более чем на единицу, и вдобавок те вагоны, что вмещают больше ящиков, распределились вдоль состава по возможности равномерно. Программа должна вывести количество ящиков в каждом из **N** вагонов.

Контрольные примеры:

Входные данные	Результат
N=13 M=20	1 2 1 2 1 2 1 2 1 2 1 2 2

3.3 Отпуск сотрудников

Источник: "Песни о Паскале" 20-В

Два сотрудника подали своему начальнику заявления на отпуск. Первый попросил отпустить его в дни с **A1** по **B1** (дни отсчитываются с начала года), второй – с **A2** по **B2** день. Предполагается, что **A1<B1** и **A2<B2**. Однако дело требует, чтобы кто-то из сотрудников находился на рабочем месте. Мало того, при смене отдыхающих необходимо не менее 3-х дней их совместной работы – для передачи дел. Напишите программу с процедурой, принимающей четыре указанных выше параметра, и печатающую заключение о том, удовлетворяют ли заявления работников требованиям начальника.

3.4 Размышления у светофора

Источник: "Песни о Паскале" 20-Г

Подойдя к перекрестку, пешеход думает о том, переходить ли ему улицу, или остановиться. На решение влияет характер пешехода и еще два фактора: сигнал светофора и близость опасно движущегося транспорта. Напишите программу с процедурой, которая принимает и печатает решение в зависимости от переданных в неё трех параметров, а именно.

- Параметр A = true, если горит зеленый;
- Параметр B = true, если поблизости опасно движется транспорт;
- Параметр C – это число, определяющее характер пешехода так:

1 - послушный и осторожный – учитывает светофор и опасность;
2 - послушный, но беспечный – смотрит только на светофор;
3 - хитрый вольнодумец – идет только на красный, если это ничем не грозит;
4 - непримиримый вольнодумец – идет только на красный, невзирая ни на что;
5 - «экстремал» – идет только на красный, но так, чтобы грозила опасность;
6 - «безбашенный» – идет, несмотря ни на что;
7 - запуганный – никогда не идет через дорогу, а ищет подземный переход.

3.5 Количество дней в месяце

Источник: 1

Напишите функцию, принимающую год и месяц, и возвращающую количество дней в этом месяце с учётом високосности года. Если месяц указан неверно, возвращается 0.

3.6 Сравнение чисел

Источник: --

Программа читает из файла ряд целых чисел (количество чисел заранее неизвестно), а затем печатает «Равны» или «Неравны» в зависимости от того, равны ли все числа между собой.

3.7 Подсчёт отрицательных чисел чисел

Источник: --

Программа читает из файла ряд целых чисел и подсчитывает количество отрицательных. Дополнительное требование: не использовать условные операторы **IF**, **CASE**.

П о д с к а з к а . Выражение **Ord(A>B)** даёт 1, если **A>B**, и ноль в противном случае.

3.8 Наблюдения температуры

Источник: --

Входной файл содержит значения температуры атмосферы, зафиксированные через некоторые интервалы времени. Определите, сколько раз за время наблюдения происходили превращения воды в лёд и обратно.

3.9 Нахождение дробной части числа

Источник: "Песни о Паскале" 30-А

Функция **Trunc** выделяет целую часть вещественного числа, например.

```
      Writeln (Trunc( 12.345 ));    { 12 }
```

Исследуйте её и придумайте способ выделения дробной части вещественного числа. Напишите подходящую функцию и программу для её проверки.

3.10 Дорогая покупка (1)

Источник: 1

Каждая строка входного файла содержит данные о покупке, то есть два числа: цену товара и его количество. Напечатать информацию о самой дорогой покупке: цену, количество и стоимость.

3.11 Дорогая покупка (2)

Источник: 1

Каждая строка входного файла содержит данные о покупке: 1) цену товара, 2) его количество и 3) наименование. Напечатать наименование и стоимость самой дорогой покупки.

3.12 Отбор учеников

Источник: "Песни о Паскале" 31-Б

Файл с физическими данными старшеклассников содержит три колонки: фамилия, рост и вес ученика. Создайте программы для решения следующих задач:

- отбор кандидатов для занятий баскетболом, – рост кандидата должен составлять не менее 175 см;
- поиск учеников с избыточным весом, для которых разница между ростом ученика (см) и его весом (кг) составляет менее 100.

Ваши программы должны сформировать соответствующие файлы с фамилиями и данными учеников.

3.13 Вычисление температуры воды

Источник: --

В ванну вылито несколько сосудов воды с разной температурой, надо напечатать температуру получившейся смеси с точностью до десятой доли градуса. Входной файл содержит по два числа в каждой строке: массу воды в сосуде и её температуру.

П о д с к а з к а . Температура смеси равна частному от деления содержащейся в воде теплоты на массу воды. А количество теплоты равно произведению количества воды на её температуру.

Контрольные примеры:

Входные данные	Результат
5 12 7 40 3 60	34,7

3.14 Классификация последовательности

Источник: --

Файл содержит последовательность целых чисел (возможно, что ни одного числа). Программа должна классифицировать файл, выведя на экран код классификации в соответствии со следующей таблицей.

Код	Классификация	Описание
0	Пустой	Файл не содержит чисел
1	Единичный	Файл содержит одно число
2	Постоянный	Все числа в файле одинаковы
3	Неубывающий	Каждое следующее число больше предыдущего, или равно ему
4	Возрастающий	Каждое следующее число больше предыдущего
5	Невозрастающий	Каждое следующее число меньше предыдущего, или равно ему
6	Убывающий	Каждое следующее число меньше предыдущего
7	Немонотонный	Все прочие случаи (встречаются и убывающие, и возрастающие пары чисел)

Совет. Во избежание влияния пустых символов и строк, при чтении файла применяйте функции **`SeekEoln`** и **`SeekEof`**.

3.15 Слежка за автомобилем

Источник: --

От спутниковой системы слежения за грузовиком получен файл, содержащий пройденные им расстояния по маршруту, зафиксированные через каждую минуту. Числа в файле образуют неубывающую последовательность, а когда встречается повторение нескольких чисел подряд, это значит, что здесь грузовик останавливался на отдых.

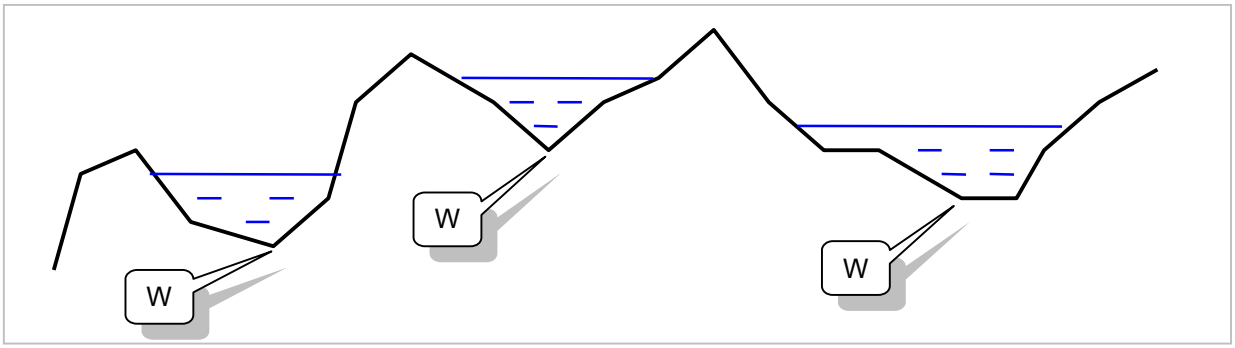
Программа должна определить: 1) количество остановок, 2) общую продолжительность остановок и 3) наиболее продолжительную остановку.

3.16 Сканирование марсианской поверхности

Источник: "Песни о Паскале" 30-Г

Сканирование марсианской поверхности дало файл, содержащий высоту отдельных его точек вдоль одного из направлений. Найдите точки, где вероятней всего обнаружить

марсианскую воду, – на следующем ниже рисунке они обозначены буквами W. Программа должна напечатать две колонки: порядковый номер точки относительно начала файла (счет от нуля) и высоту точки (такие точки математики называют локальными минимумами).



3.17 Автоматическая дверь

Источник: --

Автоматическая дверь супермаркета услужливо пропускает посетителей. Дверь отворяется плавно, степень её открытия определим следующими числами: 0 — полностью закрыта, 1 — открыта на одну четверть, 2 — наполовину, 3 — на ¾, и 4 — полностью открыта. Степень открытия зависит от направления движения посетителя и его скорости. Если посетитель приближается к двери, то, в зависимости от времени его предполагаемого достижения двери, дверь открывается так:

Время предполагаемого достижения двери, секунды	Степень открытия двери
более 4	0
от 4 до 3	1
от 3 до 2	2
от 2 до 1	3
1 и менее	4

При удалении посетителя дверь сразу закрывается.

Дан файл целых чисел, содержащий расстояния посетителей до двери в сантиметрах, зарегистрированные с интервалом в 1 секунду. Расстояния могут быть отрицательными (когда посетитель слева от двери) и положительными (когда посетитель справа от двери). Создайте выходной файл, содержащий состояния двери в разные моменты времени.

П о д с к а з к а . Время, необходимое для подхода к двери, определяется делением расстояния до двери на скорость посетителя, которая, в свою очередь определяется разностью двух соседних значений расстояния. Так получаем формулу для времени:

$$T_i = L_i / (L_i - L_{i-1})$$

Здесь L_i *и* L_{i-1} — значения расстояния для двух соседних моментов времени.

3.18 Не думай о секундах свысока

Источник: "Песни о Паскале" 32-Г

«Не думай о секундах свысока…». Штирлицу подарили секундомер, который показывал секунды, прошедшие с начала суток. Пусть ваша программа переведет это число в привычные часы, минуты и секунды. Вот две подсказки: во-первых, воспользуйтесь операциями **DIV** и **MOD**. Во-вторых, объявляя переменную для секунд, примените тип **LONGINT** (а не **INTEGER**), поскольку количество секунд в сутках (86400) не поместится в типе **INTEGER**.

3.19 Проверка на степень

Источник: 1

Напишите булеву функцию, принимающую два целых числа **M** и **N**, и возвращающую **TRUE,** если первое из чисел является степенью второго. То есть, когда все делители числа **M** без остатка делятся на **N**.

Входные данные	Результат
15 3	false
27 3	true
27 2	false
32 2	true

3.20 Возведение в степень (1)

Источник: 12

Напишите функцию для возведения действительного числа **R** в целую степень **N**.

П о д с к а з к а . Допускается упрощённое решение перемножением действительного числа **N** раз.

3.21 Возведение в степень (2)

Источник: 12

Напишите функцию для возведения действительного числа **R** в целую степень **N** ускоренным способом так, чтобы количество умножений действительного числа составляло порядка **Log2(N)** .

П о д с к а з к а . В ходе перемножения надо анализировать чётность/нечётность текущего показателя степени.

3.22 Квадратный корень

Источник: "Песни о Паскале" 33-Е

Квадрат – это равносторонний прямоугольник, его площадь вычисляется по формуле **S=D•D**, где **D** – сторона квадрата. А когда площадь **S** известна, и надо определить сторону **D**? Тогда из **S** извлекают квадратный корень (обозначается символом √) . Так, если **S=9**, то **D=√9=3**.

Для извлечения корня в Паскале есть функция **SQRT**. Напишите собственную функцию **MySQRT**, прибегнув к методу последовательных приближений. В грубом, нулевом приближении примем $D_0=1$. Последующее, более точные значения **D** будем вычислять по формуле

$$D_{i+1} = (D_i + S/D_i)/2$$

Так, при **S=9** получим $D_1=(1+9/1)/2=\underline{5}$, $D_2=(5+9/5)/2=\underline{3.4}$ и так далее, пока абсолютная разность между двумя последовательными значениями **D** станет пренебрежимо мала. Функция **MySQRT** должна принять число и вычислить его корень с точностью **0.0001**. Внутри функции напечатайте промежуточные значения **D**.

П о д с к а з к а : для D_i и D_{i+1} вам потребуются лишь две локальные переменные.

3.23 Состав теста

Источник: "Песни о Паскале" 33-Ж

В тесто кладут четырех главных ингредиента: муку, сахар, яичный порошок и молоко. Все это смешивается в пропорции, заданной рецептом. Например, рецепт **100:5:7:500** означает, что на 100 граммов муки кладут 5 граммов сахара, 7 граммов яичного порошка и 500 граммов молока. У пекаря есть некоторое количество всех ингредиентов, и он намерен замесить из них максимально возможное количество теста по данному рецепту. В программе вводятся:

- Рецепт – это 4 целых числа.
- Исходное количество ингредиентов – это 4 действительных числа.

Программа должна напечатать:

- Общее количество полученного теста с точностью два знака после точки.
- Остатки ингредиентов – 4 числа с точностью два знака после точки.

3.24 «Добрый» банкир

Источник: --

Папа Карло получал жалованье один раз в месяц. Но однажды ему пришлось взять в банке кредит в объёме своей месячной зарплаты. «Вернёшь мне его с процентами в любое время, но не позднее года, а иначе я заберу твою каморку», — поставил условие «добрый» банкир. Будучи человеком исправным, папа Карло принёс долг через месяц (как только заработал нужную сумму). «Молодец, — хлопнул его по плечу банкир, — ты хороший парень, и я доверяю тебе. Оставь себе деньги, пусть это будет новый кредит на тех же условиях, и мы продлим его ещё на 12 месяцев». Таким образом, долг папы увеличился. Заработав нужную сумму (больше прежней), папа Карло вновь принёс долг, но банкир опять уговорил его оставить деньги и продлить кредит. И так повторилось ещё несколько раз, пока папа Карло не просрочил платёж, лишившись своей каморки.

Нам известен процент **P**, который назначил банкир. Надо определить, сколько месяцев прошло от взятия первого кредита до момента, когда папа Карло оказался бездомным.

П о д с к а з к а : долг с учётом процента вычисляется по формуле:

```
долг = кредит • (100 + P) / 100
```

3.25 Букеты

Источник: --

Флорист составляет букеты по пять роз, используя белые (Б) и красные (К) цветы. Пусть программа напечатает все возможные букеты из пяти роз одного или двух цветов: (от ББББ, БББК ... до КККК).

3.26 Комплексные обеды

Источник: --

Комплексный обед включает три блюда: первое, второе и третье. Столовая располагает тремя первыми блюдами (обозначим их цифрами 1,2,3), четырьмя вторыми (A,B,C,D) и тремя третьими (a,b,c). Напечатайте все варианты комплексных обедов, которые можно составить из имеющихся блюд (1Aa, 1Ab и т.д).

Сделайте второй вариант программы, где количество разных блюд вводит пользователь, а принцип наименования блюд тот же.

3.27 Флаги

Источник: --

Некая страна, добившись независимости, задумала учредить трёхцветный государственный флаг. Три его полосы должны делить полотнище на три равные части: верхнюю, среднюю и нижнюю. В распоряжении дизайнеров было три цветных ткани: белая (Б), синяя (С) и красная (К). Пусть программа напечатает все возможные раскраски флага и количество раскрасок с учётом следующих условий (три варианта программы):

а) цвета всех полос должны быть разными;

б) хотя бы одна из полос должна отличаться (цвета двух других могут совпадать), при этом положение полос имеет значение;

в) любая полоса может иметь любой цвет, но положение полос имеет значение.

3.28 Домино

Источник: --

На костяшках домино числа изображаются точками. Рассмотрим комплект костяшек **N**-домино, где **N** — это максимальное число, изображаемое на костяшках. Комплект **N**-домино составят кости от **(0:0)** до **(N:N)**. Пользователь вводит **N**, а программа должна напечатать общее количество точек на костяшках. Оформите вычисление в виде функции.

Контрольные примеры:

Входные данные	Результат
1	3
2	12

3.29 Код сейфа

Источник: --

Сейф в кабинете Мюллера охранялся шифр-замком из четырёх цифр. Штирлицу стало известно, что цифры кода образуют возрастающую последовательность. Подсчитайте максимальное количество комбинаций, которое надо перебрать для открытия сейфа.

3.30 Горизонтальная табуляция

Источник: --

Управляющий символ горизонтальной табуляции **Chr(9)** служит для выравнивания колонок текстового файла. Пусть входной файл содержит фамилии людей. Программа должна создать выходной файл с двумя колонками: слева — фамилия, а справа — количество букв в ней. Между колонками поставьте два знака горизонтальной табуляции.

3.31 Шифрование и расшифровка файла

Источник: --

Шифровка Штирлица содержала последовательность чисел (в каждой строке файла по одному числу). Чтобы окончательно запутать врага, Штирлиц подверг файл дополнительной шифровке: каждую цифру исходного числа он заменял цепочкой ненулевых случайных чисел, ограниченную нулём так, что длина такой цепочки чисел соответствовала одной цифре исходного числа, вот к примеру два варианта шифрования числа 301:

```
      3      0    1
   <------>  <>   <->
   76 91 23 0  0  7 0

   12 16 99 0  0 32 0
```

Здесь ноль представлен пустой цепочкой чисел.

Напишите программу для зашифровки и расшифровки числового файла этим способом.

3.32 Универсам

Источник: --

В универсаме три кассы, и каждая обслуживает по одному человеку в минуту. Но не всегда они работают одновременно. Если очередь посетителей не превышает 5 человек, то работает один кассир, а иначе зовут второго. Если двое не справляются и очередь удлиняется сверх 10 человек (суммарно в двух кассах), то сажают третьего. С уменьшением очереди кассиры соответственно освобождаются.

Каждая строка входного файла содержит число от 0 до 7 — это количество покупателей, подходящих к кассам в данную минуту. Надо создать другой файл, в каждой строке которого содержатся по два числа: длина очереди в текущую минуту и количество работающих кассиров.

3.33 Взаимно простые числа

Источник: --

Функция принимает два целых положительных числа **N** и **M**, и возвращает **TRUE**, если эти числа взаимно простые. Взаимно простые числа не имеют общих делителей, кроме единицы.

Контрольные примеры:

Входные данные	Результат
15, 22	TRUE
15, 21	FALSE

3.34 Большее из двух

Источник: --

Пусть даны две целочисленные переменные **A** и **B**. Переменной **C** можно присвоить большее значение из двух следующим условным оператором:

```
if A>B then C:= A else C:=B
```

Сделайте то же самое одним выражением без условного оператора.

П о д с к а з к а : Выражение **Ord(A>B)** равно либо 1 либо 0.

4 Главы 35–38 Множества

Теоретическая база

- Тип данных «множество»
- Ввод и вывод множеств
- Операции с множествами
- Передача множеств в качестве параметров процедур

4.1 Преобразование строки

Источник: "Песни о Паскале" 37-Г

Напишите две функции, принимающие строку и возвращающие:

- строку, в которой символы исходной строки встречаются лишь по разу и следуют в алфавитном порядке, например «PASCAL» → «ACLPS»;
- то же, но порядок следования символов такой же, как в исходной строке, например «PASCAL» → «PASCL».

4.2 Анаграммы (1)

Источник: --

Анаграммами называются слова, состоящие из одних и тех же букв. Напишите булеву функцию, принимающую по ссылке **`CONST`** две строки, и возвращающую **`TRUE`**, если строки состоят из одних и тех же символов (количество каждого символа не имеет значения). Напишите программу для проверки функции (ввести две строки).

4.3 Автомобильный парк

Источник: "Песни о Паскале" 38-Б

В небольшом островном государстве действовали забавные законы по части транспортных средств – автобусов, грузовиков и легковушек. Во-первых, общее количество автомобилей на острове не должно было превышать 256. Автомобилям назначались номера с 0 до 255, при этом соблюдались следующие правила.

Номера, делившиеся без остатка на 7, назначались автобусам. Те, что делились без остатка на 5, назначались грузовикам, а все прочие – легковушкам. Так, например, номера 35 и 70 (делятся и на 7, и на 5) доставались автобусам, а не грузовикам.

Схожие правила применялись и к окраске автомобилей, а именно.

Если номер авто делился на 4, его красили красным, если на 3 – желтым, если на 2 – белым, а остальные автомобили красили черным.

- Сформируйте три множества по классам автомобилей – автобусы, грузовики и легковушки. Вычислите количество машин каждого класса (Ответ: 37, 44, 175).
- Сформируйте четыре множества по цвету автомобилей – красные, желтые, белые и черные. Определите количество машин каждого цвета (Ответ: 64, 64, 43, 85).
- Столица того государства – деревня Кокосовка – страдала от пробок. Для борьбы с ними ввели ограничение на въезд транспорта. Так, в один из дней недели в столицу пускали только красные легковушки, белые грузовики и все автобусы. Найдите номера всех этих машин. Сколько всего автомобилей могло въехать в столицу в тот день?

4.4 Сортировка базы данных

Источник: "Песни о Паскале" 38-В

Полицейская база островного государства содержала номера угнанных автомобилей – числа от 1 до 255. Это был тестовый файл такого, например, вида:

```
120 31 16 25
```

То есть, номера перечислялись через пробел и следовали в произвольном порядке, что неудобно при поиске «вручную». Ваша программа должна создать файл с номерами, упорядоченными по возрастанию. Подсказка: примените множество чисел.

4.5 Генерация пароля

Источник: "Песни о Паскале" 38-Г

Генерация пароля длиной не менее восьми символов. В пароль входят символы трёх сортов: цифры, маленькие и большие латинские буквы, например: «**7UpJ7rsT**», «**PasCal701**». Сделайте четыре варианта так, чтобы соблюдались следующие условия:

- символ каждого сорта входит в пароль не менее двух раз, некоторые символы могут повторяться;
- все символы пароля уникальны (примените множество);
- символы одного сорта не соседствуют, например: «Pa7sCaL5», уникальность символов не требуется;
- символы одного сорта не соседствуют и все символы уникальны.

4.6 Проверка качества пароля

Источник: "Песни о Паскале" 38-Д

Напишите булевы функции, проверяющие, является ли введенная пользователем строка правильно сформированным паролем согласно условиям предыдущей задачи.

4.7 Сотрудники предприятия

Источник: --

Сотрудникам предприятия назначены уникальные табельные номера — числа от 1 до 255. Предприятие содержит ряд отделов, обозначаемых буквами A–Z.

Дан файл, где каждая строка содержит перечень сотрудников, работающих в некотором отделе (отделы следуют в алфавитном порядке). Известно, что сотрудник может работать лишь в одном отделе. Программа должна определить, содержит ли данный файл ошибки, когда один сотрудник указан в нескольких отделах. В случае ошибок вывести таких сотрудников и перечень отделов, куда они ошибочно причислены.

Б о н у с . Создайте программу для случайного формирования безошибочного файла по условию данной задачи.

4.8 Семейные пары

Источник: --

Сотрудникам предприятия назначены уникальные табельные номера, это числа от 1 до 255, причём мужчинам присвоены нечётные номера, а женщинам — чётные. Семейным парам назначены соседние числа, например, 7 (муж) и 8 (жена), или 25 и 26. Предприятие содержит ряд отделов, обозначаемых буквами A–Z.

Дан файл, каждая строка в нём содержит перечень сотрудников, работающих в некотором отделе (отделы следуют в алфавитном порядке). Необходимо вывести те отделы, где трудятся семейные пары, а также сами эти пары, например:

```
D 13 14 45 46
F 15 16
```

4.9 Актёры и фильмы

Источник: --

В некоторой стране актёров и актрис именовали строчными латинскими буквами a–z, а фильмы нумеровали числами от 1 до 255.

Дан файл, каждая строка которого начинается с буквы (актёра), за которой следуют одно или несколько чисел — это фильмы, в которых он снялся. Программа должна создать файл, каждая строка которого начинается с числа (фильма), за которым следуют буквы — это снявшиеся в фильме актёры и актрисы.

4.10 Режиссёры и актёры

Источник: --

В некоторой стране актёров и актрис именовали строчными латинскими буквами a–z, режиссёров — прописными буквами A–Z, а фильмы нумеровали числами от 1 до 255.

Даны два файла. В первом каждая строка начинается со строчной буквы (актёра), за которой следуют одно или несколько чисел — это фильмы, в которых он снялся. Во втором каждая строка начинается с прописной буквы (режиссёра), за которой следуют одно или несколько чисел — это снятые им фильмы. Известно, что режиссёр и актёр (актриса) являются супругами, если они обозначены одинаковыми буквами, например: **`F`** и **`f`**, **`D`** и **`d`**.

Программа должна вывести режиссёров, снявших в фильмах своего супруга, и сами эти фильмы.

4.11 Количество остановок

Источник: --

Остановки городского транспорта пронумерованы числами от 1 до 255. Автобус следует по круговому маршруту, посещая часть этих остановок в порядке от меньших номеров к большим.

Входной файл содержит две строки. В первой перечислены все остановки автобуса в произвольном порядке (а не в порядке их посещения). Во второй даны два числа — это исходная и конечная остановки, между которыми надо проехать пассажиру. Пусть программа определит количество проезжаемых пассажиром остановок (1 — если это соседние остановки).

4.12 Посещение достопримечательностей

Источник: --

Маршруты городских автобусов названы заглавными буквами латинского алфавита **`A`**–**`Z`**, а остановки автобусов обозначены числами. Некоторые маршруты могут частично совпадать (когда автобусы посещают одни и те же остановки). Некоторые остановки расположены у достопримечательностей, посещаемых туристами.

Входной файл содержит следующие данные. В первой строке — несколько чисел, — это номера остановок, которые желает посетить турист. Последующие строки — это перечень остановок каждого из городских маршрутов в их алфавитном порядке; здесь каждый маршрут представлен отдельной строкой чисел.

Турист может воспользоваться лишь одним из маршрутов так, чтобы посетить максимально возможное количество нужных ему остановок. Пусть программа выберет один из таких маршрутов, если он существует.

4.13 Надёжность пароля

Источник: --

Для регистрации на сетевом сервисе требуется ввести надёжный пароль, который должен отвечать следующим требованиям:

- длина пароля — не менее 8 символов;
- в пароле должна быть хотя бы одна строчная буква a...z:
- в пароле должна быть хотя бы одна заглавная буква A...Z:
- в пароле должна быть хотя бы одна цифра 0...9.

Пользователь вводит строку, а программа печатает, скольким критериям он соответствует.

Контрольные примеры:

Входные данные	Результат
QWERTY12345	3
123	1
QWER123ty	4

4.14 Шифровальная таблица

Источник: --

Для передачи донесений Штирлиц использовал символы с кодами от 32 до 127 (всего 96). При зашифровке символы заменялись в соответствии с шифрующей таблицей, которая состояла из 48 пар символов (это файл в две колонки). Для зашифровки символа Штирлиц находил его в таблице (в любой из двух колонок) и менял на символ из другой колонки. Расшифровка выполнялась точно так же.

В целях безопасности Штирлиц время от времени обновлял шифрующую таблицу. Напишите программу для случайной генерации такой таблицы. Результатом работы вашей программы будет файл из 48 пар символов: по два символа в строке, разделённых пробелом.

Б о н у с : напишите программу для зашифровки и расшифровки файла по описанному выше методу. Символы, не попавшие в таблицу, не шифровать.

4.15 Танковые экипажи (1)

Источник: --

Командиру полка из вновь поступивших бойцов надо сформировать танковые экипажи по три человека в каждом, а именно: 1) командир-наводчик, 2) механик-водитель и 3) стрелок-радист. Новобранцы были предварительно обучены и представлены полковнику тремя списками соответственно своим специальностям.

Дан файл из трёх строк, где каждая строка содержит ряд чисел от 0 до 255 — это закодированные фамилии бойцов трёх специальностей, перечисленные в произвольном порядке. Ваша программа должна:

- проверить, не содержат ли списки ошибок (каждый боец должен состоять лишь в одном из трёх списков);
- если ошибок нет, сформировать ряд экипажей и вывести их каждый в отдельной строке;
- если в списках остались бойцы, которым не нашлось места в экипажах, вывести их на экран.

4.16 Танковые экипажи (2)

Источник: --

Пусть по условиям предыдущей задачи (4.15) были сформированы два файла следующего содержания: в первом три строки — это перечисления кодов бойцов трёх специальностей, а второй файл содержит перечисления составленных из них танковых экипажей (по три числа в строке). Экипажи составлены вручную, и потому файл может содержать ошибки. Ваша программа должна проверить файлы на предмет следующих ошибок:

- какой-либо боец состоит более чем в одном из трёх списков;
- в одном экипаже оказались бойцы одной специальности (два или три);
- какой-либо боец оказался более чем в одном танковом экипаже.

4.17 Выбор магазина

Источник: --

Товары в городских магазинах закодированы числами от 0 до 255. Входной файл содержит следующие данные: первая строка — перечисление кодов товаров, которые надо приобрести покупателю (в произвольном порядке), а последующие строки содержат перечисления кодов товаров, которые имеются в нескольких окрестных магазинах (каждый магазин — отдельной строкой). За неимением времени, покупатель может успеть лишь в один из магазинов. Подберите магазин, где он может удовлетворить свою

потребность в наибольшей степени. Магазины нумеруются с единицы, однако номера магазинов в файле не указаны, а лишь подразумеваются.

4.18 Поход в магазины

Источник: --

Товары в городских магазинах закодированы числами от 0 до 255. Входной файл содержит следующие данные: первая строка — перечисление кодов товаров, которые необходимы покупателю (в произвольном порядке), а последующие строки содержат перечисления кодов товаров, которыми торгуют в нескольких магазинах (каждый магазин — отдельной строкой). Магазины перечислены в порядке удалённости от покупателя. Программа должна вывести номера ближайших магазинов, которые придётся ему посетить. Магазины нумеруются с единицы, однако номера магазинов в файле не указаны, а лишь подразумеваются.

4.19 Доставка товаров (1)

Источник: --

Оптовый поставщик снарядил грузовик с товарами для доставки их в городские магазины, товары закодированы числами от 0 до 255. Грузовик посещает магазины последовательно, а магазин берёт нужные ему товары только тогда, когда этого товара в магазине нет, причём забирает весь такой товар из грузовика.

Входной файл содержит следующие данные: первая строка — перечисление кодов товаров, которые изначально загружены в грузовик (в произвольном порядке); последующие строки содержат перечисления товаров, которые уже имеются в нескольких магазинах (каждый магазин — отдельной строкой). Магазины перечислены в порядке посещения их грузовиком. Программа должна вывести перечни товаров в магазинах после посещения их грузовиком (каждый магазин — отдельной строкой), а также коды товаров, оставшихся в грузовике.

5 Главы 39–40 Обработка массивов

Теоретическая база

- Понятие массива
- Ввод и вывод массивов
- Перебор элементов массива
- Передача массивов в качестве параметров процедур

5.1 Заполнение массива

Источник: 4

Объявите массив из 10 целых чисел (с индексами от 1 до 10) и заполните его несколькими способами, а именно:

- нулями;
- нарастающими значениями от 2 до 20;
- убывающими значениями от 10 до 1;
- случайными значениями от 0 до 9999.

После каждого заполнения распечатайте массив (выведите на экран). Для распечатки массива создайте специальную процедуру.

5.2 Поиск минимумов и максимумов

Источник: 4

Объявите массив из 10 целых чисел (с индексами от 1 до 10) и заполните его случайными числами от 0 до 9999. Затем определите в массиве:

- минимальный элемент и его индекс;
- максимальный элемент и его индекс.

Контрольные примеры:

Входные данные	Результат
57 85 95 40 63 89 75 33 17 11	11 10 95 3

5.3 Функции обработки массива

Источник: 4

Объявите массив из 10 целых чисел (с индексами от 1 до 10) и напишите несколько функций, возвращающих:

- минимальный элемент;
- максимальный элемент;
- сумму элементов массива;
- количество чётных элементов массива.

Функции должны принимать массив как параметр по ссылке **CONST**.

Контрольные примеры:

Входные данные	Результаты
57 85 95 40 63 89 75 32 17 11	11 – минимальный
	95 – максимальный
	564 – сумма
	2 – количество чётных

5.4 Процедуры удаления и вставки элементов

Источник: 4

Объявите массив из 10 целых чисел и напишите процедуры для следующих операций:

- Удаления элемента с заданным индексом: удаляемый элемент заменяется стоящим справа, а все последующие также сдвигаются влево. Последний элемент не удаляется, а дублируется. Процедура принимает два параметра: ссылку на массив и индекс удаляемого элемента.
- Вставки элемента с заданным индексом: вставляемый элемент заменяет текущий, а текущий и все последующие сдвигаются вправо. Последний элемент теряется. Процедура принимает три параметра: 1) ссылку на массив, 2) позицию вставки и 3) вставляемый элемент.

Процедуры должны принимать массив как параметр по ссылке **VAR**.

Контрольный пример на удаление:

Входные данные	Результат
2 4 6 8 10 12 14 16 18 20 6 – индекс удаляемого	2 4 6 8 10 14 16 18 20 20

Контрольный пример на вставку:

Входные данные	Результат
2 4 6 8 10 12 14 16 18 20 6 – место вставки 11 – вставляемое значение	2 4 6 8 10 11 12 14 16 18

5.5 Вращение массива вправо и влево

Источник: --

Напишите две процедуры, принимающие массив по ссылке **VAR**, одна из которых вращает массив влево, а другая вправо.

При вращении влево первый элемент становится последним, последний — предпоследним и т.д., а второй становится первым.

При вращении вправо первый элемент становится вторым, второй — третьим и т.д., а последний становится первым.

Контрольный пример:

Входные данные	Результат
1 2 3 4 5 6 7 8 9 10	2 3 4 5 6 7 8 9 10 1 – влево
1 2 3 4 5 6 7 8 9 10	10 1 2 3 4 5 6 7 8 9 – вправо

5.6 Накопление расстояния (интегрирование)

Источник: 1

Массив размера **N** содержит значения скорости автомобиля, фиксируемые через каждый час (это средняя скорость за прошедший час). Необходимо сформировать массив, содержащий преодолённое автомобилем расстояние с момента начала движения.

5.7 Вычисление скорости (дифференцирование)

Источник: 1

Массив возрастающих чисел содержит преодолённое автомобилем расстояние с момента начала движения. Расстояние фиксировалось каждый час. Необходимо сформировать массив, содержащий среднюю скорость автомобиля в течение каждого часа.

5.8 Фильтрация сигнала

Источник: 1

Массив размера **N** содержит результаты измерений, полученные от датчика через равные промежутки времени. Сигнал от датчика изменяется плавно, но, ввиду помех, отдельные результаты слегка «прыгают» в ту или иную сторону. Для снижения шума (так это явление называют инженеры) массив подвергают цифровой фильтрации так: первый и последний элементы не изменяют, а все прочие заменяют средним арифметическим значением трёх соседних:

```
Y[i] = (X[i-1] + X[i] + X[i+1])/3
```

Напишите процедуру фильтрации числового массива, переданного по ссылке **VAR**. Сделайте два варианта: с использованием вспомогательного массива и без него.

5.9 Велосипедная передача

Источник: --

В цепной передаче велосипеда есть две группы звёздочек: передняя и задняя. Переднюю составляют **M** звёздочек, количество зубьев на каждой из них представим массивом возрастающих чисел **P[1..M]**, где **P[1]<P[2]<P[3]** и т.д. Соответственно заднюю группу шестерен из **N** звёздочек представим массивом убывающих чисел **Q[1..N]**, где **Q[1]>Q[2]>Q[3]** и т.д.

Предположим, что велосипедист вращает педали с постоянной скоростью, при этом фактическая скорость велосипеда **Vфакт** определяется по формуле:

```
V факт = K • P[i] / Q[j]
```

Здесь **K** — известный постоянный коэффициент; **i**, **j** — индексы задействованных звёздочек. Предположим, также, что велосипедист может включать любую комбинацию звёздочек (хотя в реальности это не так).

В зависимости от рельефа местности (подъём/спуск) и направления ветра (встречный/попутный) велосипедист подбирает некую оптимальную скорость **Vопт**, которую считаем известной. Итак, нам даны два массива звёздочек **P** и **Q**, коэффициент **K**

и требуемая оптимальная скорость **Vопт**. Программа должна подобрать комбинацию соответствующих звёздочек так, чтобы фактическая скорость **Vфакт** была как можно ближе к оптимальной **Vопт**.

5.10 Процедура для зеркальной перестановки элементов

Источник: 4

Объявите массив из 10 целых чисел, заполните случайным образом, и посредством процедуры выполните зеркальную перестановку элементов: первого и последнего, второго и предпоследнего и т.д.

Контрольный пример:

Входные данные	Результат
57 85 95 40 63 89 75 33 17 11	11 17 33 75 89 63 40 95 85 57

5.11 Проверка на одинаковость

Источник: --

Напишите функцию, принимающую числовой массив по ссылке **CONST**, и возвращающую **TRUE**, если все элементы массива одинаковы.

Контрольный пример:

Входные данные	Результат
57 85 95 40 63 89 75 33 17 11	FALSE
33 33 33 33 33 33 33 33 33 33	TRUE

5.12 Проверка на неубывание

Источник: --

Напишите функцию, принимающую числовой массив (по ссылке **CONST**), и возвращающую **TRUE**, если элементы массива расположены по неубыванию, то есть, последующий не меньше предыдущего.

Контрольный пример:

Входные данные	Результат
10 12 20 15 18 17 35 10 48 71	FALSE
11 15 15 15 17 26 32 45 45 63	TRUE

5.13 Проверка на уникальность

Источник: --

Дан массив, содержащий числа в пределах от 0 до 255. Напишите функцию, принимающую массив по ссылке **CONST**, и возвращающую **TRUE**, если все его элементы уникальны, то есть, встречаются лишь по разу.

П о д с к а з к а : примените множества.

Контрольный пример:

Входные данные	Результат
10 12 20 15 18 17 35 95 48 71	TRUE
11 15 15 15 17 26 32 45 45 63	FALSE

5.14 Проверка на чередование

Источник: --

Напишите функцию, принимающую массив по ссылке **CONST**, и возвращающую **TRUE**, если все его элементы (кроме первого и последнего) расположены так, что слева и справа располагаются числа, либо большие, либо меньшие текущего.

Контрольный пример:

Входные данные	Результат
10 30 20 25 18 40 35 95 48 71	TRUE
11 15 15 15 17 26 32 45 45 63	FALSE

5.15 Доставка товаров (2)

Источник: --

Оптовый поставщик снарядил грузовик со штучными товарами для доставки их в магазины. Номенклатура составляет **N** товаров, которым соответствуют индексы от 1 до **N**. Грузовик посещает магазины последовательно, а магазин берёт нужные ему товары лишь тогда, когда наличие их в магазине составляет менее 10 штук (магазин добирает до 10 или забирает из грузовика остатки нужных ему товаров).

Входной файл составляют строки по **N** чисел в каждой. Первая строка содержит количество товаров взятых в грузовик (ноль, если какой-то товар не загружен). Последующие строки перечисляют начальное количество товаров в магазинах (каждый магазин — отдельной строкой). Магазины даны в порядке посещения их грузовиком. Программа должна сформировать файл с наличием товаров в магазинах после посещения

их грузовиком (каждый магазин — отдельной строкой), а в последней строке показать остаток товаров в грузовике.

5.16 Доставка товаров (3)

Источник: --

Оптовый поставщик снарядил грузовик со штучными товарами для доставки их в магазины. Номенклатура составляет **N** товаров, которым соответствуют индексы от 1 до **N**. Магазин берёт нужные ему товары лишь тогда, когда наличие их в магазине составляет менее 10 штук (магазин добирает до 10 или берёт из грузовика остатки нужных ему товаров). Для снижения пробега грузовика необходимо посетить как можно меньше магазинов, то есть избавиться от товаров побыстрее.

Входной файл составляют строки по **N** чисел в каждой. Первая строка содержит количество товаров взятых в грузовик (ноль, если какой-то товар не загружен). Последующие строки перечисляют начальное количество товаров в **i**-м магазине (каждый магазин — отдельной строкой). Программа должна выдать оптимальную последовательность посещения магазинов, магазины нумеруются с 1 в порядке их представления в файле.

5.17 Смешанная эстафета

Источник: --

В биатлоне популярны смешанные эстафеты, где в каждой команде участвуют четверо: двое мужчин и две женщины. Однако не каждая страна может выставить такую команду.

Даны три массива размером **N**: массивы **M[]** и **W[]** содержат количество мужчин и женщин, выступающих от **i**-ой страны (количество составляет от 0 и выше), а массив **C[]** — названия стран. Надо вывести названия стран, способных выступить в смешанной эстафете.

5.18 Коммерческий бег

Источник: --

На коммерческих соревнованиях в беге на длинные дистанции иногда побуждают спортсменов ускоряться такой мерой: на первой части дистанции судьи не вмешиваются, а на последних **N** кругах на каждом круге снимают одного отставшего спортсмена. Поэтому к финишу приходит на **N** бегунов меньше, чем стартовало.

Дан файл из **N+1** строк, каждая строка которого содержит **M** чисел. Здесь **M** — это количество стартовавших бегунов (**M>N**). Первая строка содержит времена прохождения первой части дистанции в десятых долях секунды (это целые числа), а последующие строки — времена преодоления каждого из последующих **N** кругов при условии, как если

бы все спортсмены продолжали бег. Но на самом деле часть из них будет снята с дистанции, и программа должна определить номера снятых бегунов (их индексы) на каждом контрольном круге.

5.19 Кёрлинг

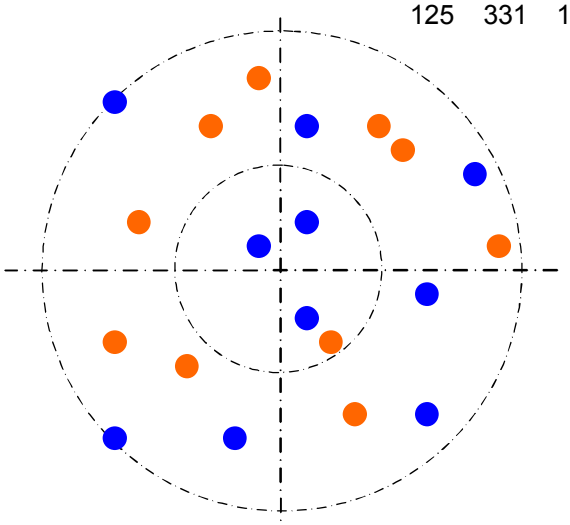
Источник: --

В кёрлинг играют две команды: «красная» и «синяя». Они поочерёдно толкают камни красного и синего цвета, стремясь поставить их как можно ближе к центру круга (так называемого «дома»). Выигрывает команда, камень которой окажется ближе камней соперника. Причём победителю начисляется 1 и более очков по количеству таких близких камней.

Даны два массива по 10 чисел (назовём их **`Red`** и **`Blue`**), содержащие результаты толкания камней в порядке очерёдности — расстояния от камней до центра в миллиметрах. Надо определить выигравшую команду и количество набранных ею очков.

Контрольный пример

Красные (Red)
120 332 115 210 189 153 205 211 146 105
Синие (Blue)
125 331 115 82 19 143 251 11 163 177



Результат: 3 очка в пользу «синих»

5.20 Отбор абитуриентов

Источник: --

Входной файл содержит четыре колонки: фамилию абитуриента и три числа — количество баллов по трём ЕГЭ. Чтобы попасть на бюджет, надо набрать не менее 200 баллов, причём оценка по любому из трёх экзаменов должна составлять не менее 60 баллов. Программа должна напечатать фамилии и баллы принятых абитуриентов.

5.21 Отсев простых чисел решетом Эратосфена

Источник: --

Напишите программу распечатки простых чисел в диапазоне от 1 до 1000 методом решета Эратосфена, применив массив чисел.

П о д с к а з к а : для вычёркивания чисел обнуляйте соответствующие элементы массива.

5.22 Подсчёт несовпадающих чисел (1)

Источник: 4

Дан массив размера **N**, заполненный случайными целыми числами в диапазоне от 0 до 255. Надо посчитать количество разных чисел в этом массиве.

П о д с к а з к а : воспользуйтесь множествами.

Контрольный пример:

Входные данные	Результат
10 2 3 4 5 6 2 8 2 10	7
1 2 1 2 1 1 1 1 1 1	2

5.23 Подсчёт несовпадающих чисел (2)

Источник: 4

Дан массив размера **N**, заполненный случайными целыми числами в диапазоне от 0 до 1023. Подсчитайте количество разных чисел в этом массиве.

П о д с к а з к а : примените вспомогательный массив того же размера.

5.24 Подсчёт разных чисел

Источник: 4

Дан массив размера **N**, заполненный случайными целыми числами в диапазоне от 1 до 20. Подсчитайте, сколько раз в массиве встречается каждое из этих чисел.

П о д с к а з к а : примените массив счётчиков.

5.25 Посещение кинозала

Источник: 4

Круглосуточный кинозал за сутки посетили **N** зрителей (**N=100**). Входить и уходить из зала можно в любое время. Для каждого зрителя известно время прихода и время ухода, причём время указано в минутах относительно начала суток (от 0 до 24*60-1 минут). Даны два массива размером **N**, первый содержит времена прихода зрителей, а второй — времена их ухода (i-й элемент соответствует i-му зрителю).

Программа должна определить максимальное количество зрителей, одновременно находящихся в зале, и временные отрезки, когда это происходило.

5.26 Взлом зашифрованного файла

Источник: "Песни о Паскале" 60-Б

Контрразведка перехватила несколько зашифрованных файлов, подозревая, что это тексты написанных на Паскале вирусов. Позвали Шерлока Ивановича Холмского в надежде, что тот расшифрует их. Шерлок Иванович предположил, что шифровали методом Юлия Цезаря. Нужен ключ! После недолгих раздумий Шерлок Иванович создал программу для подбора ключей к таким текстам. Повторите еще один «подвиг контрразведчика», или слабо? Подсказка: в таких файлах после расшифровки обязательно встречаются ключевые слова **BEGIN** и **END** – воспользуйтесь этим фактом.

5.27 Система выборщиков

Источник: —

В одной очень-очень демократической стране существует очень-очень архаичная система голосования, доставшаяся ей со времён воловьих упряжек. На выборах отдельные деревни сначала голосуют независимо друг от друга за одного из двух кандидатов в президенты, а затем шлют своих представителей — выборщиков — в столицу с тем, чтобы они отдали голоса тому кандидату, за которого проголосовало большинство в их деревне. При этом не учитывается населённость деревень, поэтому президентом может оказаться кандидат, за которого проголосовало меньшинство избирателей страны.

Входной файл содержит строки чисел (по одной строке для каждой деревни) где 1 — означает голос за первого кандидата, 2 — за второго. Ваша программа должна напечатать два числа (1 или 2) соответственно итогам голосования:

- через систему выборщиков;
- по общепринятой в мире пропорциональной системе.

6 Главы 41–43 Сортировка и поиск в массиве

Теоретическая база

- Методы сортировки: «пузырьковый» и «быстрый»
- Линейный поиск в массиве
- Двоичный поиск в массиве

6.1 Ускорение алгоритмов сортировки

Источник: --

Усовершенствуйте процедуры сортировки (пузырьковую, выбором и быструю) исходя из следующих соображений. Если при очередном проходе массива (или его части) не выполнено ни одной перестановки элементов, то массив уже отсортирован и работу надо завершить.

Исследуйте усовершенствованные процедуры на массиве из 10000 элементов и сравните с исходными процедурами (см. главу 43 в книге «Песни о Паскале»).

6.2 Гарантировано несортированный массив

Источник: 4

Напишите процедуру для формирования заведомо несортированного массива.

П о д с к а з к а . Если заполнить массив случайно, есть вероятность, что он окажется отсортированным. Для гарантии неупорядоченности отсортируйте массив и поменяйте попарно соседние элементы.

Контрольный пример:

Входные данные	Результат
1 2 3 4 5 6 7 8 9 10	2 1 4 3 6 5 8 7 10 9

6.3 Ближайшие по росту

Источник: 1

Несортированный массив содержит числа — рост учеников класса. Найдите двух учеников, рост которых либо совпадает, либо отличается в наименьшей степени (а если таких пар несколько, то любую из них).

6.4 Баскетбольные команды

Источник: --

Несортированный массив чётной длины содержит числа — рост учеников класса. Из этих учеников надо справедливо составить две баскетбольные так, чтобы средний рост учеников в них отличался в наименьшей степени.

П о д с к а з к а . Надо отсортировать массив и отобрать: чётные индексы — в одну команду, а нечётные — в другую.

6.5 Новичок

Источник: 1

На физкультуре школьники привыкли строиться в порядке убывания роста, чётко помня, кто за кем следует. Однажды в классе появился новичок, и стал, куда попало, нарушив порядок строя. Дан массив чисел, все элементы которого, за исключением одного, отсортированы по убыванию. Надо найти индекс нарушителя и индекс того ученика, с кем ему надо поменяться местами для восстановления порядка.

6.6 Выбор краски

Источник: 1

Магазин торгует банками с краской, цвета которых обозначим числами от 1 до **M**. Несортированный массив размера **N** содержит перечень имеющихся в наличии банок (то есть, их цвета). Причём размер массива **N** намного превышает количество красок **M**. Для окраски забора папе Карло годится любой цвет, лишь бы количество такой краски было побольше. Помогите папе Карло выбрать подходящий вариант, — надо найти краску такого цвета, наличие которой в магазине максимально.

6.7 Судейство

Источник: --

Соревнование по художественным прыжкам в сторону оценивают 10 судей, выставляющих оценки по 10-балльной шкале. При этом в итоговую оценку идут лишь 8 из 10, а две крайние оценки — наибольшая и наименьшая — отбрасываются.

Оценки двадцати спортсменов содержатся в файле из 20 строк (каждая строка — по 10 оценок). Напечатайте номера спортсменов в порядке убывания набранных ими баллов.

6.8 Формирование сборной

Источник: 1

В небольшой стране всего два футбольных клуба по 20 игроков в каждой. В ходе внутреннего чемпионата тренер сборной формировал рейтинги каждого игрока, а когда настало время составлять сборную, отобрал в неё 20 сильнейших.

Даны два несортированных массива **A** и **B**, содержащие рейтинги игроков (заполните их случайно), необходимо сформировать массив **C** с рейтингами двадцати сильнейших игроков.

6.9 Социальное неравенство

Источник: --

Массив чисел размером **N** содержит в произвольном порядке годовые доходы граждан некоторой страны. Фактором социальной стабильности считается коэффициент, равный отношению суммы доходов 20% богатейших граждан к сумме доходов 20% беднейших. Определите обе суммы и вычислите коэффициент социального неравенства. Напомню, что 20% составляют 1/5 часть граждан.

6.10 Выбор товаров

Источник: --

Входной файл содержит следующие исходные данные (целые числа). В первой строке указана сумма, которую можно потратить на покупку товаров. В следующей строке содержится **N** чисел, показывающих наличие некоторой номенклатуры товаров в магазине (ноль, если товара нет), а в третьей строке из **N** чисел указаны цены соответствующих товаров. Четвёртая строка содержит **N** чисел, показывающих, сколько каждого из товаров требуется купить (ноль для ненужных товаров). Значение **N** известно заранее.

При достаточных средствах и наличии товаров покупатель мог бы приобрести всё нужное. Но если средств или товаров не хватает, он купит лишь часть необходимого. Путь программа подыщет любой из возможных вариантов покупки, при котором потратится максимально возможная часть отпущенные средств (в идеале — все деньги). Ответом должна быть строка чисел с количеством каждого купленного товара и сумма их стоимости.

6.11 Индексирование массива

Источник: 1

Дан числовой несортированный массив **A**. Необходимо создать массив **B** того же размера, содержащий индексы чисел из массива **A** в порядке возрастания этих чисел. То есть первый элемент **B** — это индекс наименьшего числа в **A**, а последний — индекс наибольшего.

Контрольный пример:

Входные данные	Результат
3 5 7 9 8 6 5	1 2 7 6 3 5 4

6.12 Сортировка чётных элементов массива

Источник: 4

Напишите процедуру для сортировки только чётных элементов массива (позиции нечётных не изменяются).

П о д с к а з к а . Воспользуйтесь вспомогательным массивом, в который занесите индексы чётных элементов.

Контрольный пример:

Входные данные	Результат
97 93 62 88 44 80 22 1 30 62 77 35 60 9 17 14 28 18 82 7	97 93 14 18 22 28 30 1 44 60 77 35 62 9 17 62 80 82 88 7

6.13 Соединение сортированных файлов

Источник: 1

На двух путях стоят отсортированные в порядке возрастания своих номеров вагоны, все номера уникальны, то есть, не повторяются. Надо собрать их в единый состав в порядке возрастания номеров.

Исходные данные содержатся в двух текстовых файлах: это числа, упорядоченные по возрастанию. Написать процедуру и программу для объединения их в третий файл с тем же порядком нумерации — по возрастанию.

6.14 Кодовый замок

Источник: "Песни о Паскале" 42-Д

Папа Карло опасался Буратино, и прятал спички в сейфе. Код замка из четырех цифр он доверил лишь своему приятелю – честному малому Джузеппе, который не поддавался ни на какие уговоры деревянного мальчишки. Тогда тот пустился на хитрость. Ладно, – предложил Буратино, – не можешь открыть мне код, – не надо. Давай тогда в игру сыграем: я буду задавать вопросы, а ты отвечай только «да» или «нет». Первый вопрос был таким: код замка больше 5000? Через несколько минут Буратино уже рылся в папином сейфе. Сделайте программу для быстрого угадывания числа методом Буратино. Роль Буратино (угадывающего) должен исполнять компьютер.

6.15 Сортировка дынь и арбузов

Источник: "Песни о Паскале" 43-Г

В одном ряду вперемежку лежат дыни и арбузы. Можно ли отсортировать их за один проход ряда так, чтобы в начале оказались все дыни, а в конце ряда – все арбузы? Напишите такую программу, обозначив арбузы единицами, а дыни – нулями.

6.16 Рейтинговое голосование

Источник: "Песни о Паскале" 60-В

Рейтинговое голосование. По избирательному закону Иксляндии каждый избиратель голосует не за одного, а за всех кандидатов, включенных в бюллетень, расставляя их в порядке своего предпочтения. Побеждает кандидат, набравший наименьшую сумму мест (если таковых несколько, то проводят второй тур). Предположим, баллотируются четыре кандидата, а бюллетени содержат следующие предпочтения избирателей:

```
3 4 2 1
2 4 3 1
4 1 3 2
```

Здесь первый кандидат набирает сумму 10, второй – 8, третий – 7, четвертый – 5. Таким образом, побеждает четвертый кандидат в списке.

Количество кандидатов известно и равно пяти. Ваша программа принимает файл, каждая строка которого содержит 5 чисел – данные одного бюллетеня. Надо выдать список победителей голосования (одного или нескольких).

7 Глава 44
Обработка строк

Теоретическая база

- Процедуры и функции обработки строк
- Кодирование символов

7.1 Забой символов

Источник: --

При редактировании текста для удаления предыдущего символа используют клавишу забоя «←» (BackSpace). Пусть дана строка введённых пользователем символов, где забой условно изображён знаком минус «-». Напишите процедуру, принимающую эту строку по ссылке **VAR**, и возвращающую отредактированную строку, не содержащую символов забоя и удаляемых ими символов.

Контрольный пример:

Входные данные	Результат
abcd--12345-	ab1234

7.2 «Прополка» строки

Источник: 1

Напишите процедуру, принимающую строку по ссылке **VAR**, и возвращающую «прополотую» строку. Суть «прополки» — в удалении части символов так, чтобы оставшиеся символы следовали в неубывающем порядке (в смысле их кодов).

Контрольный пример:

Входные данные	Результат
aBa1bEc6c5dHef	aabccdef

7.3 Подсчёт подстроки

Источник: 1

Функция принимает две строки (строку **S1** и подстроку **S2**) и возвращает число, показывающее, сколько раз подстрока **S2** встречается в строке **S1**.

Контрольный пример:

Входные данные	Результат
'12222345', '22'	2
'12222345', '222'	1

7.4 Набор букв

Источник: "Песни о Паскале" 44-И

Напишите булеву функцию, определяющую, можно ли из букв первого слова составить второе (например, «клавиша» и «вилка» – **TRUE**). Учитывается только набор букв, а не их количество.

7.5 Максимальная сумма кодов

Источник: "Песни о Паскале" 44-К

Дана строка, содержащая не менее трёх символов. Найти в ней три стоящих подряд символа, сумма кодов которых максимальна.

7.6 Сортировка слов по алфавиту

Источник: 4

Дано предложение (строка), состоящее из нескольких слов. Надо напечатать отдельные слова в алфавитном порядке.

Контрольный пример:

Входные данные	Результат
Жили-были дед да баба	баба да дед Жили-были

7.7 Сортировка слов по длине

Источник: 4

Дано предложение (строка), состоящее из нескольких слов. Надо напечатать отдельные слова в порядке возрастания их длины.

Контрольный пример:

Входные данные	Результат
Жили-были дед да баба	да дед баба Жили-были

7.8 Удаление крайних пробелов

Источник: --

Написать процедуру, удаляющую из переданной в неё по ссылке **VAR** строки лидирующие и концевые пробелы, если они там есть.

7.9 Выравнивание текста

Источник: --

Современные текстовые процессоры могут выравнивать текст четырьмя способами:

- по левому краю (лидирующие и концевые пробелы удаляются);
- по правому краю (добавляются лидирующие пробелы);
- по центру (добавляются лидирующие пробелы);
- по ширине (лидирующие и концевые пробелы удаляются, но добавляются между словами).

Напишите процедуры (или одну процедуру с параметром), форматирующую строку указанными выше способами при условии, что ширина страницы составляет 80 колонок.

Напишите программу, форматирующую текстовый файл этими способами.

7.10 Замена табуляторов пробелами

Источник: --

Невидимый управляющий символ горизонтальной табуляции с кодом 9 служит для выравнивания колонок текста в определённые позиции. Обычно выравнивание идёт по границе, кратной восьми: 8, 16, 24 и т.д. Например, следующая исходная строка с двумя табуляциями (здесь пробелы условно изображены точкой, а табуляция — решёткой #):

```
123456789#abcdefghikl#ABC
```

будет изображена в тексте так:

```
123456789......abcdefghikl.....ABC
```

Но не все редакторы воспринимают символ табулятора. Напишите процедуру для замены табуляторов нужным количеством пробелов.

7.11 Проверка идентификатора

Источник: 1

Написать булеву функцию, принимающую строку по ссылке **CONST**, и возвращающую **TRUE**, если строка является правильным идентификатором языка Паскаль.

7.12 Проверка электронного адреса

Источник: --

Написать булеву функцию, принимающую строку по ссылке **CONST**, и возвращающую **TRUE**, если строка является правильным почтовым адресом в рунете.

Контрольный пример:

Входные данные	Результат
baba.yaga@zmey.ru	true
sobaka@@layet	false

7.13 Разбор полного имени файла

Источник: 1

Полное имя файла включает в себя следующие элементы:

- имя диска (например, C:);
- каталог (например, \Windows\System\);
- собственно имя;
- тип файла (расширение).

Процедура принимает ссылку **CONST** на правильное имя файла и возвращает по отдельности четыре названных выше элемента

Контрольный пример:

Входные данные	Результат
c:\Windows\System32\Sort.exe	c: \Windows\System32\ Sort .exe

7.14 Представление целого числа

Источник: --

Строка правильно изображает целое число, если:

- начинается либо с цифры, либо со знаков «+» или «–»;
- если первый символ не цифра, то в последующих символах содержится, хотя бы одна цифра, и ничего, кроме цифр.

Напишите функцию, принимающую строку по ссылке **CONST**, и возвращающую **TRUE**, если строка правильно изображает целое число.

7.15 Представление числа с плавающей точкой

Источник: --

Строка правильно изображает число с плавающей точкой, если:

- начинается либо с цифры, либо со знаков «+» или «–»;
- целая часть числа содержит не менее одной цифры;
- дробная часть либо отсутствует, либо содержит не менее одной цифры, и отделяется от целой части одной точкой;
- не содержит иных символов, кроме упомянутых выше.

Напишите функцию, принимающую строку по ссылке **CONST**, и возвращающую **TRUE**, если строка правильно изображает число с плавающей точкой.

7.16 Скобки трёх уровней

Источник: 1

Дана строка, содержащая разные символы и скобки трех уровней: «()» — внутренние, «[]» — средние, «{} — внешние». Написать функцию, возвращающую **TRUE**, если скобки расставлены без ошибок (прочие символы строки игнорируем).

Контрольный пример:

Входные данные	Результат
m*{2*[3*(a+b)+c]+5}+1	true
m*{2*[3*(a+b)+c}+5]+1	false

7.17 Простой калькулятор

Источник: 1

Строка содержит последовательность из цифр и знаков «+» и «-», первый и последний её символы — цифры. Необходимо вычислить это арифметическое выражение, выполняя действия слева направо.

Контрольный пример:

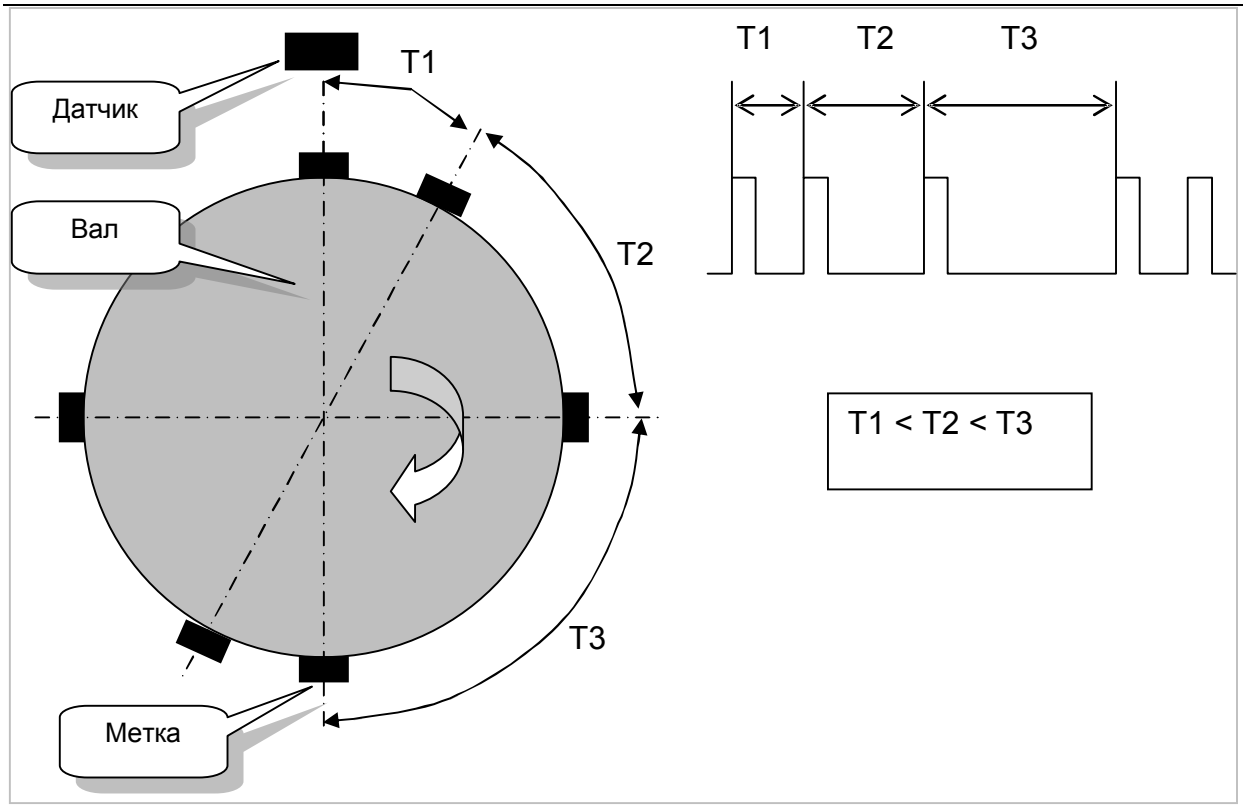
Входные данные	Результат
45-20+1	26
38	38

7.18 Датчик направления (1)

Источник: --

В системах управления электрическими машинами применяют датчики, определяющие скорость и направление их вращения, рассмотрим один из вариантов такого датчика.

Датчик реагирует на проходящие мимо него метки, закреплённые на валу, при этом формирует короткие импульсы, и передаёт их в микропроцессор (см. следующий ниже рисунок).



Метки, количество которых кратно трём (3, 6 и т.д.) распределены по окружности так, что при равномерном вращении (или при небольших ускорениях и торможениях) формируют временные интервалы между импульсами неодинаковой длительности. При этом всегда соблюдается соотношение длительностей:

T1 < T2 < T3

Для простоты положим, что на вход программы поступают не числа, а символы «1», «2» и «3» (то есть, строка из этих символов), наша задача — определить направление вращения.

Итак, дана строка длиной **N** символов, надо сформировать и записать в файл **N-2** чисел, соответствующих направлению вращения так: -1 — по часовой стрелке, +1 — против часовой, и 0 когда направление неизвестно (меняет знак).

Контрольный пример:

Входная строка	Результат
231231231233213213213211	+1 +1 +1 +1 +1 +1 +1 +1 +1 0 0 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0

7.19 Датчик направления (2)

Источник: --

Входной файл содержит последовательность из **N** (**N>2**) натуральных чисел — это времена между импульсами датчика вращения, рассмотренного в предыдущей задаче.

Надо сформировать выходной файл из **N-2** чисел, соответствующих направлению вращения: **TRUE** — по часовой стрелке, **FALSE** — против часовой. Когда направление меняет знак, разрешено формировать любое из двух значений.

Подсказка. Можно анализировать три следующих подряд числа, заменяя их разности к числам +1 и -1.

Контрольный пример:

Входной файл	Результат (для наглядности добавлены исходные данные)
100 200 300 120 220 330 225 125 335 210 110 290	100 200 300 TRUE 200 300 120 TRUE 300 120 220 TRUE 120 220 330 TRUE 220 330 225 FALSE 330 225 125 FALSE 225 125 335 FALSE 125 335 210 FALSE 335 210 110 FALSE 210 110 290 FALSE

7.20 Расстояние Хэмминга (1)

Источник: --

Расстояние Хэмминга характеризует различие между двумя строками одинаковой длины. Оно определяется по количеству несовпадающих символов, стоящих в одинаковых позициях (см. контрольный пример). Напишите функцию, принимающую две строки одинаковой длины, и возвращающую расстояние Хэмминга между ними.

Контрольный пример:

Входные данные	Результат
«МАМА», «ПАПА»	2
«ПАПА», «РЫБА»	3
«РЫБКА», «ОКУНЬ»	5

7.21 Расстояние Хэмминга (2)

Источник: --

Расширим понятие «расстояние Хэмминга» на строки разной длины. Определим его по количеству несовпадающих символов, стоящих в одинаковых позициях слева направо по той строке, что короче, а затем добавим к этому разность длин двух строк.

Напишите функцию, принимающую две строки, и возвращающую расстояние Хэмминга между ними.

Контрольный пример:

Входные данные	Результат
«МАМА», «МАМУЛЯ»	3
«ПАПА», «РЫБАК»	4
«РЫБА», «ОКУНЬ»	5

7.22 Вывод множества

Источник: "Песни о Паскале" 52-Г

Распечатывая числовое множество, мы выводили все его элементы по одному, не заботясь об экономии бумаги или места на экране. Напишите «экономную» процедуру печати множества, которая должна учитывать подряд идущие диапазоны чисел. Вот примеры желаемой распечатки:

```
1,5..255
0..200,210..255
0..255
2,5,7,10..20,30..40
```

8 Глава 45
Очереди и стеки на основе строк

Теоретическая база

- Понятия очереди и стека
- Операции с очередями и стеками
- Реализация очередей и стеков посредством строк и массивов

8.1 Разделение символов

Источник: --

Дана строка символов, состоящая из латинских букв и цифр. Составить другую строку, где те же символы следуют так: сначала латинские прописные, затем цифры, а потом латинские строчные (внутри групп порядок прежний). Дополнительное требование: исходная строка должна обрабатываться за один проход.

8.2 Модель записи в танцевальный кружок

Источник: "Песни о Паскале" 45

Дана строка символов, состоящая из прописных и строчных латинских букв. Прописными обозначены мальчики, а строчными — девочки Составить другую строку, где те же символы следуют попарно: мальчик+девочка, мальчик+девочка и т.д., — так составляются пары для танцевального кружка в порядке следования символов в исходной строке. Строка — это очередь, поэтому должна быть обработана за один проход.

8.3 Модель сортировочной горки

Источник: "Песни о Паскале" 45

Дан файл из нескольких строк, каждая из которых содержит три сорта символов: 1) цифры, 2) прописные и 3) строчные буквы. Считаем, что символы — это вагоны, а строки — это составы, где воображаемый локомотив прицеплен к началу строки. Необходимо пропустить все составы через сортировочную горку так, чтобы получить в итоге три других состава, три строки, каждая из которых содержит символы одного сорта: цифры, прописные буквы, строчные буквы.

8.4 Бюрократическая очередь

Источник: "Песни о Паскале" 46-Г

Жители райцентра Бюрократовка дневали и ночевали в очереди за справками. Все потому, что там применяли механический текстовый файл – огромную скрипучую книгу, которая листалась лишь в одном направлении – от начала к концу файла. Если первая буква фамилии очередного посетителя следовала по алфавиту далее, чем у предыдущего, то чиновник продолжал листать страницы с текущей позиции, а иначе открывал на первой и листал с начала. Переход от одной буквы алфавита к другой и возврат в начало занимали один час. Так, если буквы следовали в порядке «АБВ», то на выдачу справок тратилось три часа, а если в обратном порядке – «ВБА», – то шесть часов (3+2+1). Если же первые буквы фамилий совпадали, то книгу все равно листали заново, поэтому на «ББВ» тратилось шесть часов. Создайте функцию, принимающую «очередь посетителей» – строку из прописных латинских букв – и возвращающую время, необходимое для выдачи всех справок.

8.5 Усовершенствованная очередь

Источник: "Песни о Паскале" 46-Д

Томясь в бюрократической очереди, свинопас Гришка нашел способ ускорить выдачу справок путем частичного упорядочения очереди (см. предыдущую задачу). Создайте функцию, возвращающую такую частично упорядоченную строку (воспользуйтесь множеством символов). Напишите программу для сравнения времен по условиям этой и предыдущей задачи.

8.6 Музыкальный проигрыватель

Источник: "Песни о Паскале" 52-В

Программист Ник обожал музыку. Но компьютерный музыкальный проигрыватель раздражал его, поскольку при случайном выборе мелодий повторял одни песни, напрочь «забывая» о других. Предположим, в списке 10 песен, но звучали почему-то только три из них: 3, 6, 5, 6, 3, 6, 5 и т.д.

Ник создал «справедливый» проигрыватель, который выбирал мелодии иначе. Все песни состояли в одном из двух списков: «белом» или «черном». Изначально все они были в «белом» списке, и очередная мелодия выбиралась из него случайно, а после проигрывания ставилась в конец «черного». Если в этот момент в «черном» списке состояла половина мелодий, то первая мелодия из «черного» списка возвращалась в «белый». Затем снова случайно выбиралась мелодия из «белого» списка. Так гарантировалось отсутствие повторов ранее проигранных песен в течение достаточно длительного времени. Создайте программу, генерирующую случайные числа (мелодии) в диапазоне от 1 до **N** представленным выше методом. Значение **N** не превышает 255.

8.7 «Глупый» винчестер

Источник: "Песни о Паскале" 53-Г

Рассмотрим очень упрощенную модель винчестера, быстродействие которого в основном определяется частотой вращения диска и скоростью перемещения головки чтения-записи. Время одного оборота диска примем за единицу - квант. За это время головка полностью читает или записывает одну дорожку. Количество дорожек на диске – 256, а нумерация идет с нуля (0...255). Время для перемещения головки на соседнюю дорожку тоже примем равным одному кванту.

Винчестером управляет контроллер, работающий несравнимо быстрее механических узлов - диска и головки. Поэтому издержками времени на его работу пренебрежем. Через известный интервал времени контроллер просматривает входную очередь, содержащую запросы на чтение или запись дорожек. Эта очередь формируется всеми запущенными программами. Мы заменим её текстовым файлом, каждая строка которого содержит по несколько чисел в диапазоне от 0 до 255 – это номера запрашиваемых дорожек. Пустая строка говорит об отсутствии запросов в текущий момент времени. Для первой строки файла сделаем исключение, поместив там лишь одно число - период просмотра этой очереди.

Контроллер «рулит» так. Прочитав список запросов (очередную строку файла), он перемещает их в свою внутреннюю очередь и далее обрабатывает её в том же порядке: смещает головку в нужную позицию и выполняет чтение-запись. По ходу этой работы он следит за таймаутом, и, по истечении оного, читает следующую порцию входной очереди (строку файла). Ваша программа должна подсчитать общее время обработки запросов для набора данных из входного файла.

8.8 «Умный» винчестер

Источник: "Песни о Паскале" 54-Д

В предыдущей задаче была представлена модель «глупого» винчестера. «Умный» винчестер отличается организацией внутренней очереди и челночным движением головки. Она следует попеременно то от внутренней дорожки к внешней, то обратно, попутно выполняя все накопившиеся в очереди запросы. Направление движения переключается тогда, когда в текущем направлении не остается запросов, поэтому головка редко достигает крайних дорожек.

Ваша программа должна подсчитать общее время обработки запросов «умным» контроллером для набора данных из входного файла, составленного по условию предыдущей задачи. Создайте несколько наборов таких данных и сравните время их обработки двумя типами контроллеров.

П о д с к а з к а . Для организации внутренней очереди контроллера примените массив чисел (счётчиков). Каждый счётчик будет хранить количество запросов для своей дорожки. При постановке запроса в очередь счётчик наращивается, а при извлечении (обработке) уменьшается.

9 Глава 46
Огромные числа

Теоретическая база

- Реализация сверхбольших чисел через массивы и строки
- Выполнение арифметических действий «в столбик»

9.1 Сложение и вычитание огромных чисел

Источник: --

Постройте сверхбольшие числа на основе строковых переменных, напишите две функции: для сложения и вычитания таких чисел.

9.2 Умножение числа на цифру

Источник: --

Напишите функции:

- для умножения огромного числа на 10 (это очень просто!);
- для умножения огромного числа на однозначное число от 0 до 9.

Оптимизируйте два частных случая: умножение на ноль и единицу.

Контрольный пример:

Входные данные	Результат
32859305092145 * 5	164296525460725

9.3 Перемножение огромных чисел

Источник: --

Напишите функцию, принимающую две ссылки на огромные числа и возвращающую их произведение.

П о д с к а з к а . Примените принцип умножения «в столбик», то есть, последовательно умножайте одно число на цифры другого, сдвигайте эти произведения влево, приписывая нужное количество нулей, и накапливайте сумму этих частичных произведений. Воспользуйтесь функциями для суммирования огромных чисел и умножения их на цифру.

Контрольный пример:

Входные данные	Результат
123456789 * 1234567	152415675233024136 3

10 Глава 47
Системы счисления

Теоретическая база

- Системы счисления: десятичная, двоичная, 16-ричная
- Перевод чисел между системами счисления

10.1 Определение длины числа

Источник: --

Пользователь вводит целое положительное число (не строку!), а программа печатает количество содержащихся в нём десятичных цифр. Напишите функцию, определяющую количество цифр в числе.

Контрольный пример:

Входные данные	Результат
12	2
3000	4

10.2 Поиск чисел по цифрам

Источник: --

Пользователь вводит число, а программа печатает все числа в интервале от 0 до 999, составленные из тех же цифр, что составляют введённое число.

Контрольный пример:

Входные данные	Результат
12	12 21 112 121 212 221
3	3 33 333

10.3 Цифровой корень числа

Источник: 4

Цифровой корень многозначного числа определяется так: все его цифры складываются, затем, если сумма содержит два и более знака, её цифры опять складываются и т.д., пока не получится однозначное число. Например: 9567 → 27 → 9.

Напишите функцию, принимающую длинное целое число (**Longint**), и возвращающую его цифровой корень.

Контрольный пример:

Входные данные	Результат
7	7
37	1
9567	9

10.4 Правильная дробь

Источник: 12

Дробь **N/M**, у которой числитель **N** меньше знаменателя **M**, называется правильной (такая дробь меньше единицы). Напишите функцию, возвращающую строку, которая выражает правильную дробь в виде десятичной дроби с заданным количеством значащих цифр. Функция принимает три целых числа: числитель, знаменатель и количество требуемых цифр.

Контрольный пример:

Входные данные	Результат
5, 7, 32	0,71428571428571428571428571428571

10.5 Сумма цифр

Источник: "Песни о Паскале" 47-Б

Программист Ник наловчился запоминать сумму цифр в номерах всех автомобилей, попадавших ему на глаза. Однажды он стал свидетелем происшествия, виновник которого скрылся. Ник сообщил полицейским только сумму цифр в номере нарушителя (сам номер Ник не помнил). Помогите полиции, и напишите программу, распечатывающую все трехзначные номера (от 1 до 999), сумма цифр которых равна **N** (значение **N** вводит пользователь).

10.6 Кассовый аппарат

Источник: "Песни о Паскале" 47-З

В заморской стране обращались денежные купюры достоинством в 1, 2, 5, 10 и 25 пиастров. Напишите программу для кассового аппарата, определяющую наименьший набор купюр, необходимый для выдачи сдачи на указанную сумму. Например, для сдачи 33 пиастров программа напечатает: 25 + 5 + 2 + 1.

10.7 Шифрование файла

Источник: "Песни о Паскале" 47-И

Программа шифрования текстового файла заменяет каждый символ двумя шестнадцатеричными цифрами его кода. Например, три символа '**405**' заменяются на шесть символов '**343035**'. Символы разбивки строк не затрагиваются. Напишите программу для зашифровки и расшифровки файла по этой системе.

10.8 Цифры-делители

Источник: "Песни о Паскале" 47-Л

Напечатайте все числа, не превышающие 1000, такие, что делятся без остатка на каждую из своих цифр. Например: 24, 36, 184, 612. Определите количество таких чисел.

11 Глава 48
Железная логика

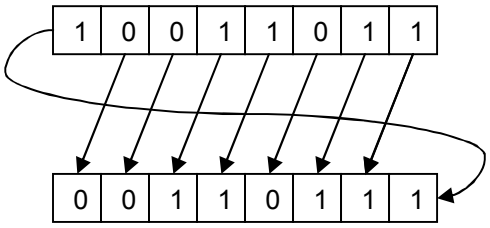
Теоретическая база

- Представление чисел в двоичной системе
- Побитовые операции с числами
- Сдвиги влево и вправо

11.1 Циклический сдвиг

Источник: "Песни о Паскале" 48-Б

Наряду со сдвигами регистра, в процессорах заложены операции **циклического** (кругового) сдвига. При таком сдвиге (см. рисунок ниже) выдвигаемый бит не теряется, а попадает соответственно в младший бит (при сдвиге влево) или в старший бит (при сдвиге вправо).



Циклический сдвиг влево

В Паскале операции циклического сдвига не предусмотрены. Напишите функции для циклического сдвига слова влево и вправо. Подсказка: перед сдвигом каждого бита надо проверить состояние теряемого бита, а затем восстановить его в младшем или старшем разряде.

11.2 Ёлочная гирлянда (1)

Источник: --

Ёлочная гирлянда составлена из 16 ламп и управляется микропроцессором. Каждая лампа во включенном состоянии потребляет ток 1 ампер. Микропроцессор зажигает лампы в соответствии с двоичным кодом последовательно наращиваемых чисел: 0, 1, 2, 3 и т.д. до 65535. Единицы в коде соответствуют включенным лампам, а нули — погашенным.

Пусть дано число **N**, определите, какой ток будут потреблять лампы в этом случае (токи включенных ламп суммируются). Напишите соответствующую функцию и программу.

Контрольный пример:

Входные данные	Результат
0	0
32767	15
32768	1

11.3 Ёлочная гирлянда (2)

Источник: --

Управляемая микропроцессором ёлочная гирлянда состоит из 16 ламп, каждая из которых во включенном состоянии потребляет некоторый ток (см. 11.1). Однако, в связи с ограниченной мощностью электросети, разрешено зажигать одновременно не более **L** ламп (**L** задаётся от 1 до 15). Пользователь вводит значение **L**, а программа выводит:

- общее количество допустимых кодовых комбинаций из **L** ламп;
- допустимые значения кодов гирлянды (т.е. десятичные числа, соответствующие этим кодам).

11.4 Шифрование

Источник: --

Логическая операция «Исключающее ИЛИ» (**XOR**), применённая к числам, опрокидывает те биты в первом операнде, на которые указывают единицы во втором операнде. Она интересна следующим свойством:

```
(A xor B) xor B = A
```

То есть, повторное её применение возвращает исходное число (в данном случае **A**). Это свойство удобно при шифровании данных. Напишите функцию для зашифровки и расшифровки строки, функция принимает ключ шифрования **K** и строку **S** по ссылке **VAR**. Создайте программу для проверки функции.

11.5 Расстояние Хэмминга (3)

Источник: --

Расстояние Хэмминга характеризует различие между двумя строками одинаковой длины. Оно равно количеству несовпадающих символов, расположенных в одинаковых позициях. Напишите функцию, определяющую расстояние Хэмминга между двумя числами в их двоичном представлении.

П о д с к а з к а . Переводить число в строку не требуется, выполните с числами операцию «Исключающее ИЛИ» (**XOR**), а в полученном результате подсчитайте количество единичных битов.

Контрольный пример:

Входные данные	Двоичный код (на примере 4-х младших битов)	Результат
7 и 5	0111 и 0101	1
7 и 8	0111 и 1000	4
8 и 4	1000 и 0100	2

11.6 Расстояние Хэмминга (4)

Источник: --

Расстояние Хэмминга для чисел рассмотрено в задаче 11.5. Напечатайте таблицу взаимных расстояний Хэмминга для чисел от **0** до **N**, таблица должна иметь **N+2** строк и столбцов, например, для **N**=3:

	0	**1**	**2**	**3**
0	0	1	1	2
1	1	0	2	1
2	1	2	0	1
3	2	1	1	0

11.7 Код Грея

Источник: --

Создателя программы к ёлочной гирлянде (см. 11.1) попросили, чтобы количество включенных ламп в двух следующих по времени состояниях гирлянды отличалось на единицу (это снижало уровень порождаемых сетевых помех). Другими словами, расстояние Хэмминга между этими двоичными кодами должно составлять единицу (см. 11.4). Обычный двоичный код этому требованию не отвечает; так, например, при смене числового кода с 31 на 32 гаснут четыре лампы, а включается лишь одна (расстояние Хемминга здесь равно трём).

Порывшись в киберпространстве, программист наткнулся на код Грея. Последовательно нарастающие числа в коде Грея по количеству единиц отличаются ровно на единицу. Вот для сравнения коды нескольких чисел.

Число	Двоичный код	Код Грея
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

Напишите функцию для преобразования числа в код Грея и напечатайте таблицу: слева исходные числа от 0 до 32, а справа те же числа в коде Грея (в десятичном и двоичном представлении).

П о д с к а з к а . Код Грея получается путём выполнения операции «Исключающее ИЛИ» (**XOR)** числа с им самим, сдвинутым на один разряд вправо. При сдвиге вправо старший разряд заполняется нулём, а младший теряется.

11.8 Код Джонсона («электричка»)

Источник: --

От создателя программы к ёлочной гирлянде (см. 11.1) потребовали, чтобы освещённость в помещении плавно нарастала от минимума до максимума и обратно, а количество включенных ламп в двух последовательных по времени состояниях гирлянды отличалось на единицу (это снижало уровень порождаемых сетевых помех). Другими словами, расстояние Хэмминга между этими двоичными кодами должно составлять единицу (см. 11.4). Обычный двоичный код и код Грея этим требованиям не отвечают.

Программист добился желаемого, формируя код по принципу проходящей вдоль платформы электрички: сначала появляется один вагон, затем два и т.д. В какой-то момент все **N** вагонов оказываются у платформы, а затем уходят в той же последовательности и остаются: **N-1** вагон, **N-2** вагона и т.д. Соответствующий код называется кодом Джонсона, вот несколько его значений на примере 4-х бит:

Число	Код Джонсона
0	0000
1	0001
2	0011
3	0111
4	1111
5	1110
6	1100
7	1000

Напишите программу для формирования последовательных кодов Джонсона для заданного диапазона чисел.

11.9 Контроль по Хэммингу

Источник: --

В системах связи данные передают по байтам, а байты — побитно (оптоволокно, радио). При этом из-за помех возможно искажение отдельных битов (передан ноль, а принята единица, или наоборот). Для обнаружения и устранения таких ошибок, передаваемые байты снабжают дополнительными контрольными битами.

Рассмотрим последовательную передачу некоторого массива байтов (например, строки или файла), содержащего символы с кодами от 0 до 127 (это первая часть кодовой таблицы ASCII, где старший 7-й бит равен нулю). Каждый байт этого кода снабдим одним дополнительным контрольным битом, поместив его в старшей 7-й позиции. Этот бит выберем так, чтобы общее количество единичных битов в передаваемом байте оказалось чётным, — это придумка Хэмминга. Тогда, если в принятом байте количество единичных битов окажется нечётным, мы вправе утверждать, что в ходе передачи байта было искажено нечётное количество битов: один, три, пять или семь (чётное количество искажений мы не обнаружим).

Напишите две функции: одну — для подсчёта битов в байте, и другую — для снабжения байта контрольным битом в старшем 7-м разряде. Напишите программы для кодирования файла и его декодирования, а также для подсчёта количества искажённых байтов (искажения в закодированный файл внесите редактором текста).

11.10 Контрольная сумма строки

Источник: --

В системах передачи данных проверяют не только отдельные байты (см. 11.9), но и блоки данных из многих байтов. Один из способов состоит в подсчёте контрольной суммы блока операцией «Исключающее ИЛИ» (**`XOR`**). Это не обычная сумма, её называют суммой по модулю 2, и для **`n`** байтов она подсчитывается так:

```
CS = B1 xor B2 xor B3 ... xor Bn
```

Сумма **`CS`** тоже является байтом, каждый бит которого зависит от количества единиц в тех же битах контролируемого блока: ноль — если оно чётное, и единица, если нечётное. Контрольный байт подсчитывают перед передачей и добавляют к передаваемому блоку. На приёмной стороне тем же способом вновь подсчитывают контрольную сумму, но уже с учётом контрольного байта. Если результат окажется не нулевым, значит, при передаче было искажено нечётное количество соответствующих битов.

Пусть дана строка символов, напишите функцию для подсчёта её контрольной суммы и процедуру для добавления контрольной суммы в конец строки. Проверьте контрольную сумму вновь полученной строки, равняется ли она нулю?

Примечание. Контроль отдельных байтов (поперечный) и блока байтов в целом (продольный) снижают вероятность ошибок при передаче данных.

11.11 Шифрование перестановкой битов

Источник: --

Для шифрования чисел Штирлиц использовал перестановку битов в двоичном представлении числа в соответствии с некоторой таблицей, известной лишь ему и получателю шифровки. Например, для 8-битовых чисел таблица из 4-х строк могла быть такой (биты нумеруются от 0 до 7):

```
3 1 — переставляются 3-й и 1-й бит
7 2 — переставляются 7-й и 2-й бит
4 0 — переставляются 4-й и 0-й бит
5 6 — переставляются 5-й и 6-й бит
```

То есть, каждый бит числа меняется с каким либо другим битом, но не более одного раза. Таким образом, и зашифровка, и расшифровка выполняются одинаково.

Пусть шифровальная таблица задана в виде файла: четыре строки по два числа в каждой. Введите её в массив (или в два массива), а затем используйте для шифрования и расшифровки однобайтовых чисел.

12 Глава 49
Двумерные и сложные массивы

Теоретическая база

- Двумерные массивы (матрицы)
- Обработка столбцов, строк, диагоналей матрицы
- Массивы множеств

12.1 Построчная распечатка матрицы

Источник: --

Объявите матрицу размером **N=5** строк и **M=4** столбца, заполните матрицу случайными числами в интервале от **0** до **99** и распечатайте её построчно: на экран или в текстовый файл.

Примечание. Эта матрица будет содержать исходные данные для последующих задач.

12.2 Поиск максимума и минимума

Источник: --

Напишите две функции: для поиска минимального и максимального чисел в матрице (см. 12.1). Матрицу передавайте параметром по ссылке **CONST**.

12.3 Поиск максимума в строке

Источник: --

Напечатайте максимальные числа из каждой строки матрицы. Для поиска максимального числа создайте функцию, принимающую два параметра: 1) матрицу по ссылке **CONST** и 2) номер обрабатываемой строки.

12.4 Обнуление нечётных элементов

Источник: --

Напишите процедуру для обнуления элементов матрицы, содержащих нечётные числа. Матрицу передавайте параметром по ссылке **VAR**.

Контрольный пример:

Входные данные	Результат
10 9 11 4	10 0 0 4
23 64 91 84	0 64 0 84
41 86 18 31	0 86 18 0
14 37 32 76	14 0 32 76
64 28 71 8	64 28 0 8

12.5 Сортировка строк

Источник: 1

Результаты сдачи ЕГЭ **M** учениками по **N** предметам представлены матрицей **M×N**. Надо упорядочить учеников (строки матрицы) двумя способами:

- в порядке убывания максимального из **N** баллов;
- в порядке убывания суммы баллов по всем экзаменам.

С о в е т . Учредите функцию поиска максимального элемента в строке.

12.6 Сортировка столбцов

Источник: 1

Результаты сдачи ЕГЭ **M** учениками по **N** предметам представлены матрицей **M×N**. Надо упорядочить предметы (столбцы матрицы) двумя способами:

- в порядке убывания максимального из **M** баллов;
- в порядке убывания суммы баллов по всем ученикам.

С о в е т . Учредите функцию поиска максимального элемента в столбце.

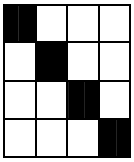
12.7 Распечатка главной и побочной диагоналей

Источник: --

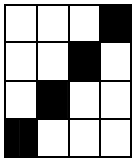
Объявите квадратную матрицу размером **NxN (N=5)**, заполните матрицу случайными числами в интервале от **0** до **99,** а затем распечатайте главную и побочную диагонали.

П р и м е ч а н и е . Главная и побочная диагонали соответствуют следующим клеткам.

Главная диагональ:



Побочная диагональ:



Контрольный пример:

Входные данные	Главная диагональ	Побочная диагональ
10 9 11 4 72	10	72
23 64 91 84 11	64	84
41 86 18 31 6	18	18
14 37 32 76 15	76	37
64 28 71 8 32	32	64

12.8 Заполнение побочной диагонали

Источник: 4

Дана квадратная матрица размером **NxN**, заполните её побочную диагональ по возрастанию снизу вверх так, как показано в контрольном примере (пустые клетки содержат нули).

Контрольный пример:

Входные данные	Побочная диагональ
N = 5	5 4 3 2 1

12.9 Садовая ограда

Источник: "Песни о Паскале" 49-Г

Садовая ограда. Вернувшись с курорта, фермер Лефт обнаружил на своем поле чудесно выросший сад. Для сохранения деревьев он обнес его прямоугольной оградой. Пусть ширина и высота поля заданы константами CX и CY, пустые места обозначены точками, а деревья – звездочками. Засадите поле случайным образом и распечатайте его. Затем найдите левый верхний и правый нижний углы ограды и постройте ограду символом решетки (она должна охватывать деревья, но не выходить за пределы поля). Распечатайте сад с оградой.

12.10 Футбольный чемпионат

Источник: --

В чемпионате, проходящем в один круг, участвуют **N** команд. Матрица **NxN** выше главной диагонали заполнена числами, показывающими, сколько очков взяла i-я команда в матче с k-тым соперником (0, 1 или 3 очка), например:

i / k	1	2	3	4	5	6
1		3	3	0	1	3
2			1	1	1	1
3				3	0	1
4					0	0
5						0
6						

Ваша программа должна заполнить таблицу (случайно или вводом из файла), затем определить количество очков, набранное каждой командой, и распечатать номера команд в порядке убывания набранных очков. Количество команд **N** задать константой.

12.11 Формирование сборной (2)

Источник: 1

В чемпионате страны играют 16 клубов по 20 игроков в каждой. В ходе внутреннего чемпионата тренер сборной формировал рейтинги каждого игрока, а когда настало время создавать сборную, решил отобрать в неё 20 сильнейших.

Дана несортированная матрица размером 16x20, содержащая рейтинги игроков (заполните её случайно), необходимо сформировать массив из 20 элементов с рейтингами сильнейших игроков.

12.12 Матрица «террас»

Источник: 4

Дано число **N>1**, надо построить матрицу «террас» размером **2•N-1** так, как показано в контрольном примере.

П о д с к а з к а . Заполняйте побочные диагонали воображаемых матриц размером **NxN**, сдвигая их вправо и вниз.

Контрольный пример:

Входные данные	Матрица террас
N = 5	5 4 10 3 9 15 2 8 14 20 1 7 13 19 25 6 12 18 24 11 17 23 16 22 21

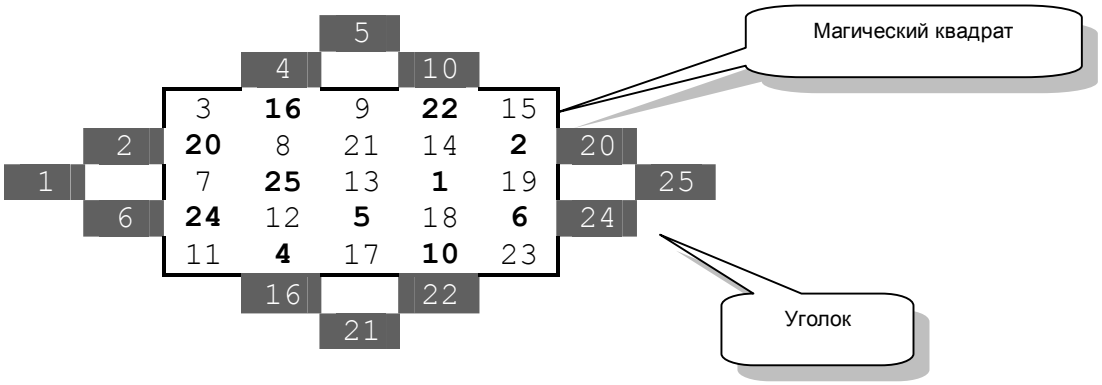
12.13 Магический квадрат

Источник: 4

Магический квадрат — это прямоугольная матрица с числами, сумма которых одинакова для каждой строки, столбца и обеих диагоналей. Для нечётного **N>4** постройте магический квадрат так, как показано в контрольном примере.

П о д с к а з к а . Сначала постройте вспомогательную матрицу террас размером **2•N-1** (см. предыдущую задачу), затем «уголки конверта», выступающие за пределы магического квадрата, сдвиньте внутрь, как показано на следующем рисунке (сдвинутые

числа выделены жирным). Тогда магический квадрат окажется в центре вспомогательной матрицы.



Контрольный пример:

Входные данные	Магический квадрат				
N = 5	3	16	9	22	15
	20	8	21	14	2
	7	25	13	1	19
	24	12	5	18	6
	11	4	17	10	23

12.14 Количество путей в клетку (арифметический квадрат)

Источник: 4

Дано клеточное поле размером **NxM** строк и столбцов. Робот может перемещаться по полю, двигаясь из клетки с координатами [1,1] только слева направо и сверху вниз. Для каждой клетки надо подсчитать количество разных путей, по которым робот может в неё попасть.

Контрольный пример:

Входные данные	Результат						
N=7	1	1	1	1	1	1	1
M=7	1	2	3	4	5	6	7
	1	3	6	10	15	21	28
	1	4	10	20	35	56	84
	1	5	15	35	70	126	210
	1	6	21	56	126	252	462
	1	7	28	84	210	462	924

12.15 Путь с минимальной ценой

Источник: 4

Дано клеточное поле размером **NxM** строк и столбцов. Робот должен переместиться из верхнего левого угла с координатами **[1,1]** в правый нижний угол с координатами **[N,M]** двигаясь только слева направо и сверху вниз. По пути в каждой клетке он тратит энергию, количество которой указано в клетке. Необходимо найти такой оптимальный путь, при котором затраченная энергия (стоимость) будет минимальна.

Входной файл содержит таблицу стоимостей клеток. В выходной файл надо записать изображение оптимального пути посредством точек и решёток, как показано в контрольном примере.

П о д с к а з к а . Сначала постройте матрицу стоимости, исходя из следующих соображений: в клетки верхней строки можно попасть только слева, в клетки левого столбца — только сверху, а в остальные клетки — либо слева, либо сверху (здесь надо выбрать более дешевый вариант). Затем, двигаясь по матрице стоимости от конечной точки к начальной, отметить оптимальный путь каким-либо особым числом (например, нулём).

Контрольный пример:

Входные данные	Результат
4 3 5 9 2 2 3	# # . . . . .
5 5 7 4 5 1 5	. # . . . . .
6 1 1 8 4 3 4	. # . . . . .
8 1 5 8 4 5 8	. # # . . . .
7 7 7 1 3 4 5	. . # # # . .
6 8 6 7 3 3 8	. . . . # # .
7 6 2 9 4 3 7	. . . . . # #

12.16 Расстояние между клетками

Источник: --

В клеточном пространстве можно двигаться лишь влево-вправо и вверх-вниз (срезать углы не позволено).

Пусть даны координаты двух клеток: **x1**, **y1** — первая клетка; **x2**, **y2** — вторая. Здесь **x** — номер строки, **y** — номер столбца. Напишите функцию для вычисления расстояния между клетками (функция принимает четыре числа — координаты клеток).

12.17 Числовой крест

Источник: 4

Дана матрица размером **N** строк на **M** столбцов. Заполнить её числами, симметрично возрастающими от центра матрицы к краям так, как показано в контрольных примерах. Вместо заполнения матриц можно вывести таблицы на экран.

П о д с к а з к а . Числа в клетках — это расстояния от центра матрицы к этим клеткам.

Контрольные примеры:

Входные данные	Результат
7 строк x 7 столбцов	7 6 5 4 5 6 7 6 5 4 3 4 5 6 5 4 3 2 3 4 5 4 3 2 1 2 3 4 5 4 3 2 3 4 5 6 5 4 3 4 5 6 7 6 5 4 5 6 7
8 строк x 8 столбцов	7 6 5 4 4 5 6 7 6 5 4 3 3 4 5 6 5 4 3 2 2 3 4 5 4 3 2 1 1 2 3 4 4 3 2 1 1 2 3 4 5 4 3 2 2 3 4 5 6 5 4 3 3 4 5 6 7 6 5 4 4 5 6 7
7 строк x 8 столбцов	7 6 5 4 4 5 6 7 6 5 4 3 3 4 5 6 5 4 3 2 2 3 4 5 4 3 2 1 1 2 3 4 5 4 3 2 2 3 4 5 6 5 4 3 3 4 5 6 7 6 5 4 4 5 6 7

12.18 Наибольший среди наименьших

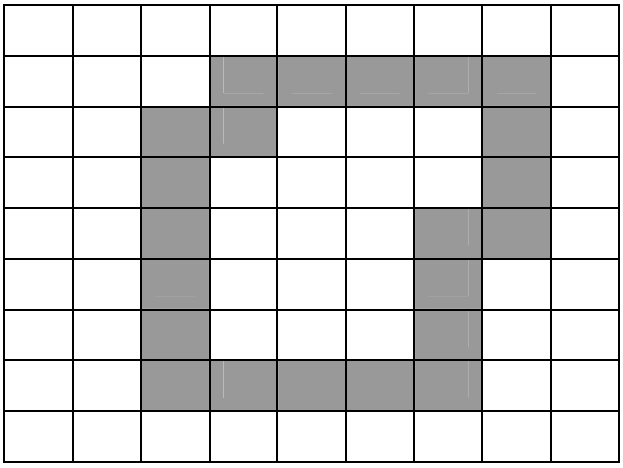
Источник: --

Дан числовой массив размером **MxN** (введите его из файла или заполните случайным образом). Программа должна найти индексы элементов, содержащие числа, одновременно наименьшие в своей строке и наибольшие в своём столбце. Если таких элементов нет, вывести сообщение об этом.

12.19 Замкнутая область

Источник: --

На клеточном поле размером **NxN** изображена замкнутая область (см. рисунок ниже). Создайте входной текстовый файл, изображающий такое поле в виде точек и звёздочек (соответственно белые и серые клетки). Введите файл в двумерный массив, затем выведите этот массив на экран. Пусть программа подсчитает количество белых клеток внутри замкнутой области.



12.20 Заливка замкнутой области

Источник: --

Напишите программу для заливки замкнутой области, рассмотренной в предыдущей задаче (12.18), символом решётки (#). Распечатайте результат на экране.

12.21 Подсчёт несоприкасающихся областей

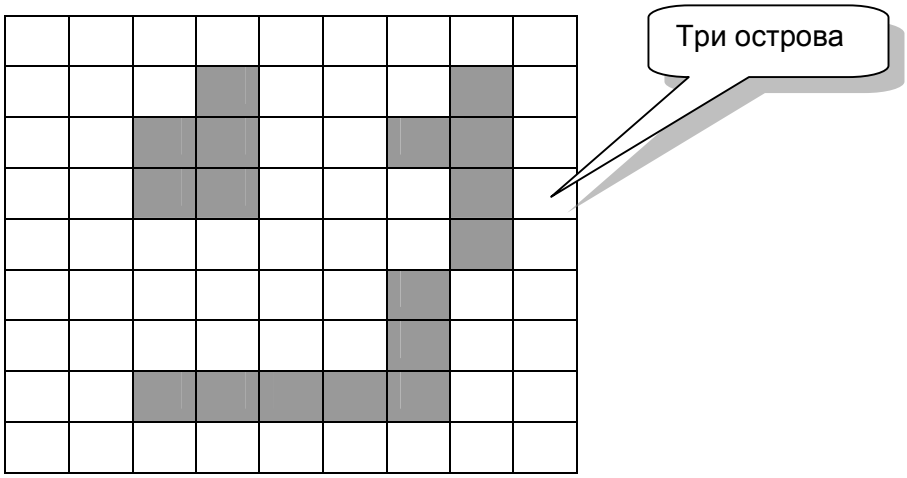
Источник: --

Входной файл (см. задачу 12.18) изображает клеточное поле, часть клеток которого закрашена серым. Пусть программа определит, на сколько несоприкасающихся частей из белых клеток делится это поле. Части не соприкасаются, если их полностью разделяет ограда из серых клеток. То есть, воображаемый робот, двигающийся только вверх-вниз и вправо-влево, не может покинуть огороженную часть.

12.22 Подсчёт островов

Источник: --

Входной файл (см. задачу 12.18) изображает клеточное поле, часть клеток которого закрашена серым, образуя острова (см. рисунок ниже). Подсчитайте количество таких островов с учётом того, что островам разрешено соприкасаться уголками своих клеток.

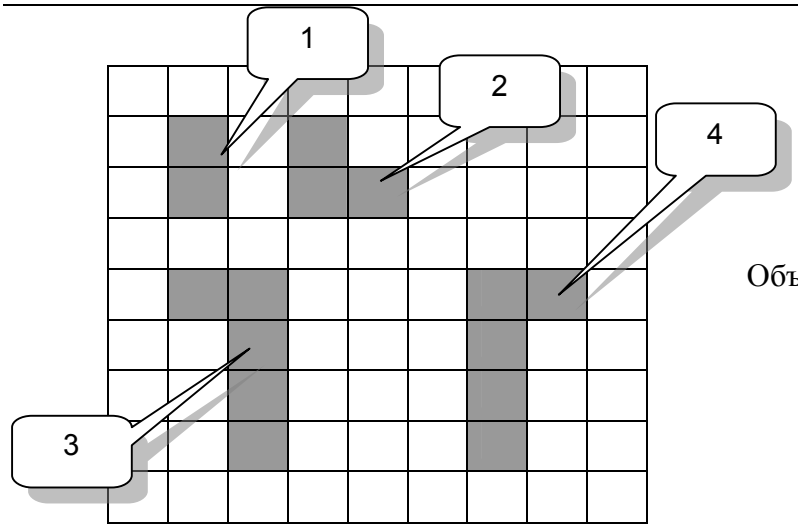


12.23 Космическая разведка (1)

Источник: --

По фотографии из космоса создан входной файл (см. задачу 12.18), который изображает на клеточном поле одну из четырёх фигур — это интересующие разведку объекты (см. рисунок ниже). Пусть функция в вашей программе определит, какой из этих объектов изображён на поле (1-4), или вернёт ноль, если изображение не соответствует ни одному из них.

В н и м а н и е ! Объект может быть повёрнут на любой угол: 90, 180 или 270 градусов.



Объекты и их номера

12.24 Космическая разведка (2)

Источник: --

По фотографии из космоса создан входной файл, который изображает несколько объектов (см. предыдущую задачу 12.23). Написать программу, определяющую все объекты на этой карте (вывести номера обнаруженных объектов: числа от 1 до 4).

12.25 Лабиринт

Источник: --

Входной файл (см. задачу 12.18) изображает клеточное поле, часть клеток которого закрашена серым, изображая стены лабиринта. Левая верхняя и правая нижняя клетки свободны — это начало и конец маршрута, который должна найти ваша программа. Перемещаться из клетки в клетку возможно лишь влево-вправо и вверх-вниз. Пусть программа изобразит символом решётки кратчайший маршрут из одной клетки в другую, если он существует.

12.26 Путь с пересадкой

Источник: --

Маршруты городских автобусов названы заглавными буквами латинского алфавита: **A**, **B**, **C** и т.д., а остановки автобусов обозначены числами от 1 до 255. Количество маршрутов **N** известно заранее (определено константой). Некоторые маршруты могут пересекаться или частично совпадать (когда автобусы посещают одни и те же остановки).

Входной файл содержит следующие данные. В первой строке — два числа, указывающие, откуда и куда едет пассажир (начальную и конечную остановки). Последующие строки — это перечень остановок каждого из городских маршрутов в их алфавитном порядке; здесь каждый маршрут представлен отдельной строкой чисел.

Программа должна выяснить, может ли пассажир проследовать от одной остановки до другой, сделав не более одной пересадки.

С о в е т . Воспользуйтесь массивом множеств, индексируемым символами.

13 Глава 50
Записи (структуры)

Теоретическая база

- Тип данных «запись»
- Ввод и вывод записей
- Массивы записей
- Передача записей в качестве параметров процедур

13.1 Время

Источник: 1

Опишите тип данных (структуру) **`TTime`** для представления времени в виде часов, минут и секунд. Затем напишите следующие процедуры и функции.

Процедура **`ReadTime(var arg: TTime)`** вводит время с клавиатуры.

Процедура **`PrintTime(const arg: TTime)`** печатает время в формате ЧЧ:ММ:СС.

Функция **`IncTime(var arg: TTime):boolean`** увеличивает время на 1 секунду и возвращает **`TRUE`**, если после приращения времени часы переполняются (то есть, сбрасываются с 23 на 0 часов).

Функция **`TestTime(const arg: TTime):boolean`** возвращает **`TRUE`**, если переданное ей время логически корректно, то есть часы имеют значение в пределах от 0 до 23, а минуты и секунда — в пределах от 0 до 59.

На основе процедур и функций напишите программу для ввода времени, его наращивания на **`N`** секунд, проверки и печати результата.

13.2 Дата

Источник: 1

Опишите тип данных (структуру) **`TDate`** для представления даты в виде года, месяца и дня. Затем напишите следующие процедуры и функции.

Процедура **`ReadDate(var arg: TDate)`** вводит дату с клавиатуры.

Процедура **PrintDate** принимает тип **TDate** по ссылке **const** и печатает дату в формате ГГГГ-ММ-ДД.

Процедура **IncDate(var arg: TDate)** увеличивает дату на одни сутки.

Функция **TestDate(const arg: TDate): boolean** возвращает **TRUE**, если переданная ей дата логически корректна (дни соответствуют месяцам с учётом високосных годов).

На основе процедур и функций напишите программу для ввода даты, её наращивания на **N** суток, проверки и печати результата.

13.3 Дата и время

Источник: 1

Пусть дано следующее описание типов данных для представления даты и времени:

```
TDate = record  mYear, mMonth, mDay : word end;

TTime = record  mHour, mMin, mSec : word end;

TDateTime = record
  mDate : TDate;
  mTime : TTime;
end;
```

Напишите процедуры и функции для ввода даты и времени в переменную типа **TDateTime**, его наращивания на **N** секунд, проверки корректности и распечатки. Воспользуйтесь процедурами и функциями из двух предыдущих задач.

13.4 Сравнение даты и времени

Источник: --

Пусть дано описание типов данных для даты и времени, как предложено в задаче 13.3. Напишите функцию сравнения двух времён **T1** и **T2**, переданных в неё по ссылкам **CONST**. Функция должна возвращать **+1**, если **T1<T2**; **-1**, если **T1>T2**; и ноль, если времена совпадают.

13.5 Домино

Источник: "Песни о Паскале" 50-Д

В этой игре используют 28 костяшек, каждая из которых содержит пару чисел от 0 до 6. Например: 0:0, 1:5, 6:6. Представьте костяшку записью, а игральный набор – массивом

этих записей. Заполните массив костяшек и распечатайте его. «Смешайте» костяшки случайным образом и вновь распечатайте массив. Для удобства направьте распечатку в текстовый файл.

13.6 Карты

Источник: "Песни о Паскале" 50-Д

Колода содержит 36 карт четырех мастей: трефы и пики — черные, а бубны и червы — красные. Относительная сила карты определяется числом от 6 до 14. Представьте карту записью, содержащей её масть, цвет и силу. Представьте колоду массивом записей, сформируйте полную колоду и распечатайте в текстовый файл. «Перетасуйте» колоду и вновь распечатайте в файл. При распечатке силу карт от 11 до 14 напечатайте их названиями: валет, дама, король, туз.

13.7 Полицейская база данных

Источник: --

Полицейская база данных содержит следующие данные о жителях города:

- возраст в годах;
- номер паспорта (6-значное целое число);
- код городского района, где проживает гражданин (числа от 1 до 9);
- модель автомобиля, которой владеет житель (код модели — это числа от 1 до 9; если же он не автовладелец, то 0).

Подготовьте подходящий файл и введите из него данные в массив записей (или заполните массив случайными числами). Затем напишите три процедуры, выводящие данные о гражданах, которые удовлетворяют следующим требованиям:

- живущих в районе **`R`** и не моложе **`G`** лет;
- граждан возраста от **`G1`** до **`G2`** лет, владеющих автомобилями модели **`A`**;
- жителей районов **`R1`** и **`R2`**, не владеющих автомобилями.

Процедуры принимают соответствующие параметры, а данные извлекают из массива записей.

13.8 Школьная статистика

Источник: --

Сведения о выпускниках школ содержатся в массиве записей (заполните его случайно) и включают следующие данные:

- номер школы (числа от 1 до **`N`**, пусть **`N=5`**);

- сумма баллов ученика по трём ЕГЭ (в пределах 300);
- факт поступления на бюджетный факультет (**TRUE**, если поступил).

Необходимо определить, по меньшей мере, две школы:

- ту, где ученики в среднем набрали больше баллов по ЕГЭ;
- ту, где процент поступивших на бюджет больше.

В случае равных результатов в нескольких школах вывести их все.

П о д с к а з к а . Можно завести для школ вспомогательный массив из **N** записей, которые содержат следующие поля:

- номер школы;
- общее количество учеников школы;
- общее количество баллов по ЕГЭ;
- общее количество поступивших на бюджет.

13.9 Торговая фирма

Источник: --

Торговой фирме надо закупить крупную партию самоваров, которые производятся в других городах, причём цены на самовары существенно отличаются. Пусть массив записей содержит следующие сведения (заполните массив случайно):

- расстояние до города, где делают самовары (от 100 до 5000 км);
- цена одного самовара (от 1000 до 5000 руб).

Пусть известно количество закупаемых самоваров **N**, а стоимость проезда одного километра составляет **C** рублей. Помогите менеджеру выбрать подходящий город (то есть, индекс в массиве) так, чтобы сумма затрат была минимальна. Учтите, что транспорт следует туда и обратно.

13.10 Анаграммы (2)

Источник: --

Анаграммами называются слова, состоящие из одних и тех же букв. Напишите булеву функцию, принимающую по ссылке **CONST** две строки, и возвращающую **TRUE**, если строки состоят из одних и тех же символов, при этом требуется, чтобы каждый символ присутствовал в одинаковых количествах.

П о д с к а з к а . Для каждой строки создайте массив записей, включающих символ и счётчик, отсортируйте массивы записей по алфавиту и сравните.

14 Главы 51–56 Списки, очереди, стеки

Теоретическая база

- Указатели и динамические переменные
- Создание и уничтожение динамических переменных
- Организация связанных списков
- Обработка (проход) связанного списка
- Организация очередей и стеков посредством списков

14.1 Создание списка

Источник: --

Дан текстовый файл с числами. Написать процедуру и программу для чтения файла в односвязный список чисел.

14.2 Создание сортированного списка

Источник: --

Дан текстовый файл с числами. Написать процедуру и программу для чтения файла в сортированный односвязный список.

14.3 Создание сортированного списка из уникальных

Источник: --

Дан текстовый файл с числами. Написать процедуру и программу для чтения файла в сортированный односвязный список так, чтобы все числа в списке были уникальными (надо отбросить дубликаты).

14.4 Соединение и сортировка файлов

Источник: --

Даны два текстовых файла с числами. Написать процедуру и программу для объединения их в третий файл, составленный из уникальных чисел, упорядоченных по возрастанию. Каждый входной файл читать единожды.

П о д с к а з к а . Предварительно перенесите данные в сортированные списки уникальных чисел.

14.5 Разделение чисел

Источник: --

Входной текстовый файл содержит неупорядоченный набор целых положительных чисел. Создайте выходной файл с теми же числами, но сгруппированными так: сначала следуют нечётные, затем чётные. Внутри этих групп порядок чисел прежний.

П о д с к а з к а . Примените очередь чисел.

14.6 Разделение уникальных чисел

Источник: --

Входной текстовый файл содержит неупорядоченный набор целых положительных чисел. Создайте выходной файл с теми же числами следующими так: сначала нечётные, затем чётные. Внутри этих групп порядок чисел прежний, но дубликаты отбрасываются, то есть, каждое число в выходном файле встречается лишь однажды.

П о д с к а з к а . Создайте очередь, в которую добавляются лишь числа, в ней отсутствующие, уникальные.

14.7 Подсчёт слов из одних и тех же букв

Источник: "Песни о Паскале" 55-В

Создайте программу для подсчета в файле слов, составленных из одних и тех же символов: цифр и букв, например: «**end**» и «**deen**», «**121**» и «**221**». Для каждой такой группы слов программа должна распечатать перечень входящих в них символов (каждый символ – по разу) и количество обнаруженных слов этой группы, например, для приведенных выше слов:

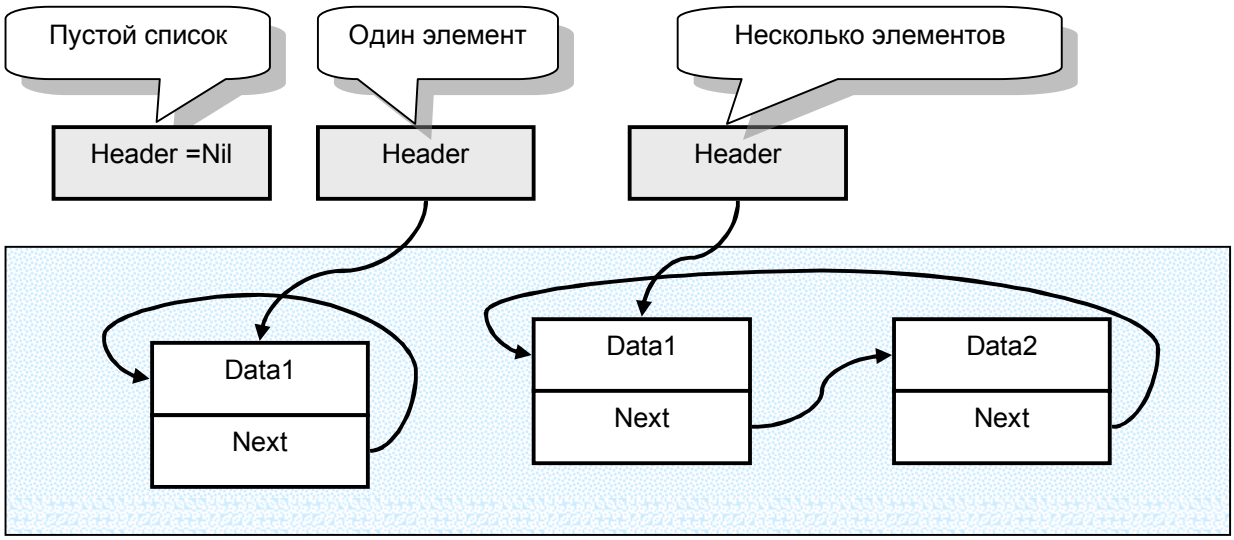
```
1 2 – 2
d e n – 2
```

П о д с к а з к а : примените список записей, которые содержат в себе множество символов. Для слов и чисел, составленных из одинаковых символов, эти множества совпадают.

14.8 Кольцевой список

Источник: --

Кольцевой односвязный список отличается от обычного тем, что последний его элемент связан с первым (см. рисунок ниже). Поэтому понятия «первый» и «последний» в кольцевом списке весьма условны, и заголовок списка может указывать на любой его элемент. Двигаясь по кольцу, можно пройти весь список. Если список содержит единственный элемент, то он связывается сам с собой.



Дан текстовый файл с числами. Написать процедуры чтения его в кольцевой список и распечатки кольцевого списка.

14.9 Длина кольцевого списка

Источник: --

Создайте кольцевой список (см. задачу 14.7), затем напишите функцию, принимающую указатель на кольцевой список, и возвращающую количество элементов в нём (для пустого указателя — ноль).

П о д с к а з к а . Надо двигаться по списку, пока не вернётесь на прежнее место.

14.10 «Русская рулетка»

Источник: --

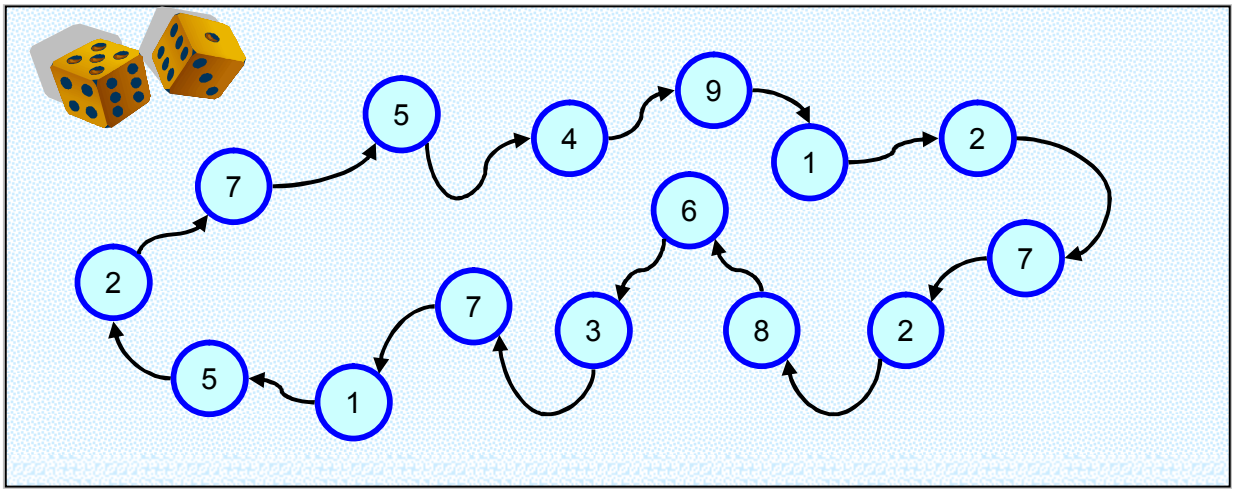
Создайте кольцевой список (см. задачу 14.7), затем напишите функцию, возвращающую число из этого списка, выбранное случайным образом.

П о д с к а з к а . Сдвигайте текущую позицию в списке на случайное число шагов.

14.11 Модель настольной игры

Источник: --

Поле для настольной игры — это соединённые линиями кружочки, образующие замкнутый маршрут. Кружочки содержат цифры от 1 до 9. В игре участвуют два игрока, представленные на игровом поле двумя фишками: синей и красной. Игроки делают свои ходы, поочерёдно подбрасывая кубик (что порождает числа от 1 до 6). Далее будут рассмотрены несколько вариантов игры по разным правилам.



Пусть имеются два текстовых файла, первый из которых содержит несколько десятков чисел от 1 до 9, — создайте из него кольцевой список (см. задачу 14.7). Второй файл содержит результаты подбрасывания кубика (ходы игроков), в каждой строке по два числа от 1 до 6: для синей фишки и для красной.

Правило 1. Оба игрока делают равное число ходов из некоторой исходной точки маршрута, количество шагов определяется количеством строк входного файла, а каждая строка содержит по два числа — это результаты подбрасывания кубика для двух игроков. Подбросив кубик, игрок перемещается на несколько шагов вперёд и прибавляет на свой счёт число из посещённого им кружочка. Выигрывает тот, чья сумма окажется больше. Программа должна определить выигравшую фишку и напечатать одно из трёх: «синяя», «красная», «ничья».

Правило 2. То же, что и Правило 1, но если текущая накопленная сумма окажется кратной 10, то фишка получает бонус: делает ещё один шаг вперёд и добавляет баллы из этого посещённого кружочка.

Правило 3. Количество шагов для фишки определяется суммированием того, что дал кубик, с тем, что содержится в текущем кружочке (в итоге будет от 2 до 15). Движение продолжается, пока фишка не совершит полный круг (или **N** кругов), то есть достигнет исходной точки или пройдёт через неё. Если файл ходов исчерпан, то повторно читается с начала. Здесь тоже возможна ничья.

Правило 4. Шаги вычисляются по Правилу 3, но выигрывает фишка, первая посетившая все кружочки (под посещением понимается остановка в данном кружочке). Ничья не исключена.

П о д с к а з к а . Для правила 4 в элементах списка (кружочках) надо предусмотреть два булевых поля, отражающих факт посещения её фишками.

14.12 Числа Хэмминга

Источник: --

Числами Хэмминга называют такие натуральные числа, которые делятся без остатка только на 2, 3 и 5. Вот несколько первых чисел Хэмминга:

```
   2   3   4   5   6   8   9  10  12  15  16  18  20  24  25  27
```

Напишите программу для формирования первых **N** чисел Хэмминга, где **N** задаётся пользователем.

П о д с к а з к а : Основная идея — в применении упорядоченной очереди (списка) уникальных чисел. Вначале в очередь помещают первые три элемента: числа 2, 3 и 5. Затем в цикле извлекают первый элемент **R** (это очередной результат) и добавляют три новых, равных **2•R**, **3•R** и **5•R**. В ходе решения очередь растёт, но не очень быстро, поскольку повторная вставка дубликатов не выполняется.

Входные данные	Результат
N=100	2 3 4 5 6 8 9 10 12 15 16 18 20 24 25 27 30 32 36 40 45 48 50 54 60 64 72 75 80 81 90 96 100 108 120 125 128 135 144 150 160 162 180 192 200 216 225 240 243 250 Осталось в очереди: 36 элементов

15 Главы 57–58 Графы

Теоретическая база

- Внешнее и внутреннее представление графа
- Ввод и вывод графа в файл
- Обход графа «в ширину»

15.1 Исследование графа (1)

Источник: --

Входной файл содержит произвольное перечисление вершин ненаправленного графа: в каждой строке сначала следует номер вершины, а затем её соседи (если они есть).

Программа должна проверить:

- является ли данный граф связанным (когда можно попасть из любой вершины в любую другую);
- является ли данный граф деревом (дерево — это связанный граф, где от вершины к вершине возможен лишь единственный путь);

П о д с к а з к а . Если при обходе в ширину обнаружена хотя бы одна серая вершина, то такой граф не является деревом.

15.2 Исследование графа (2)

Источник: --

Входной файл содержит произвольное перечисление вершин графа: в каждой строке сначала следует номер вершины, а затем её соседи.

Программа должна проверить, является ли граф однонаправленным, когда любая пара вершин связана не более чем одной дугой. Например, когда есть связь **`A --> B`**, но нет связи **`A <-- B`**. Если хотя бы одна пара взаимно связана, считать граф двунаправленным.

15.3 Опадающие листья

Источник: --

Входной файл содержит произвольное перечисление вершин ненаправленного графа: в каждой строке сначала следует номер вершины, а затем номера её соседей. Количество вершин в графе нечётное.

Этот граф изображает города и посёлки, связанные сетью дорог. Некоторые посёлки связаны лишь с одним населённым пунктом — это окраины, листья графа. Из них люди ежегодно переселяются в соседние посёлки и города. После этого посёлки пустеют и удаляются с карты: так листья графа «отсыхают», но при этом могут образоваться новые листья. В следующем году всё повторяется, если в графе ещё остаются листья. Программа должна найти конечный состав графа после «опадения» всех возможных листьев (надо вывести оставшиеся вершины).

15.4 Путь с пересадками (1)

Источник: --

Маршруты городских автобусов названы заглавными буквами латинского алфавита: **A**, **B**, **C** и т.д., а остановки автобусов обозначены числами. Некоторые маршруты могут пересекаться или частично совпадать (когда автобусы посещают одни и те же остановки).

Входной файл содержит следующие данные. В первой строке — два числа, указывающие, откуда и куда едет пассажир (начальную и конечную остановки). Последующие строки — это перечень остановок каждого из городских маршрутов в их алфавитном порядке; здесь каждый маршрут представлен отдельной строкой чисел.

Стремясь сэкономить, пассажир выбирает транспорт так, чтобы количество пересадок для него было минимальным (в идеале — прямой маршрут). Пусть программа выберет ему подходящий транспорт: надо напечатать последовательность маршрутов или слово «Нет», если ни один маршрут не подходит.

15.5 Путь с пересадками (2)

Источник: --

Маршруты городских автобусов названы заглавными буквами латинского алфавита: **A**, **B**, **C** и т.д., а остановки автобусов обозначены числами. Некоторые маршруты могут пересекаться или частично совпадать (когда автобусы посещают одни и те же остановки).

Входной файл содержит следующие данные. В первой строке — два числа, указывающие, откуда и куда следует пассажир (начальную и конечную остановки). Последующие строки — это перечень остановок каждого из городских маршрутов в их алфавитном порядке;

здесь каждый маршрут представлен отдельной строкой чисел, а номера остановок в строке следуют в порядке посещения их автобусом.

Пассажир выбирает транспорт так, чтобы снизить количество проезжаемых остановок. Пусть программа выберет ему подходящий маршрут: здесь надо напечатать последовательность промежуточных остановок или слово «Нет», если путь невозможен.

15.6 Граф с расстояниями

Источник: "Песни о Паскале" 58-Б

Пусть узлы графа – это города, а ребра – дороги между ними. Расстояния между городами разные, они известны и хранятся в поле **mDist** записи **TLink** так:

```
TLink = record       { Тип список связей }
  mLink : PNode;     { указатель на смежный узел }
  mDist : integer;   { расстояние между городами }
  mNext : PLink;     { указатель на следующую запись в списке }
end;
```

Предложите формат входного файла, содержащего в числе прочего расстояния между городами.

15.7 Ввод графа с указанием расстояний

Источник: "Песни о Паскале" 58-Г

Пусть выбран следующий формат входного файла с учетом расстояний между городами (приведена одна строка):

```
A C 20 E 40
```

Здесь первый символ, как и ранее, обозначает текущий узел. Затем перечисляются его соседи с указанием расстояний до них. Например, между узлами «A» и «C» 20 км, а между узлами «A» и «E» – 40 км. Напишите процедуру для ввода графа из такого файла.

15.8 Поиск кратчайшего пути с учётом расстояний

Источник: "Песни о Паскале" 58-Д

Напишите программу для поиска кратчайшего пути с учетом расстояний между городами. Подсказка: измените процедуру обхода в ширину так, чтобы серый кандидат исследовал всех соседей (не только белых), проверяя в них поле расстояния **mDist**. Если путь к соседу через кандидата окажется короче того, что уже отмечен в соседе, то следует

изменить как расстояние, так и обратную ссылку в соседе. Вдобавок если сосед не серый, он ставится в очередь.

15.9 Купеческий конгресс (1)

Источник: "Песни о Паскале" 58-Ж

С некоторых пор купцы учредили свой ежегодный съезд – Континентальный Купеческий Конгресс, где обсуждали свои проблемы. Каждая страна отправляла на съезд по одному делегату, а расходы на пересечение границ (визы) оплачивались из общей кассы. Посчитайте эти расходы (1 пиастр за каждое пересечение), если известна страна проведения конгресса. Считайте, что купцы следовали на съезд кратчайшими маршрутами, а расстояния не учитываются.

15.10 Купеческий конгресс (2)

Источник: "Песни о Паскале" 58-З

Напишите программу для определения страны, где можно провести съезд с наименьшими издержками (см. предыдущую задачу).

15.11 Купеческий конгресс (3)

Источник: "Песни о Паскале" 58-И

Решите две предыдущие задачи для случая разной стоимости виз на границах.

15.12 Императорские заботы

Источник: "Песни о Паскале" 59-Б

Императорские заботы. После постройки империи бывшие независимые государства стали провинциями и породили новые проблемы. Для доставки туда правительственных бумаг император нанял гонцов, которые для ускорения доставки следовали из столицы кратчайшими путями и лишь в одном направлении — от центра к окраинам империи. Сколько гонцов для этого нужно? — вот первый вопрос. Сколько времени потребуется для достижения самых дальних окраин, если переход из провинции в провинцию отнимает сутки? — это второй вопрос. В конечных пунктах (на окраинах) перед возвращением гонцам нужен отдых, на каких окраинах построить гостиницы? — это третий вопрос.

П о д с к а з к а : возьмите за основу программу обхода графа в ширину.

15.13 Дешёвая дорога

Источник: —

Жители нескольких глухих деревень договорились соединиться шоссейными дорогами так, чтобы можно было проехать из одной деревни в любую другую. Для экономии решили минимизировать суммарную длину дорог.

Пусть дан файл, первая строка которого содержит расстояния между первой деревней и всеми остальными, вторая — между второй деревней и всеми остальными и так далее. Ваша программа должна определить минимальную суммарную длину дорог.

П о д с к а з к а : воспользуйтесь алгоритмом Прима, суть которого такова. Сначала выбирается любая деревня, например, первая, отыскивается ближайшая к ней деревня и они соединяются. Затем среди оставшихся деревень отыскивается ближайшая к уже соединённым и соответствующая пара деревень соединяется. Так продолжается, пока остаются не присоединённые деревни.

16 Вычислительная геометрия

Теоретическая база

- Представление точки на плоскости (координаты)
- Расстояние между точками
- Представление линии
- Представление фигур

16.1 Точка на плоскости

Источник: 1

Точка на плоскости задаётся двумя своими координатами: абсциссой **X** и ординатой **Y**. Опишем её записью (здесь координаты — целые числа, они могут быть и отрицательными):

```
TPoint = record  X, Y : integer end;
```

Пусть дан массив из таких записей — это результаты стрельбы по мишени, измеренные посредством особой компьютерной системы. Надо определить индекс самого меткого выстрела (центр мишени — это начало координат).

П о д с к а з к а . Расстояние R от точки до начала координат вычисляется по теореме Пифагора:

$$R = \sqrt{X^2 + Y^2}$$

Но для сравнения расстояний двух точек квадратный корень извлекать не обязательно.

16.2 Расстояния между точками

Источник: 1

Дан массив точек, каждая из которых представлена записью (см. 16.1): Найдите:

а) две ближайшие точки (их индексы в массиве);

б) две наиболее удалённые точки.

16.3 Столица

Источник: 1

Положение городов на карте дано в виде массива точек (см. 16.1). Правительство решило перенести столицу в один из этих городов так, чтобы сумма расстояний от столицы до прочих городов была минимальна. Пусть программа определит подходящую точку для столицы (индекс в массиве).

16.4 Удалённость от центра (1)

Источник: 1

Задан неупорядоченный массив точек **P** с целочисленными координатами (см. 16.1) — это координаты городов страны, столица которой расположена в точке (0,0) — начале координат.

Специальный самолёт развозит правительственные документы из столицы в порядке удалённости городов от цента, а удалённость от столицы определена так: точка **P1** ближе точки **P2**, если:

- либо **P1.X < P2.X**,
- либо **(P1.X = P2.X)** и **(P1.Y < P2.Y)**.

Отсортируйте массив точек в порядке удалённости от центра.

16.5 Удалённость от центра (2)

Источник: 1

Задан неупорядоченный массив точек **P** с положительными координатами как в задаче 13.4 — это координаты городов страны, столица которой расположена в точке (0,0) — начале координат.

Правительственный самолёт развозит документы из столицы в порядке удалённости городов от неё, а удалённость от столицы определена так: считается, что точка **P1** ближе точки **P2**, если:

- либо **(P1.X + P1.Y) < (P2.X + P2.Y)**,
- либо **(P1.X + P1.Y) = (P2.X + P2.Y) и (P1.X < P2.X)**.

Отсортируйте массив точек в порядке удалённости от центра.

16.6 Удалённость от центра (3)

Источник: 1

Задан неупорядоченный массив точек **P** с координатами как в задаче 13.4 — это координаты городов страны, столица которой расположена в точке (0,0) — начале координат.

Правительственный самолёт развозит документы из столицы в порядке удалённости городов от неё, а удалённость от столицы определена так: считается, что точка **P1** ближе точки **P2**, если:

(P1.X*P1.X + P1.Y*P1.Y) < (P2.X*P2.X + P2.Y*P2.Y).

Отсортируйте массив точек в порядке удалённости от центра.

П о д с к а з к а . Во избежание переполнений используйте действительный тип данных.

16.7 Прямоугольник

Источник: --

Прямоугольник (**Rectangle**) определяют парой точек **P1** и **P2** (см. рисунок ниже). Соответственно в программах его представляют записью из двух точек:

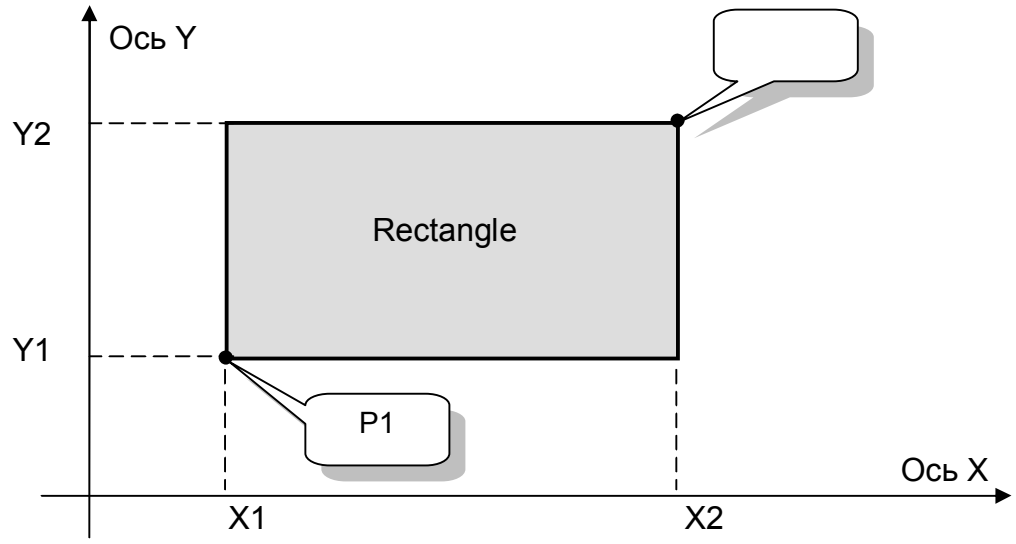
```
TPoint = record              { точка }
           X, Y : integer    { абсцисса и ордината точки }
         end;

TRect = record          { прямоугольник }
         P1 : TPoint;   { левая нижняя точка }
         P2 : TPoint;   { правая верхняя точка }
       end;
```

В программе надо объявить переменную типа прямоугольник, ввести координаты двух его точек и вычислить:

- площадь прямоугольника;
- периметр прямоугольника.

Оформите вычисление площади и периметра двумя функциями, принимающими ссылку **CONST** на данные типа **TRect**. и возвращающие тип **Longint**.



16.8 Сдвиг прямоугольника

Источник: --

В этой и нескольких последующих задачах определим ряд операций с прямоугольниками.

Напишите и протестируйте процедуры, выполняющие сдвиги прямоугольника следующими способами:

```
Shift(var B: TRect; dX, dY : integer); { на dX и dY вдоль осей}
Move1(var B: TRect; P: TPoint);        { точку P1 смещаем в точку P }
Move2(var B: TRect; P: TPoint);        { точку P2 смещаем в точку P }
```

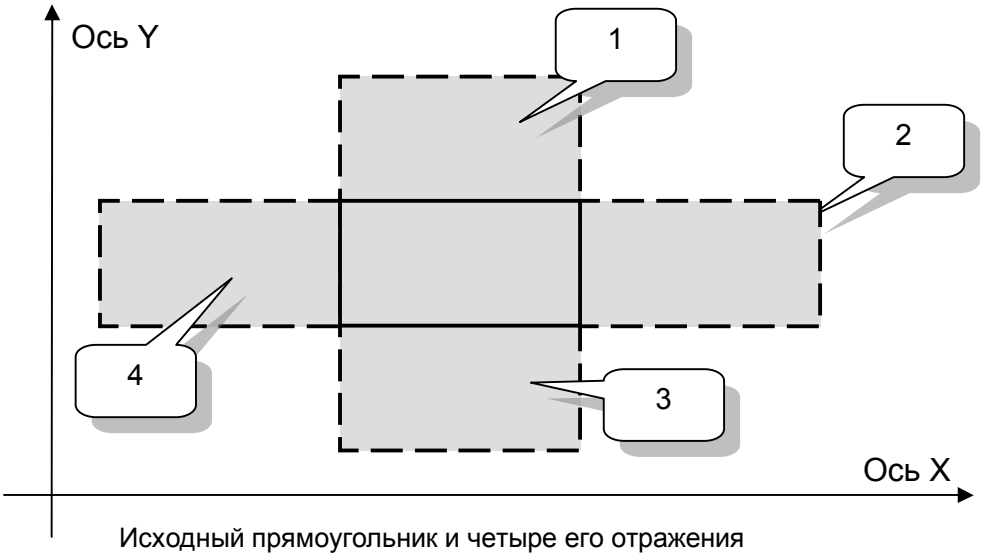
При этом площади и периметры сдвинутых прямоугольников не изменяются.

16.9 Повороты вокруг сторон

Источник: --

Напишите процедуру, принимающую два параметра: 1) ссылку **VAR** на прямоугольник типа **TRect**, и 2) целое число от 1 до 4, указывающее одну из сторон, относительно которой надо повернуть прямоугольник:

- 1 — относительно верхней;
- 2 — относительно правой;
- 3 — относительно нижней;
- 4 — относительно левой.



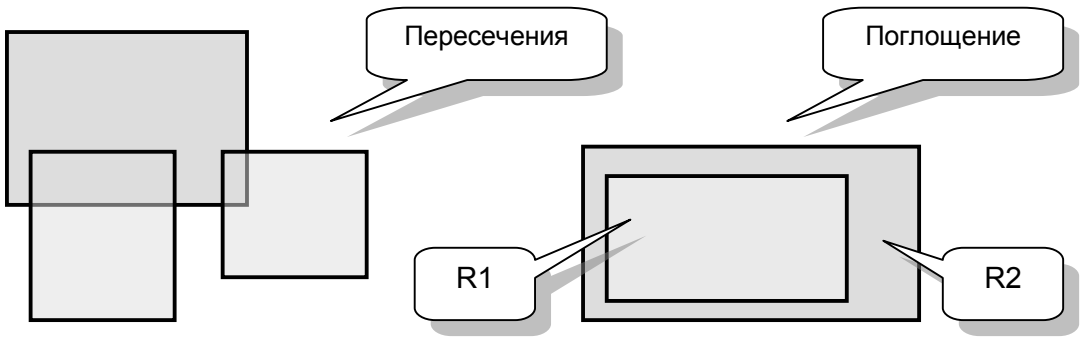
Исходный прямоугольник и четыре его отражения

Для проверки убедитесь, что площади и периметры полученных прямоугольников не изменились.

16.10 Проверка на пересечение и поглощение

Источник: --

Напишите две булевы функции, принимающие по ссылкам **CONST** два прямоугольника типа **TRect**. Первая возвращает **TRUE**, если прямоугольники пересекаются, то есть, содержат хотя бы одну общую точку. Вторая возвращает **TRUE**, если первый прямоугольник **R1** совпадает со вторым **R2** или полностью им поглощается (см. рисунок ниже).



16.11 Сложение прямоугольников

Источник: --

Суммой двух прямоугольников **R1+R2** будем считать такой минимальный прямоугольник **R3**, который охватывает два исходных (см. рисунок ниже).

Напишите процедуру, принимающую по ссылкам **CONST** два прямоугольника типа **TRect**. и возвращающую по ссылке **VAR** их сумму. Примените функцию из предыдущей задачи для проверки, что вновь полученный прямоугольник **R3** поглощает оба исходных.