

Учреждение образования
«Гомельский государственный университет
имени Франциска Скорины»

Е. А. РУЖИЦКАЯ, М. В. МОСКАЛЕВА

УЧЕБНАЯ ВЫЧИСЛИТЕЛЬНАЯ ПРАКТИКА

Язык программирования Pascal

Практическое пособие

для студентов специальности
1–31 03 03–02 «Прикладная математика
(научно-педагогическая деятельность)»

Гомель
ГГУ им. Ф. Скорины
2019

УДК 004.43(076)
ББК 32.973.22я73
Р837

Рецензенты:

кандидат технических наук В. Д. Левчук,
кандидат технических наук О. И. Еськова

Рекомендовано к изданию научно-методическим советом
учреждения образования «Гомельский государственный
университет имени Франциска Скорины»

Ружицкая, Е. А.

Р837 Учебная вычислительная практика : язык программирования Pascal : практическое пособие / Е. А. Ружицкая, М. В. Москалева ; Гомельский гос. ун-т им. Ф. Скорины. – Гомель : ГГУ им. Ф. Скорины, 2019. – 48 с.
ISBN 978-985-577-558-5

Практическое пособие предназначено для оказания помощи студентам при выполнении заданий по учебной вычислительной практике на 1 курсе в 1 семестре. В нем излагается теоретический материал, даны практические задания по учебной вычислительной практике по разделу «Язык программирования Pascal» и примеры их выполнения.

Адресовано студентам 1 курса специальности 1–31 03 03–02 «Прикладная математика (научно-педагогическая деятельность)».

**УДК 004.43(076)
ББК 32.973.22я73**

ISBN 978-985-577-558-5 © Ружицкая Е. А., Москалева М. В., 2019
© Учреждение образования «Гомельский
государственный университет
имени Франциска Скорины», 2019

Оглавление

Предисловие.....	4
Тема 1. Вычисление значений функции, представленной разложением в ряд.....	5
Тема 2. Задачи целочисленной арифметики	10
Тема 3. Методы сортировок	14
Тема 4. Модульная структура программы	24
Тема 5. Файлы записей	28
Тема 6. Графические возможности языка Pascal	41
Требования к оформлению отчета	46
Литература	47

Предисловие

Учебная вычислительная практика предназначена для формирования прочных знаний и практических навыков в области алгоритмизации и программирования.

На первом курсе в первом семестре при прохождении учебной вычислительной практики изучаются следующие темы:

1. Вычисление значений функции, представленной разложением в ряд. Рассматриваются три типа общего члена ряда и особенности его вычисления: непосредственное, рекуррентное и комбинированное.

2. Задачи целочисленной арифметики. Представление чисел в виде массива цифр. Схема Горнера. Делимость чисел.

3. Методы сортировок. Методы, основанные на выборе: линейный выбор, линейный выбор с обменом, линейный выбор с подсчетом. Методы сортировки обменом: парный обмен, стандартный обмен, метод просеивания. Метод линейной вставки.

4. Модульная структура программы. Модули, структура модулей. Стандартный модуль CRT. Организация меню.

5. Файлы записей. Определение файлового типа, процедуры и функции обработки файлов.

6. Графические возможности языка Pascal. Библиотека Graph. Построение графиков функций. Рисование геометрических фигур.

Практическое пособие предназначено для оказания помощи студентам в овладении языком программирования Pascal. Излагается теоретический материал, даны практические задания и примеры их выполнения.

Адресовано студентам 1 курса специальности 1–31 03 03–02 «Прикладная математика (научно-педагогическая деятельность)».

Тема 1. Вычисление значений функции, представленной разложением в ряд

Функцию $f(x)$, $n+1$ раз дифференцируемую на интервале (a, b) , содержащем точку c , можно разложить в ряд Тейлора, т. е. представить в виде суммы многочлена n -й степени и остаточного члена R_n :

$$f(x) = f(c) + \frac{f'(c)}{1!}(x-c) + \frac{f''(c)}{2!}(x-c)^2 + \dots + \frac{f^{(n)}(c)}{n!}(x-c)^n + R_n.$$

Выражение $Q_n = \frac{f^{(n)}(c)}{n!}(x-c)^n$ называется общим членом ряда. Необходимым условием сходимости ряда является то, что общий член ряда по абсолютной величине стремится к нулю.

Вычислять значение суммы будем следующим образом: зададим начальное значение суммы, вычислим первый член суммы и добавим его к начальному значению, вычислим второй член суммы, третий и т. д., до тех пор, пока значение n -го члена суммы, по абсолютной величине, не будет меньше заданной точности ε (добавление его к сумме не повлияет на значение суммы). Формула общего члена ряда может принадлежать к одному из трех типов:

- 1) $\frac{\cos nx}{n}; \frac{\sin(2n-1)x}{2n-1}; \frac{\cos 2nx}{4n^2-1}; \dots$
- 2) $\frac{x^n}{n!}; (-1)^n \frac{x^{2n+1}}{(2n+1)!}; \frac{x^{2n}}{(2n)!}; \dots$
- 3) $\frac{x^{4n+1}}{4n+1}; (-1)^n \frac{\cos nx}{n^2}; \frac{n^2+1}{n!} \cdot \left(\frac{x}{2}\right)^n; \dots$

В первом случае каждый член суммы вычисляется непосредственно по общей формуле. Во втором случае для вычисления суммы лучше всего использовать рекуррентные соотношения, т. е. выражать последующий член ряда через предыдущий. В последнем случае член суммы представляется в виде произведения двух сомножителей, первый из которых вычисляется с использованием рекуррентных соотношений, а второй вычисляется непосредственно.

Найдем рекуррентные соотношения для ряда, общий член которого имеет вид: $u_n = \frac{x^n}{n!}$, тогда $u_{n+1} = \frac{x^{n+1}}{(n+1)!}$. Найдем отношение $\frac{u_{n+1}}{u_n}$:

$$\frac{u_{n+1}}{u_n} = \frac{x^{n+1}}{(n+1)!} \cdot \frac{n!}{x^n} = \frac{x^n \cdot x}{n! \cdot (n+1)} \cdot \frac{n!}{x^n} = \frac{x}{n+1}. \text{ Значит, } u_{n+1} = u_n \cdot \frac{x}{n+1}.$$

Если при подстановке $n = 1$ в общий член суммы будет получен первый член ряда, то начальное значение $S = 0$, если же будет получен второй член ряда, то за начальное значение S_n принимают значение первого члена ряда.

Начальное значение для рекуррентных соотношений определяется из первых членов суммы ряда путем выделения в них той части, которая вычисляется рекуррентно.

Вычисление суммы организовывается в цикле. Когда при прохождении цикла номер члена суммы изменяется на 1, то сумма изменяется на его n -й член, т. е. $S_{n+1} = S_n + Q_{n+1}$, S_n – сумма n членов. Вычисления проводят до тех пор, пока не будет выполнено неравенство $|Q_n| \leq \varepsilon$.

Практическое задание

С точностью $\varepsilon = 0,001$ подсчитать значение S_n функции $F(x)$, представленной разложением в ряд $S = S(x)$. Результат сравнить со значением функции $F(x)$. Вычисления произвести в диапазоне изменения аргумента $a \leq x \leq b$ с заданным шагом $h = (b - a) / k$, $k = 10$. На печать выдавать в виде таблицы: аргумент x , значения S и F , количество членов ряда n , обеспечивающих заданную точность, и значения n -го члена ряда Q_n . Написать два варианта программы, используя операторы циклов While и Repeat.

Варианты

№	$S(x)$	$F(x)$	$[a, b]$
1	$x + \frac{x^3}{3} + \frac{x^5}{5} + \dots + \frac{x^{2n+1}}{2n+1} + \dots$	$\frac{1}{2} \ln \frac{1+x}{1-x}$	$[0,1; 1)$
2	$1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} + \dots + \frac{(-1)^n x^n}{n!} + \dots$	e^{-x}	$[0,1; 0,6]$
3	$1 + \frac{2x}{1!} + \dots + \frac{(2x)^n}{n!} + \dots$	e^{2x}	$[0,1; 0,6]$
4	$x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} - \dots + \frac{(-1)^n x^{n+1}}{n+1} + \dots$	$\ln(x+1)$	$[0,1; 1]$

5	$-\cos x + \frac{\cos 2x}{2^2} + \dots + (-1)^n \frac{\cos nx}{n^2} + \dots$	$\frac{1}{4} \left(x^2 - \frac{\pi^2}{3} \right)$	$\left[\frac{\pi}{5}; \pi \right]$
6	$x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \dots + (-1)^{n+1} \frac{x^n}{n} + \dots$	$\ln(1+x)$	$[0,2; 1]$
7	$\cos x + \frac{\cos x}{3^2} + \dots + \frac{\cos(2n-1)x}{(2n-1)^2} + \dots$	$\frac{\pi^2}{8} - \frac{\pi}{4} x $	$\left[\frac{\pi}{5}; \pi \right]$
8	$-(1+x)^2 + \frac{(1+x)^4}{2} - \dots + (-1)^n \frac{(1+x)^{2n}}{n} + \dots$	$\ln \frac{1}{2+2x+x^2}$	$[-2; 0]$
9	$x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \dots + (-1)^n \frac{x^{2n+1}}{2n+1} + \dots$	$\arctg(x)$	$[0,1; 1]$
10	$1 + 2\frac{x}{2} + \dots + \frac{n^2+1}{n!} \left(\frac{x}{2} \right)^n + \dots$	$\left(\frac{x^2}{4} + \frac{x}{2} + 1 \right) e^{\frac{x}{2}}$	$[0,1; 1]$
11	$\frac{x^3}{3} - \frac{x^5}{15} + \dots + (-1)^{n+1} \frac{x^{2n+1}}{4n^2-1} + \dots$	$\frac{x^2+1}{2} \arctg(x) - \frac{x}{2}$	$[0,1; 0,6]$
12	$-\frac{(2x)^2}{2} + \frac{(2x)^4}{24} - \dots + (-1)^n \frac{(2x)^{2n}}{(2n)!} + \dots$	$2(\cos^2(x) - 1)$	$[0,1; 1]$
13	$1 - x^2 + \frac{x^4}{2!} - \frac{x^6}{3!} + \dots + \frac{(-1)^n x^{2n}}{n!} + \dots$	e^{-x^2}	$[0,1; 1]$
14	$1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots + \frac{(-1)^n x^{2n}}{(2n)!} + \dots$	$\cos(x)$	$[0; \pi]$
15	$1 - \frac{x^2}{3!} + \frac{x^4}{5!} - \frac{x^6}{7!} + \dots + \frac{(-1)^n x^{2n}}{(2n+1)!} + \dots$	$\frac{\sin(x)}{x}$	$[0,1; 1]$
16	$\frac{x-1}{x+1} + \frac{1}{3} \left(\frac{x-1}{x+1} \right)^3 + \dots + \frac{1}{2n+1} \left(\frac{x-1}{x+1} \right)^{2n+1} + \dots$	$\frac{1}{2} \ln(x)$	$[0,2; 1]$
17	$\frac{x-1}{x} + \frac{(x-1)^2}{2x^2} + \frac{(x-1)^3}{3x^3} + \dots + \frac{(x-1)^{n+1}}{(n+1)x^{n+1}} + \dots$	$\ln(x)$	$[0,6; 0,9]$
18	$\cos x + \frac{\cos 2x}{2} + \dots + \frac{\cos nx}{n} + \dots$	$-\ln \left 2 \sin \frac{x}{2} \right $	$\left[\frac{\pi}{5}; \frac{9\pi}{5} \right]$
19	$1 + 3x^2 + \dots + \frac{2n+1}{n!} x^{2n} + \dots$	$(1+2x^2)e^{x^2}$	$[0,1; 1]$
20	$\frac{1}{5} + \frac{1}{5^2}x + \frac{1}{5^3}x^2 + \dots + \frac{1}{5^{n+1}}x^n + \dots$	$-\frac{1}{x-5}$	$[1; 2]$
21	$x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots + \frac{(-1)^n x^{2n+1}}{(2n+1)!} + \dots$	$\sin(x)$	$[0,1; 1]$

22	$x + \frac{x^3}{6} + \frac{3x^5}{40} + \dots \frac{(2n)!x^{2n+1}}{(2n+1)(n!)^2 4^n} + \dots$	$\arcsin(x)$	$[0,1; 1)$
23	$\frac{\pi}{2} - \left(x + \frac{x^3}{6} + \frac{3x^5}{40} + \dots + \frac{(2n)!x^{2n+1}}{(2n+1)n!^2 4^n} + \dots \right)$	$\arccos(x)$	$[0,1; 1)$
24	$1 + \frac{x}{2} - \frac{x^2}{8} + \frac{x^3}{16} - \dots + \frac{(-1)^n (2n)!x^n}{(1-2n)(n!)^2 4^n} + \dots$	$\sqrt{1+x}$	$[0,1; 0,9]$
25	$1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \dots + \frac{x^{2n}}{(2n)!} + \dots$	$\frac{e^x + e^{-x}}{2}$	$[0,1; 1]$
26	$x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots + \frac{(-1)^{n+1} x^{2n-1}}{(2n-1)!} + \dots$	$\sin(x)$	$[0,1; 1]$
27	$3x + 8x^2 + \dots + n(n+2)x^n + \dots$	$\frac{x(3-x)}{(1-x)^3}$	$[0,1; 0,8]$
28	$1 - x + x^2 + \dots + (-1)^n x^n + \dots$	$\frac{1}{1+x}$	$[0,1; 1)$
29	$\sin(x) - \frac{\sin(2x)}{2} + \dots + (-1)^{n+1} \frac{\sin(nx)}{n} + \dots$	$\frac{x}{2}$	$\left[\frac{\pi}{5}; \frac{4\pi}{5} \right]$
30	$\frac{1}{x} + \frac{1}{3x^3} + \frac{1}{5x^5} + \dots + \frac{1}{(2n+1)x^{2n+1}} + \dots$	$\frac{1}{2} \ln \frac{x+1}{x-1}$	$[0,1; 0,8]$

Пример выполнения задания с использованием цикла **while**

Пусть дано разложение функции $F = \frac{x^2 + 1}{2} \operatorname{arctg}(x) - \frac{x}{2}$ в ряд

$$S = \frac{x^3}{3} - \frac{x^5}{15} + \dots + (-1)^{n+1} \frac{x^{2n+1}}{4n^2 - 1} + \dots,$$

сходящийся на интервале $0,1 \leq x \leq 0,8$.

В данном примере общий член ряда $Q_n = (-1)^{n+1} \frac{x^{2n+1}}{4n^2 - 1}$ принадлежит к третьему типу. Представим его в виде произведения двух сомножителей $Q_n = u_n p_n$, где $u_n = (-1)^{n+1} x^{2n+1}$ вычисляется рекуррентно, а $p_n = \frac{1}{4n^2 - 1}$ — непосредственно. Найдем рекуррентные соотношения:

$$\frac{u_{n+1}}{u_n} = \frac{(-1)^{n+2} x^{2(n+1)+1}}{(-1)^{n+1} x^{2n+1}} = \frac{(-1)^{n+1} (-1) x^{2n+1} x^2}{(-1)^{n+1} x^{2n+1}} = -x^2.$$

Значит, $u_{n+1} = -x^2 u_n$.


```

program uvp1;
const  eps=0.001;
        k  =5;

var
    a,b,h,x,s,q,u,f :Real;
    n                 :Integer;
begin
    Write ('Введите отрезок [a,b]->'); ReadLn (a,b);
    h:=(b-a)/k; x:=a;
    WriteLn('          Таблица значений функции ');
    WriteLn(' ');
    WriteLn('  x      S      f      Q      n  ');
    WriteLn('  ');
    while (x<=b) do
    begin
        s:=0; n:=1; u:=x*x*x; q:=u/3;
        while (abs(q)>eps) do
        begin
            s:=s+q; n:=n+1;
            u:=u*(-x*x);
            q:=u/(4*n*n-1);
        end;
        f:=(x*x+1)/2*arctan(x)-x/2;
        WriteLn(' |',x:4:2,' |',s:8:4,' |',f:8:4,' |',
                q:8:4,' |',n:3,' |');
        x:=x+h;
    end;
    WriteLn (' |-----| ');
End.

```

Результат работы программы

Введите отрезок [a,b]->0.1 0.8
Таблица значений функции

x	S	f	Q	n
0.10	0.0003	0.0003	-0.0000	2
0.24	0.0046	0.0046	-0.0001	2
0.38	0.0178	0.0178	0.0000	3
0.52	0.0446	0.0446	-0.0000	4
0.66	0.0888	0.0887	-0.0000	6
0.80	0.1533	0.1533	-0.0001	8

Тема 2. Задачи целочисленной арифметики

В десятичной системе счисления любое натуральное число может быть записано в виде суммы разрядных слагаемых:

$$647 = 6 \cdot 100 + 4 \cdot 10 + 7 = 6 \cdot 10^2 + 4 \cdot 10^1 + 7 \cdot 10^0.$$

Представление числа в виде массива цифр используется для больших чисел, если необходимо сохранить все значащие цифры числа. Расположить цифры числа в массиве можно двумя способами:

1) разряды чисел нумеруются справа налево, а число читается слева направо. Такой порядок, при котором цифра старшего разряда является первым элементом массива, называется *прямым*. Например: (6, 4, 7);

2) разряды чисел нумеруются слева направо, а число читается справа налево. Такой порядок, при котором цифра младшего разряда является первым элементом массива, называется *обратным*. Например: (7, 4, 6).

В таблице 2.1 представлены алгоритмы выделения цифр числа в прямом и обратном порядке.

Таблица 2.1 – Алгоритмы выделения цифр числа в прямом и обратном порядке

Алгоритм выделения цифр числа в обратном порядке	Алгоритм выделения цифр числа в прямом порядке
<pre>k:=0; while N>=1 do begin k:=k+1; b[k]:=N mod 10 N:=N div 10 end;</pre>	<pre>k:=0; { количество цифр в числе } L:=N; while L>=1 do begin L:=L div 10; k:=k+1; end; { выделение цифр числа } m:=k; while N>=1 do begin b[k]:=N mod 10; N:=N div 10; k:=k-1; end;</pre>

Получение числа из массива цифр. Целое число $a = \overline{a_1 a_2 a_3 \dots a_n}$, где a_1 – цифра старшего разряда, a_i – цифра старшего i -го разряда, можно представить в виде:

$$\begin{aligned} a &= a_1 \cdot 10^{n-1} + a_2 \cdot 10^{n-2} + a_3 \cdot 10^{n-3} + \dots + a_{n-1} \cdot 10 + a_n = \\ &= 10(a_1 \cdot 10^{n-2} + a_2 \cdot 10^{n-3} + a_3 \cdot 10^{n-4} + \dots + a_{n-1}) + a_n = \\ &= 10(10(a_1 \cdot 10^{n-3} + a_2 \cdot 10^{n-4} + a_3 \cdot 10^{n-5} + \dots + a_{n-2}) + a_{n-1}) + a_n = \\ &= 10(10 \dots (10 \cdot (a_1 \cdot 10 + a_2) + a_3) + \dots + a_{n-1}) + a_n. \end{aligned}$$

Для формирования числа a , представленного в виде массива цифр, можно воспользоваться алгоритмом схемы Горнера, который состоит в следующем:

1) первую цифру числа умножить на 10 и к произведению прибавить вторую цифру числа;

2) полученную сумму умножить на 10 и прибавить следующую цифру числа и т. д.

{Алгоритм схемы Горнера. Цифры числа расположены в прямом порядке}

$a := 0;$

for $i := 1$ **to** N **do**

$a := a * 10 + b[i];$

{Алгоритм схемы Горнера. Цифры числа расположены в обратном порядке}

$a := 0; r := 1;$

for $i := 1$ **to** N **do**

begin

$a := a + b[i] * r;$

$r := r * 10;$

end;

Практическое задание

Разработать программу для реализации указанных действий, согласно варианту, над целыми числами без знака типа `Longint`. Написать два варианта программы:

1) с использованием массивов, т. е. число представить в виде массива цифр, произвести необходимые действия, из массива цифр получить новое число и результат вывести в виде нового числа;

2) без использования массивов.

Варианты

1. Уменьшить каждую цифру числа на наименьшую ($375 \rightarrow 042$).
2. Увеличить каждую цифру числа, кроме 9, на 1 ($6\ 313 \rightarrow 7\ 424$).
3. Удалить каждую четвертую цифру числа ($457\ 634 \rightarrow 45\ 634$).
4. Найти скалярное произведение цифр двух чисел ($231, 23 \rightarrow 9$).
5. Найти сумму четных цифр числа ($3\ 427\ 654 \rightarrow 16$).
6. Найти сумму нечетных цифр числа ($3\ 427\ 654 \rightarrow 15$).
7. Удалить из числа цифру 3, не изменяя порядок цифр в числе ($35\ 452\ 557 \rightarrow 5\ 452\ 557$).
8. Удалить из числа цифру 5, не изменяя порядок цифр в числе ($35\ 452\ 557 \rightarrow 3\ 427$).
9. Из двух чисел получить новое число путем их соединения ($331, 23 \rightarrow 33\ 123$).
10. Циклически сдвинуть цифры на один разряд влево ($23\ 176 \rightarrow 31\ 762$).
11. Удалить из числа цифры 3, 4, 5 ($531\ 423 \rightarrow 12$).
12. Найти произведение четных цифр числа ($3\ 427\ 654 \rightarrow 192$).
13. Четные цифры числа, большие 3, уменьшить на 2 ($427\ 654 \rightarrow 227\ 452$).
14. Из двух чисел получить новое путем замены четных цифр первого числа на наибольшую цифру второго числа ($231, 73 \rightarrow 731$).
15. Найти произведение нечетных цифр числа ($3\ 427\ 654 \rightarrow 105$).
16. Нечетные цифры числа, большие 5, уменьшить на 3 ($7\ 654 \rightarrow 4\ 654$).
17. Удалить четные цифры числа ($231\ 625 \rightarrow 315$).
18. Удалить нечетные цифры числа ($231\ 625 \rightarrow 262$).
19. Удалить из числа цифры, кратные 3 ($2\ 319\ 663 \rightarrow 21$).
20. Записать цифры числа в обратном порядке ($2\ 313 \rightarrow 3\ 132$).
21. Удалить из числа цифры, кратные 4 ($4\ 345\ 863 \rightarrow 3\ 563$).
22. Удалить цифру 0 из числа ($230\ 107 \rightarrow 2\ 317$).
23. Вставить после каждой цифры 0 ($231\ 457 \rightarrow 2\ 030\ 405\ 070$).
24. Уменьшить каждую цифру числа на 2, кроме нуля и единицы ($56\ 310 \rightarrow 34\ 110$).
25. Удвоить цифры числа, меньшие 4 ($2\ 316\ 323 \rightarrow 4\ 626\ 646$).
26. Увеличить каждую цифру числа на 1, кроме 9 ($390 \rightarrow 491$).
27. Удалить цифры, кратные 5 ($2\ 856\ 423 \rightarrow 286\ 423$).
28. Уменьшить на 2 цифры, кратные 4 ($2\ 856\ 423 \rightarrow 2\ 656\ 223$).
29. Поменять местами первую и последнюю цифры числа ($231 \rightarrow 132$).
30. Удалить каждую вторую цифру числа ($231\ 457\ 634 \rightarrow 21\ 564$).

Пример выполнения задания

Уменьшить каждую цифру числа, кроме нуля, на единицу (82 011 → 71 000). В таблице 2.2 представлены примеры программ с использованием и без использования массивов.

Таблица 2.2. – Примеры программ обработки целого числа с использованием и без использования массивов

Пример программы с использованием массивов	Пример программы без использования массивов
<pre> program uvp2_1; var a,s,r:LongInt; b,c: Array[1..10] of 0..9; i,k: Byte; begin Write('Введите число->'); ReadLn(a); k:=0; {кол-во цифр в числе} while a>=1 do begin k:=k+1; b[k]:=a mod 10; a:=a div 10; end; for i:=1 to k do if b[i]>0 then c[i]:=b[i]-1 else c[i]:=b[i]; s:=0; r:=1; {степень десятки} for i:=1 to k do begin s:=s+c[i]*r; r:=r*10; end; WriteLn('Новое число->',s); end. </pre>	<pre> program uvp2_2; var a,b,c,k:LongInt; begin Write('Введите число->'); ReadLn(a); k:=1; b:=0; {начальное значение} {пока a>=1 выделяем очередную цифру числа и обрабатываем ее} while a>=1 do begin c:=a mod 10; {выделяем последнюю цифру} if c<>0 then begin c:=c-1; b:=b+c*k; end {добавление к новому числу цифры в соответствующий разряд} end; k:=k*10; a:=a div 10; end; WriteLn('Новое число->',b) end. </pre>

Результат работы программы

Введите число->82011
Новое число->71000

Тема 3. Методы сортировок

Сортировка – это процесс расстановки элементов «в некотором порядке». Различают 3 группы простых сортировок:

1. Методы, основанные на выборе: линейный выбор, линейный выбор с обменом, линейный выбор с подсчетом.
2. Сортировка обменом: парный обмен, метод стандартного обмена (пузырька), метод просеивания.
3. Метод линейной вставки.

В таблице 3.1 представлен код программ сортировки вектора различными методами.

Таблица 3.1 – Сортировка вектора разными методами

Линейный выбор	Линейный выбор с обменом
<pre> {Исходный вектор A} {Полученный вектор B} for i:=1 to n do begin min:=A[i]; i_min:=i; for j:=1 to n do if A[j]<min then begin min:=A[j]; i_min:=j; end; B[i]:=min; A[i_min]:=99; end; </pre>	<pre> for i:=1 to n-1 do begin min:=A[i]; i_min:=i; for j:=i+1 to n do if A[j]<min then begin min:=A[j]; i_min:=j; end; r:=A[i_min]; A[i_min]:=A[i]; A[i]:=r; end; </pre>
Линейный выбор с подсчетом	Парный обмен
<pre> for i:=1 to n do {инициируем вектор счетчика} S[i]:=1; for i:=1 to n-1 do {формируем вектор счетчиков} begin k:=0; {количество меньших элементов для i-го элемента} for j:=i+1 to n do if A[i]<A[j] then </pre>	<pre> repeat k:=0; {k – количество перестановок} for j:=1 to 2 do {четный и нечетный просмотр} begin i:=j; while (i<n) do begin if A[i]>A[i+1] then begin </pre>

<pre> S[j]:=S[j]+1 {в счетчики больших элементов добавляем единицы} else k:=k+1; S[i]:=S[i]+k; end; {формируем вектор B в соответствии со значениями вектора счетчика} for i:=1 to n do begin r:=S[i]; B[r]:=A[i]; end;</pre>	<pre> r:=A[i]; A[i]:=A[i+1]; A[i+1]:=r; k:=k+1; end; i:=i+2; end; end; until k=0;</pre>
<p>Метод линейной вставки</p> <pre> for i:=1 to n do begin t:=a[i]; j:=i-1; for k:=j downto 1 do if t<a[k] then begin a[j+1]:=a[j]; j:=j-1; end else break; end; a[j+1]:=t; end;</pre>	<p>Метод просеивания</p> <pre> {восходящий просмотр} for i:=1 to n-1 do if A[i]>A[i+1] then begin r:=A[i]; A[i]:=A[i+1]; A[i+1]:=r; end; {нисходящий просмотр с момента перестановки} for k:=i downto 2 do if A[k]<A[k-1] then begin r:=A[k]; A[k]:=A[k-1]; A[k-1]:=r; end else break; end; end;</pre>
<p>Метод стандартного обмена (пузырька)</p> <pre> for i:=n-1 downto 1 do for j:=1 to i do if A[j]>A[j+1] then begin r:=A[j]; A[j]:=A[j+1]; A[j+1]:=r; end; end;</pre>	

Практическое задание

Разработать программы сортировки целочисленного вектора и вещественной матрицы согласно варианту. Программы оформить с использованием подпрограмм с передачей параметров. Все переменные должны быть описаны локально в процедурах. Программы должны содержать простейшее текстовое меню.

Варианты сортировки элементов вектора

1. Если максимальный элемент вектора является нечетным числом, то отсортируйте элементы вектора по возрастанию методом просеивания.

2. Если минимальный элемент вектора является четным числом, то отсортируйте элементы вектора по убыванию методом парного обмена.

3. Если максимальный элемент вектора является первым элементом вектора, то отсортируйте элементы вектора по возрастанию методом линейного выбора.

4. Если минимальный элемент вектора является последним элементом вектора, то отсортируйте элементы вектора по убыванию методом линейного выбора с обменом.

5. Если сумма всех элементов вектора больше нуля, то отсортируйте элементы вектора по возрастанию методом стандартного обмена.

6. Если сумма всех элементов вектора меньше нуля, то отсортируйте элементы вектора по убыванию методом линейного выбора с подсчетом.

7. Если количество положительных элементов вектора больше количества отрицательных элементов вектора, то отсортируйте элементы вектора по возрастанию методом линейной вставки.

8. Если количество отрицательных элементов вектора меньше 5, то отсортируйте элементы вектора по убыванию методом просеивания.

9. Если все элементы вектора четные числа, то отсортируйте элементы вектора по возрастанию методом линейной вставки.

10. Если все элементы вектора нечетные числа, то отсортируйте элементы вектора по убыванию методом линейного выбора.

11. Если произведение всех элементов вектора больше нуля, то отсортируйте элементы вектора по возрастанию методом линейного выбора с обменом.

12. Если в векторе есть хотя бы один отрицательный элемент, то отсортируйте элементы вектора по убыванию методом парного обмена.

13. Если сумма четных элементов вектора меньше последнего элемента вектора, то отсортируйте элементы вектора по возрастанию методом линейного выбора с подсчетом.

14. Если сумма нечетных элементов вектора больше первого элемента вектора, то отсортируйте элементы вектора по убыванию методом просеивания.

15. Если среднее арифметическое значение первого и последнего элемента вектора меньше 8, то отсортируйте элементы вектора по возрастанию методом стандартного обмена.

16. Если среднее арифметическое значение второго и предпоследнего элементов вектора больше 6, то отсортируйте элементы вектора по убыванию методом линейного выбора.

17. Если среднее арифметическое значение положительных элементов вектора меньше 7, то отсортируйте элементы вектора по возрастанию методом линейного выбора с обменом.

18. Если среднее арифметическое значение отрицательных элементов вектора больше 5, то отсортируйте элементы вектора по убыванию методом линейного выбора с подсчетом.

19. Если последний элемент вектора четное число, то отсортируйте элементы вектора по возрастанию методом парного обмена.

20. Если первый элемент вектора нечетное число, то отсортируйте элементы вектора по убыванию методом стандартного обмена.

21. Если количество максимальных элементов вектора равно 3, то отсортируйте элементы вектора по возрастанию методом линейной вставки.

22. Если количество минимальных элементов вектора равно 2, то отсортируйте элементы вектора по убыванию методом линейного выбора.

23. Если в векторе есть 3 элемента, принадлежащие интервалу $[3, 7]$, то отсортируйте элементы вектора по возрастанию методом просеивания.

24. Если в векторе нет элементов, принадлежащих интервалу $[4, 8]$, то отсортируйте элементы вектора по убыванию методом парного обмена.

25. Если все четные элементы вектора стоят на нечетных местах, то отсортируйте элементы вектора по возрастанию методом линейного выбора с обменом.

26. Если все нечетные элементы вектора стоят на четных местах, то отсортируйте элементы вектора по убыванию методом стандартного обмена.

27. Если все отрицательные элементы вектора нечетные числа, то отсортируйте элементы вектора по убыванию методом парного обмена.

28. Если все положительные элементы вектора четные числа, то отсортируйте элементы вектора по возрастанию методом линейной вставки.

29. Если максимальный элемент вектора четное число, то отсортируйте элементы вектора по возрастанию методом линейного выбора.

30. Если минимальный элемент вектора нечетное число, то отсортируйте элементы вектора по убыванию методом линейного выбора с обменом.

Варианты сортировки элементов матрицы

1. Если среднее арифметическое максимального и минимального элементов первого столбца равно среднему арифметическому максимального и минимального элементов всей матрицы, то отсортируйте элементы столбцов матрицы по убыванию методом линейного выбора.

2. Если максимальный элемент матрицы встречается более двух раз, то отсортируйте строки матрицы по возрастанию методом линейного выбора с обменом.

3. Если максимальный элемент матрицы встречается более трех раз, то отсортируйте элементы строк матрицы по возрастанию методом парного обмена.

4. Если минимальный элемент первой строки равен максимальному элементу последнего столбца, то отсортируйте элементы строки матрицы по убыванию методом стандартного обмена.

5. Если все отрицательные элементы матрицы расположены на главной диагонали, то отсортируйте элементы строк матрицы по возрастанию методом просеивания.

6. Если сумма минимальных элементов каждой строки больше суммы минимальных элементов каждого столбца, то отсортируйте элементы столбцов матрицы по возрастанию методом линейной вставки.

7. Если количество ненулевых элементов матрицы равно модулю минимального элемента матрицы, то отсортируйте элементы столбцов матрицы по возрастанию методом линейного выбора с подсчетом.

8. Если среднее арифметическое максимального и минимального элементов первой строки равно среднему арифметическому максимального и минимального элементов последнего столбца, то отсортируйте элементы столбцов матрицы по убыванию методом линейного выбора.

9. Если среднее арифметическое элементов первой строки превышает среднее арифметическое элементов всей матрицы, то отсор-

тируйте столбцы матрицы по возрастанию максимальных элементов столбцов методом парного обмена.

10. Если элементы столбцов расположены по убыванию, то отсортируйте элементы строк матрицы по возрастанию методом просеивания.

11. Если максимальный элемент каждого столбца меньше максимального элемента каждой строки, то отсортируйте столбцы матрицы по возрастанию минимальных элементов столбцов методом просеивания.

12. Если максимальный элемент матрицы принадлежит интервалу $[4, 15]$, то отсортируйте столбцы матрицы по возрастанию сумм отрицательных элементов столбцов методом линейного выбора.

13. Если минимальный элемент матрицы встречается более 3 раз и один из них находится на главной диагонали, то отсортируйте столбцы матрицы по возрастанию максимальных элементов столбцов методом парного обмена.

14. Если максимальный элемент матрицы меньше удвоенного среднего арифметического элементов, расположенных под главной диагональю, то отсортируйте строки матрицы по невозрастанию сумм элементов строк методом стандартного обмена.

15. Если максимальный элемент матрицы меньше удвоенного среднего арифметического элементов, расположенных над главной диагональю, то отсортируйте столбцы матрицы по возрастанию сумм положительных элементов столбцов методом линейного выбора с обменом.

16. Если минимальный элемент матрицы встречается более трех раз, то отсортируйте строки матрицы по возрастанию максимальных элементов столбцов методом стандартного обмена.

17. Если минимальный элемент матрицы расположен над главной диагональю, а максимальный элемент под главной диагональю, то отсортируйте элементы столбцов матрицы по убыванию методом стандартного обмена.

18. Если количество нулевых элементов матрицы больше количества повторений минимального элемента матрицы, то отсортируйте столбцы матрицы по невозрастанию максимальных элементов столбцов методом парного обмена.

19. Если количество положительных элементов над главной диагональю равно количеству отрицательных элементов под главной диагональю, то отсортируйте столбцы матрицы по возрастанию максимальных элементов столбцов методом просеивания.

20. Если среди элементов матрицы есть элементы, принадлежащие интервалу $[-3, 5]$, то отсортируйте столбцы матрицы по невозрастанию сумм элементов столбцов методом линейного выбора.

21. Если максимальный и минимальный элементы не принадлежат интервалу $[-4, 15]$, то отсортируйте столбцы матрицы по возрастанию сумм положительных элементов столбцов методом парного обмена.

22. Если максимальный элемент матрицы встречается 3 раза, причем один расположен над главной диагональю, второй – под главной диагональю, а третий – на главной диагонали, то отсортируйте столбцы матрицы по возрастанию первых элементов столбцов методом стандартного обмена.

23. Если сумма элементов первой строки больше произведения элементов первого столбца, то отсортируйте столбцы матрицы по возрастанию максимальных элементов столбцов методом линейного выбора с обменом.

24. Если сумма максимального и минимального элементов каждого последующего столбца больше предыдущего, то отсортируйте строки матрицы по возрастанию последних элементов строк методом линейного выбора.

25. Если элементы второй строки образуют невозрастающую последовательность и наибольший элемент находится в этой строке, то отсортируйте столбцы матрицы по возрастанию максимальных элементов столбцов методом стандартного обмена.

26. Если матрица является магическим квадратом, т. е. сумма элементов в каждой строке равна сумме элементов каждого столбца и равна сумме элементов на главной и побочной диагоналях, то отсортируйте элементы строк матрицы по убыванию методом линейного выбора.

27. Если максимальный четный элемент матрицы находится на главной диагонали, то отсортируйте элементы столбцов матрицы по возрастанию методом линейного выбора с обменом.

28. Если количество четных элементов матрицы выше главной диагонали больше количества нечетных элементов матрицы ниже побочной диагонали, то отсортируйте элементы каждой строки матрицы по возрастанию методом линейного выбора с подсчетом.

29. Если сумма четных элементов главной диагонали меньше суммы нечетных элементов побочной диагонали, то отсортируйте элементы строк матрицы по убыванию методом просеивания.

30. Если максимальный элемент выше главной диагонали больше максимального элемента ниже главной диагонали, то отсортируйте элементы строк матрицы по убыванию методом парного обмена.

Пример выполнения задания

Дана квадратная действительная матрица. Если минимальный элемент матрицы встречается более трех раз и один из них находится на главной диагонали, то упорядочить столбцы матрицы по возрастанию максимальных элементов столбцов методом парного обмена.

Структура программы:

```
program uvp3;
uses crt;
type
  mas=Array [1..5,1..5] of Real;
var
  a          :mas;
  m, rejim   :Byte;

{Процедура рисования рамки и установки окна}
procedure Frame (x1,y1,x2,y2,fon,cvet:Integer);
const
  a=#186; b=#187;
  c=#188; d=#200;
  e=#201; f=#205;
var
  i,j:Integer;
begin
  textColor(cvet);
  gotoXY(x1,y1);
  Write(e);
  for i:=x1+1 to x2-1 do
    Write(f);
  Write(b);
  for i:=y1+1 to y2-1 do
    begin
      gotoXY(x1,i);
      write(a);
      gotoXY(x2,i);
      Write(a);
    end;
  gotoXY(x1,y2);
  Write(d);
  window(x1,y1,x2,y2+1);
  gotoXY(2,y2-y1+1);
  for i:=x1+1 to x2-1 do
    Write(f);
```

```

Write(c);
window(x1+1,y1+1,x2-1,y2-1);
textColor(cvet);
textBackground(fon);
clrscr;
end;

procedure Vvod(var c:mas; var n:Integer);
begin
  {Ввод массива}
end;

procedure Vivod(c:mas; n:Integer);
begin
  {Вывод массива}
end;

procedure Sort(var c:mas; n:Integer);
begin
  {Сортировка массива}
end;

{Основная программа}
begin
  clrscr;
  while true do
    begin
      window(1,1,80,25);
      Frame(1,1,30,10,blue,yellow);
      WriteLn(' Меню ');
      WriteLn(' 1 - Ввод матрицы ');
      WriteLn(' 2 - Исходная матрица ');
      WriteLn(' 3 - Сортировка ');
      WriteLn(' 4 - Полученная матрица ');
      WriteLn(' 5 - Выход ');
      Write(' Выберите режим --> ');
      Read(rejim);
      window(1,1,80,25);
      case rejim of
        1 : begin
              Frame(35,1,70,10,blue,yellow);
              Vvod(a,m);
            end;
        2 : begin
              Frame(35,1,70,10,blue,yellow);

```

```

        WriteLn(' Исходная матрица');
        Vivod(a,m);
    end;
3 : begin
    Frame(35,11,70,21,blue,yellow);
    WriteLn('Сортировка');
    Sort(a,m);
    end;
4 : begin
    Frame(1,11,30,21,blue,yellow);
    WriteLn(' Полученная матрица');
    Vivod(a,m);
    end;
5 : Exit;
end;
end;
end.

```

Результат работы программы

Меню 1 - Ввод матрицы 2 - Исходная матрица 3 - Сортировка 4 - Полученная матрица 5 - Выход Выберите режим -->	Исходная матрица 2.0 1.0 5.0 1.0 1.0 1.0 3.0 6.0 2.0
Полученная матрица 2.0 5.0 1.0 1.0 1.0 1.0 3.0 2.0 6.0	Сортировка Кол-во min эл-в = 4 Один min эл-т на главной диагонали Матрица отсортирована

Тема 4. Модульная структура программы

Модуль – это автономно компилируемая программная единица, включающая в себя различные компоненты раздела описаний (типы, константы, переменные, процедуры и функции) и, возможно, некоторые исполняемые операторы иницилирующей части. Модули используются для разработки библиотек прикладных программ. Важная особенность модулей заключается в том, что компилятор языка Pascal размещает их программный код в отдельном сегменте памяти. Максимальная длина сегмента не может превышать 64 КВ, однако количество одновременно используемых модулей ограничивается лишь доступной памятью.

Модуль имеет следующую структуру:

UNIT имя_модуля; { \$директивы компилятора }	{заголовок модуля}
INTERFACE uses const type var procedure имя(параметры) ; function имя(параметры) :тип;	{интерфейсная часть} {имена подключаемых модулей} {раздел описания констант} {раздел описания типов} {раздел описания переменных} {заголовки процедур} {заголовки функций}
IMPLEMENTATION uses const type var procedure имя; begin ... end; function имя; begin ... end;	{исполняемая часть} {тело процедуры} {тело функции}
begin ... end.	{инициирующая часть}

Имя модуля служит для связи модуля с основной программой и другими модулями. Имя модуля должно совпадать с именем дискового файла, в котором находится исходный текст программы.

В секции `INTERFACE` описываются *глобальные* данные, заголовки процедур и функций, доступные основной программе и другим модулям.

В секции `IMPLEMENTATION` реализуется программный код глобальных процедур и функций и описываются локальные данные, процедуры и функции, недоступные основной программе и другим модулям.

Иницилирующая часть завершает модуль. Она может отсутствовать или быть пустой. В иницилирующей части размещаются исполняемые операторы, содержащие некоторый фрагмент программы. Эти операторы выполняются до передачи управления основной программе и обычно используются для подготовки её работы.

При компиляции модулей необходимо установить режим компиляции `Build`. После выполнения компиляции на диске создается файл с расширением `.tpu`, доступный для использования без каких-либо дополнительных описаний. Этот файл можно подключать в любую программу следующим образом:

```
uses имя модуля;
```

Практическое задание

Задачи из темы 3 реализовать с использованием модулей. Все процедуры и функции должны находиться в модуле и подключаться к основной программе. Написать две программы с передачей параметров для обработки вектора и матрицы.

Пример выполнения задания

Структура модуля:

```
unit MYBIBL;
```

```
INTERFACE
```

```
uses CRT;
```

```
type
```

```
  mas=Array[1..10,1..10] of Real;
```

```
procedure Frame(x1,y1,x2,y2,fon,cvet:Integer);
```

```
procedure Vvod(var B:mas; var n:Integer);
```

```
procedure Vivod(b:mas; n:Integer);
procedure Sort(var c:mas; n:Integer);
```

IMPLEMENTATION

```
procedure Frame(x1,y1,x2,y2,fon, cvet:Integer);
begin
  {процедура черчения рамок заданного цвета и фона
  (описана в примере из темы 3)}
end;
```

```
procedure Vvod(var B:mas; var n:Integer);
begin
  {процедура ввода матрицы}
end;
```

```
procedure Vivod(B:mas; n:Integer);
begin
  {процедура вывода матрицы}
end;
```

```
procedure Sort(var c:mas; n:Integer);
begin
  {Сортировка матрицы}
end;
```

```
Begin
End.
```

Код основной программы:

```
program menu;
uses CRT, Mybibl;
var
      a :mas;
      rejim, m :Integer;
begin
  clrscr;
  while true do
  begin
    window(1,1,80,25);
    Frame(1,1,30,10,blue,yellow);
    WriteLn(' Меню ');
    WriteLn(' 1 - Ввод матрицы ');
```

```

WriteLn(' 2 - Исходная матрица ');
WriteLn(' 3 - Сортировка ');
WriteLn(' 4 - Полученная матрица');
WriteLn(' 5 - Выход');
Write (' Выберите режим -->');
Read(rejim);
window(1,1,80,25);
case rejim of
1: begin
    Frame(35,1,70,10,blue,yellow);
    Vvod(a,m);
    end;
2: begin
    Frame(35,1,70,10,blue,yellow);
    WriteLn('Исходная матрица');
    Vivod(a,m);
    end;
3: begin
    Frame (35,11,70,21,blue,yellow);
    WriteLn('Обработка');
    Sort(a,m);
    end;
4: begin
    Frame(1,11,30,21,blue,yellow);
    WriteLn('Полученная матрица');
    Vivod(a,m);
    end;
5: Exit
end;
end;
end.

```

Тема 5. Файлы записей

Под файлом понимается либо поименованная область внешней памяти компьютера, либо логическое устройство – потенциальный источник или приемник информации. Любой программе доступны два предварительно объявленных файла со стандартными файловыми переменными: `INPUT` – для чтения данных с клавиатуры и `OUTPUT` – для вывода на экран.

Для работы с файлом необходимо описать файловую переменную одним из трех способов:

type

```
файловая_переменная=File of тип_компонент;  
файловая_переменная=Text;  
файловая_переменная=File;
```

которым соответствуют три вида файлов: типизированный файл, текстовый файл, нетипизированный файл.

Файлы записей это типизированные файлы, компонентами которых являются записи. Для работы с файлами записей необходимо описать переменную доступа к записи и файловую переменную, указав в качестве типа компонент описанную ранее переменную доступа к записи.

Формат описания файла записей:

type

```
имя_типа_записи=record  
    идентификатор_поля:тип;  
    ...  
    идентификатор_поля:тип  
end;
```

var

```
переменная_доступа_к_записи:имя_типа_записи;  
файловая_переменная:File of тип_компонент;
```

Пример описания файла, компонентами которого являются записи, содержащие следующие сведения о студентах: фамилия, номер группы, результаты сдачи трех экзаменов (алгебра, программирование, история):

type

```
Sved=record  
    Fio:String[50];
```

```

        Nom_gr:String[10];
        Algebra:0..10;
        Progr:0..10;
        History:0..10;
    end;
var
    Rv:Sved;           {переменная доступа к записи}
    Fv:File of Sved;   {файловая переменная}

```

Доступ к записям файла осуществляется через указатель файла (файловую переменную). В ней хранится текущий номер записи файла. Нумерация записей начинается с нуля.

Процедуры работы с файлами записей

Assign(ф_п, имя_файла) – связать файловую переменную (ф_п) с именем файла.

Rewrite(ф_п) – открыть новый файл для записи.

Reset(ф_п) – открыть существующий файл для чтения. Разрешается обращаться к *типизированным* файлам, открытым процедурой Reset, с помощью процедуры Write для записи информации в файл.

Read(ф_п, переменная_доступа_к_записи) – считывает запись из файла.

Write(ф_п, переменная_доступа_к_записи) – записывает запись в файл.

Seek(файловая_переменная, номер_компоненты) – смещает указатель файла к требуемой записи. Первая запись файла имеет номер ноль.

Close(ф_п) – закрыть файл.

Rename(ф_п, новое_имя) – переименовать файл.

Erase(ф_п) – уничтожить файл.

Flush(ф_п) – очищает внутренний буфер файла и, таким образом, гарантирует сохранность всех последних изменений файла на диске. Процедура игнорируется, если файл был открыт для чтения процедурой Reset.

Функции для работы с файлами записей

EOF(ф_п) – функция возвращает значение true, если указатель файла стоит в конце файла.

`FileSize (ф_п)` – функция возвращает значение типа `LongInt`, которое содержит количество компонент файла.

`FilePos (ф_п)` – функция возвращает значение типа `LongInt`, содержащее порядковый номер записи файла, который будет обрабатываться следующей операцией ввода-вывода.

Порядок работы с файлами записей

Создание файла

1. Связать файловую переменную с именем файла (`Assign`).
2. Открыть новый файл для записи (`Rewrite`).
3. Записать запись в файл (`Write`).
4. Закрыть файл (`Close`).

Использование файла

1. Связать файловую переменную с именем файла (`Assign`).
2. Открыть существующий файл для чтения (`Reset`).
3. Прочитать запись из файла (`Read`).
4. Закрыть файл (`Close`).

Для файлов записей применяются следующие виды корректировок:

1. Расширение файла за счет внесения новых записей.
2. Полная замена содержимого записи.
3. Корректировка значений полей отдельных записей.
4. Удаление записей из файла.

Расширения файла за счет внесения новых записей

1. Связать файловую переменную с именем файла (`Assign`).
2. Открыть существующий файл для чтения и записи (`Reset`).
3. Установить указатель файла за последней записью
`Seek (ф_п, FileSize (ф_п))`
4. Записать компонент в файл (`Write`)
5. Закрыть файл (`Close`).

Замена содержимого записи

1. Связать файловую переменную с именем файла (`Assign`).
2. Открыть существующий файл для чтения и записи (`Reset`).
3. Установить указатель файла перед записью с нужным номером
`Seek (ф_п, номер_записи)`
4. Прочитать запись из файла (`Read`).

5. Заменить содержимое записи.
6. Установить указатель файла перед записью с нужным номером `Seek (ф_п, номер_записи)`
7. Записать запись в файл (`Write`)
8. Закрывать файл (`Close`).

Корректировка значений полей отдельных записей

1. Связать файловую переменную с именем файла (`Assign`).
2. Открыть существующий файл для чтения и записи (`Reset`).
3. Установить указатель файла перед записью с нужным номером `Seek (ф_п, номер_записи)`
4. Прочитать запись из файла (`Read`).
5. Заменить содержимое отдельных полей записи.
6. Установить указатель файла перед записью с нужным номером `Seek (ф_п, номер_записи)`
7. Записать запись в файл (`Write`)
8. Закрывать файл (`Close`).

Удаление записи из файла

1. Связать файловую переменную с именем файла (`Assign`).
2. Открыть существующий файл для чтения (`Reset`).
3. Ввести номер удаляемой записи (`nom`).
4. Связать новую файловую переменную с именем временного файла (`Assign`).
5. Открыть временный файл для записи (`Rewrite`).
6. Пока не конец исходного файла (`while not EOF(ф_п) do ...`), считываем последовательно записи из исходного файла (`Read`) и записываем их во временный файл (`Write`), если номер текущей записи не совпадает с номером удаляемой записи.
7. Закрывать исходный файл (`Close`).
8. Закрывать временный файл (`Close`).
9. Уничтожить исходный файл (`Erase`).
10. Переименовать временный файл в исходный файл (`Rename`).
11. Уничтожить временный файл (`Erase`).

Практическое задание

Создать файл записей. Запись содержит сведения согласно варианту. Исходные данные и результаты выводить в виде таблицы. Программу оформить с помощью текстового меню, включающего создание

файла, печать файла, обработку файла, расширение файла, замену всей записи, замену отдельных полей записи, удаление записи, выход.

Варианты

1. Запись содержит сведения о проданных товарах в магазинах города. Структура записи: ФИО покупателя, номер магазина, наименование товара, количество единиц товара, цена одной единицы, общая сумма (вычислить), дата покупки. Обработка: распечатать список лиц, купивших товар на общую сумму более 1000 руб. на заданную дату.

2. Запись содержит сведения о банках. Структура записи: название банка, уставной фонд, годовой оборот, валютные ресурсы. Обработка: распечатать список банков, валютные ресурсы которых составляют более 1 млн. долларов.

3. Запись содержит сведения о странах. Структура записи: название страны, население, площадь, государственный язык, плотность населения (вычислить). Обработка: распечатать сведения о стране, государственный язык которой английский.

4. Запись содержит сведения о студентах университета. Структура записи: факультет, отделение, количество студентов, количество отличников, количество отчисленных, процент успеваемости (вычислить). Обработка: распечатать сведения о факультетах со 100 процентной успеваемостью.

5. Запись содержит сведения о сдаче лабораторных работ. Структура записи: ФИО студента, группа, лабораторная работа 1, лабораторная работа 2, лабораторная работа 3. Если лабораторная работа сдана, то в графу данной лабораторной работы заносится «1», если не сдана – «0». Обработка: распечатать список студентов, сдавших все лабораторные работы.

6. Запись содержит сведения о реках. Структура записи: название реки, площадь бассейна, название самого большого притока, длина реки. Обработка: распечатать сведения о реках длиной более 3 км.

7. Запись содержит сведения о производстве напитков. Структура записи: название завода, производство соков, производство газированных напитков, производство минеральной воды, производство кваса. Обработка: распечатать сведения о заводах, производящих продукцию всех видов.

8. Запись содержит сведения о продаже автомобилей. Структура записи: название магазина, марка автомобиля, стоимость, количество. Обработка: распечатать сведения о количестве проданных автомобилей определенной марки и их общей стоимости.

9. Запись содержит сведения о физической подготовке студентов. Структура записи: ФИО студента, факультет, группа, легкая атлетика (0–10), волейбол (0–10), баскетбол (0–10), плавание (0–10). Баллы выставлять в зависимости от степени подготовки по каждому виду спорта. Обработка: распечатать список студентов, имеющих отличную физическую подготовку (сумма баллов более 30).

10. Запись содержит сведения о заявках на материалы. Структура записи: дата заявки, наименование товара (материала), код единицы измерения, количество, цена за единицу, общая стоимость (вычислить). Обработка: распечатать заявку на материалы на заданную дату.

11. Запись содержит сведения о сдаче экзаменов. Структура записи: номер группы, ФИО, оценки сдачи четырех экзаменов. Обработка: распечатать сведения о студентах, успешно сдавших сессию.

12. Запись содержит сведения о происшествиях в городе. Структура записи: номер, тип происшествия, дата происшествия, место происшествия. Обработка: распечатать список происшествий за заданный период.

13. Запись содержит сведения о курсе валют в банках Республики Беларусь. Структура записи: название банка, американский доллар, немецкая марка, английский фунт, итальянская лира, австралийский доллар. Обработка: распечатать сведения о наименьших курсах по всем валютам.

14. Запись содержит сведения об игрушках. Структура записи: название игрушки, стоимость игрушки, возрастные ограничения (например, для детей от 2 до 5 лет). Обработка: распечатать сведения о названиях игрушек, цена которых не превышает 50 рублей и которые подходят детям до 5 лет.

15. Запись содержит сведения о рабочих нарядах. Структура записи: шифр наряда, дата, номер цеха, табельный номер, количество изготовленных деталей, количество принятых деталей. Обработка: распечатать сведения о бракованных деталях.

16. Запись содержит сведения о результатах сессии. Структура записи: ФИО, факультет, группа, количество несданных зачетов, количество несданных экзаменов. Обработка: распечатать список студентов на отчисление (если не сдано больше трех зачетов и больше двух экзаменов).

17. Запись содержит сведения о наличии товаров в магазинах города. Структура записи: номер магазина, название товара, количество единиц товара, цена одной единицы. Обработка: распечатать список магазинов, имеющих заданный товар на общую сумму более 9 000 руб.

18. Запись содержит сведения о странах. Структура записи: название страны, население, площадь, государственный язык, плотность населения (вычислить). Обработка: распечатать сведения о стране, государственный язык которой испанский.

19. Запись содержит сведения о странах. Структура записи: название страны, население, площадь, государственный язык, плотность населения (вычислить). Обработка: распечатать сведения о стране с наименьшей площадью.

20. Запись содержит сведения о школьниках. Структура записи: класс, количество учеников, количество отличников, количество двоечников, процент успеваемости (вычислить). Обработка: распечатать сведения о школьниках заданного класса.

21. Запись содержит сведения о детях в детском саду. Структура записи: группа, количество детей, ФИО ребенка, возраст, рост. Обработка: распечатать сведения о детях 5 лет, ростом более 110 см.

22. Запись содержит сведения о сотрудниках банка. Структура записи: название банка, ФИО, возраст, должность, образование. Обработка: распечатать фамилии сотрудников, имеющих высшее образование.

23. Запись содержит сведения о знании языков программирования. Структура записи: ФИО студента, группа, языки программирования (Pascal, C, Assembler, C++, Java). Обработка: распечатать список лиц студентов, знающих более трех языков программирования.

24. Запись содержит сведения о знании иностранных языков. Структура записи: ФИО студента, группа, английский (0–5 баллов), французский (0–5 баллов), немецкий (0–5 баллов). Обработка: распечатать список студентов, знающих один из иностранных языков в совершенстве.

25. Запись содержит сведения о простоях в цехах. Структура записи: дата, номер цеха, номер участка, продолжительность простоя. Обработка: распечатать сведения о цехах, продолжительность простоя в которых более 1 часа.

26. Запись содержит сведения о врачах больницы города. Структура записи: название больницы, ФИО врача, специальность, возраст. Обработка: распечатать список хирургов города в возрасте до 40 лет.

27. Запись содержит сведения о больных. Структура записи: название больницы, название отделения, ФИО больного, возраст, диагноз. Обработка: распечатать сведения о больных, заболевших гриппом в возрасте до 30 лет.

28. Запись содержит сведения об озерах. Структура записи: название озера, площадь, название государства, где расположено озеро.

Обработка: распечатать сведения об озерах, площадь которых более 100 кв. км.

29. Запись содержит сведения о жителях, обслуживаемых данной поликлиникой. Структура записи: ФИО, адрес, место работы, дата прохождения последней флюорографии. Обработка: напечатать список жителей, у которых на данный момент просрочена дата флюорографии, т. е. с момента ее прохождения прошло больше года.

30. Запись содержит сведения о машинах. Структура записи: модель, номер, цвет, ФИО владельца, дата последнего техосмотра. Обработка: распечатать данные обо всех машинах, не прошедших техосмотр в текущем году.

Пример выполнения задания

Запись содержит сведения о результатах сдачи сессии студентами. Структура записи: ФИО студента, группа, алгебра, программирование, история. Распечатать список студентов, получивших отличные и хорошие оценки.

Код программы:

```
program uvp5;
uses crt;
const
  filename : string[10] = 'stud.dat';
type
  Sved=record
    Fio:String[50];
    Nom_gr:String[10];
    Algebra:0..10;
    Progr:0..10;
    History:0..10;
  end;
var
  Rv:Sved; {переменная доступа к записи}
  Fv,Fv1:File of Sved; {файловая переменная}
  i,n, rejim: byte;
procedure frame(x1,y1,x2,y2,fon,cvet:integer);
begin
  {Процедура рисования рамки и установки окна.
   Код процедуры описан в теме 3.}
end;
procedure Sozd;
begin
  Assign(Fv,filename);
```

```

Rewrite(Fv);
while true do
  with Rv do
    begin
      clrscr;
      Write('ФИО (Окончание ввода ++) -->');
      ReadLn(Fio);
      if Fio='++' then begin Close(fv); Exit; end;
      Write('Номер группы -->'); ReadLn(Nom_gr);
      WriteLn('Результаты экзаменов:');
      Write('Алгебра -->'); ReadLn(Algebra);
      Write('Программирование -->'); ReadLn(Progr);
      Write('История -->'); ReadLn(History);
      Write(Fv,Rv);
    end;
  end;
procedure Vivod;
begin
  clrscr;
  WriteLn('Сведения о сдачи сессии:');
  WriteLn('


|     |        |         |         |         |
|-----|--------|---------|---------|---------|
| ФИО | Группа | Алгебра | Прогр-е | История |
|-----|--------|---------|---------|---------|


');
  WriteLn('


|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|


');
  Assign(Fv,filename);
  Reset(Fv);
  while not EOF(Fv) do
    with Rv do
      begin
        Read(Fv,Rv);
        WriteLn('


|        |          |           |           |           |
|--------|----------|-----------|-----------|-----------|
| ФИО:11 | Группа:6 | Алгебра:7 | Прогр-е:7 | История:7 |
|--------|----------|-----------|-----------|-----------|


');
      end;
    end;
  WriteLn('


|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|


');
  Close(Fv);
  repeat until KeyPressed;
end;
procedure Obr;
begin
  clrscr;
  WriteLn('Список студентов, хорошо сдавших сессию:');
  WriteLn('


|     |        |         |         |         |
|-----|--------|---------|---------|---------|
| ФИО | Группа | Алгебра | Прогр-е | История |
|-----|--------|---------|---------|---------|


');
  WriteLn('


|  |  |  |  |  |
|--|--|--|--|--|
|  |  |  |  |  |
|--|--|--|--|--|


');
  Assign(Fv,filename);
  Reset(Fv);

```

```

while not EOF(Fv) do
  with Rv do
    begin
      Read(Fv,Rv);
      if (Algebra>6) and (Progr>6) and (History>6)
      then
        WriteLn(' | ',Fio:11,' | ',Nom_gr:6,' | ',
          Algebra:7,' | ',Progr:7,' | ',History:7,' | ');
      end;
    WriteLn(' | _____ | ');
    Close(Fv);
    repeat until KeyPressed;
  end;
procedure Dobavl;
begin
  Assign(Fv,filename); Reset(Fv);
  Seek(Fv,FileSize(Fv));
  while true do
    with Rv do
      begin
        clrscr;
        Write('ФИО (Окончание ввода ++) -->');
        ReadLn(Fio);
        if Fio='++' then begin Close(fv); Exit; end;
        Write('Номер группы -->'); ReadLn(Nom_gr);
        WriteLn('Результаты экзаменов:');
        Write('Алгебра -->'); ReadLn(Algebra);
        Write('Программирование -->'); ReadLn(Progr);
        Write('История -->'); ReadLn(History);
        Write(Fv,Rv);
      end;
    end;
  end;
procedure Zam_zapisi;
begin
  clrscr;
  Assign(Fv,filename); Reset(Fv);
  Write('Введите номер заменяемой записи n=');
  ReadLn(n);
  Seek(Fv,n-1);
  Read(Fv,Rv);
  with Rv do
    begin
      Write('ФИО: ',Fio,' -->'); ReadLn(Fio);
      {аналогично вводятся остальные поля записи}
      ...
    end;
  end;

```

```

    end;
    Seek(Fv,n-1);
    Write(Fv,Rv);
    Close(Fv);
end;
procedure Zam_pol_zapisi;
var
    nom_pol:1..5;
begin
    clrscr;
    Assign(Fv,filename); Reset(Fv);
    Write('Введите номер заменяемой записи n=');
    ReadLn(n);
    Seek(Fv,n-1);
    Read(Fv,Rv);
    with Rv do
    begin
        Write('1 - ФИО:',Fio);
        Write('2 - Номер группы:',Nom_gr);
        WriteLn('Результаты сдачи экзаменов:');
        Write('3 - Алгебра:',Algebra);
        Write('4 - Программирование:',Progr);
        Write('5 - История:',History);
        Write('Укажите номер заменяемого поля -->');
        ReadLn(nom_pol);
        case nom_pol of
            1: begin
                Write('Номер группы -->'); ReadLn(Nom_gr);
            end;
            2: begin
                Write('Алгебра -->'); ReadLn(Algebra);
            end;
            3: begin
                Write('Программирование -->'); ReadLn(Progr);
            end;
            4: begin
                Write('История -->'); ReadLn(History);
            end
        end;
    end;
    Seek(Fv,n-1);
    Write(Fv,Rv);
    Close(Fv);
end;
procedure Udal_zapisi;

```

```

begin
  clrscr;
  Assign(Fv,filename);
  Reset(Fv);
  Write('Введите номер удаляемой записи n = ');
  ReadLn(n);
  Assign(Fv1,'temp.dat');
  Rewrite(fv1);
  i:=0;
  while not EOF(Fv) do
    begin
      Read(fv,rv);
      i:=i+1;
      if i<>n then write(Fv1,Rv);
    end;
  Close(Fv); Close(Fv1);
  Erase(Fv); Rename(Fv1,filename);
end;
Begin
  clrscr;
  while true do
    begin
      window(1,1,80,27);
      frame(1,1,30,12,blue,white);
      WriteLn('Меню:');
      WriteLn('1 - Создание файла');
      WriteLn('2 - Вывод файла');
      WriteLn('3 - Обработка файла');
      WriteLn('4 - Расширение файла');
      WriteLn('5 - Замена записи');
      WriteLn('6 - Замена поля записи');
      WriteLn('7 - Удаление записи');
      WriteLn('8 - Выход');
      Write('Выберите режим ->'); ReadLn(rejim);
      window (1,1,80,27);
      case rejim of
        1: begin
            Frame(35,1,70,10,blue,white); Sozd;
          end;
        2: begin
            Frame(1,14,70,24,blue,white);
            WriteLn('Исходный файл'); Vivod;
          end;
        3: begin
            Frame(1,14,70,24,blue,white);

```

```

        WriteLn('Обработка файла'); Obr;
    end;
4: begin
    Frame(35,1,70,10,blue,white); dobavl;
    end;
5: begin
    Frame(35,1,70,10,blue,white); Zam_zapisi;
    end;
6: begin
    Frame(35,1,70,10,blue,white); Zam_pol_zapisi;
    end;
7: begin
    Frame(35,1,70,10,blue,white); Udal_zapisi;
    end;
8: Exit
    end;
end;
end.

```

Результат работы программы

Меню: 1 - Создание файла 2 - Вывод файла 3 - Обработка файла 4 - Расширение файла 5 - Замена записи 6 - Замена поля записи 7 - Удаление записи 8 - Выход Выберите режим ->3	ФИО (Окончание ввода ++)--> Иванов И.И. Номер группы -->ПМ-12 Результаты экзаменов: Алгебра -->8 Программирование -->9 История -->10
--	--

Сведения о сдаче сессии:				
ФИО	Группа	Алгебра	Прогр-е	История
Иванов И.И.	ПМ-12	8	9	10
Петров С.М.	ПМ-12	7	9	8

Тема 6. Графические возможности языка Pascal

Процедуры работы с графикой

`InitGraph (драйвер, режим, путь)` – инициализирует графический режим. Драйвер – переменная типа `Integer`, определяет тип графического драйвера. Режим – переменная типа `Integer`, задающая режим работы графического адаптера. Путь – выражение типа `String`, содержащее имя файла драйвера и, возможно, маршрут его поиска. Процедура загружает графический драйвер в оперативную память и переводит адаптер в графический режим работы. Обычно при инициализации графики в качестве драйвера указывается значение `Detect` – режим автоопределения типа графического драйвера. В этом случае режим работы графического адаптера определяется по умолчанию.

`Closegraph` – завершает работу адаптера в графическом режиме и восстанавливает текстовый режим работы экрана.

`PutPixel (x, y, цвет)` – выводит заданным цветом точку с координатами (x, y) . Координаты задаются относительно левого верхнего угла окна или, если окно не установлено, относительно левого верхнего угла экрана.

`Line (x1, y1, x2, y2)` – вычерчивает линию с координатами начала $(x1, y1)$ и конца $(x2, y2)$. Линия вычерчивается текущим стилем и текущим цветом.

`LineTo (x, y)` – вычерчивает линию от текущего положения указателя до точки с заданными координатами (x, y) . Линия вычерчивается текущим стилем и текущим цветом.

`LineRel (dx, dy)` – вычерчивает линию от текущего положения указателя до положения, заданного приращением координат (dx, dy) . Линия вычерчивается текущим стилем и текущим цветом.

`SetLineStyle (тип, образец, толщина_линии)` – устанавливает стиль вычерчиваемых линий.

`Rectangle (x1, y1, x2, y2)` – вычерчивает прямоугольник с заданными координатами углов. Здесь $(x1, y1)$ – координаты левого верхнего, $(x2, y2)$ – правого нижнего углов прямоугольника. Прямоугольник вычерчивается с использованием текущего цвета и стиля линий.

`SetColor (цвет)` – устанавливает цвет выводимых линий и символов. Цвет определяется одной из констант, находящихся в модуле `Graph`.

`SetBkColor(цвет)` – устанавливает цвет фона.

`SetFillStyle(тип_заполнения, цвет)` – устанавливает стиль заполнения фигур и цвет. Тип заполнения определяется одной из констант, находящихся в модуле `Graph`.

`FloodFill(x, y, цвет_границы)` – заполняет произвольную замкнутую фигуру, используя текущий стиль заполнения до границы заданного цвета, (x, y) – произвольная точка внутри фигуры.

`Bar(x1, y1, x2, y2)` – заполняет прямоугольную область экрана текущим заполнителем. Здесь $(x1, y1)$ – координаты левого верхнего, $(x2, y2)$ – правого нижнего углов закрашиваемой области.

`OutText(текст)` – выводит текстовую строку, начиная с текущего положения указателя.

`OutTextXY(x, y, текст)` – процедура выводит строку, начиная с позиции (x, y) .

`SetTextStyle(шрифт, направление, размер)` – процедура устанавливает стиль текстового вывода на графический экран. Здесь шрифт – код (номер) шрифта; направление – код направления; размер – код размера шрифта.

Построение графика функции

Система координат графического экрана отличается от декартовой системы координат (ось ou направлена сверху вниз, начало координат находится в левом верхнем углу экрана), рисунок 6.1.

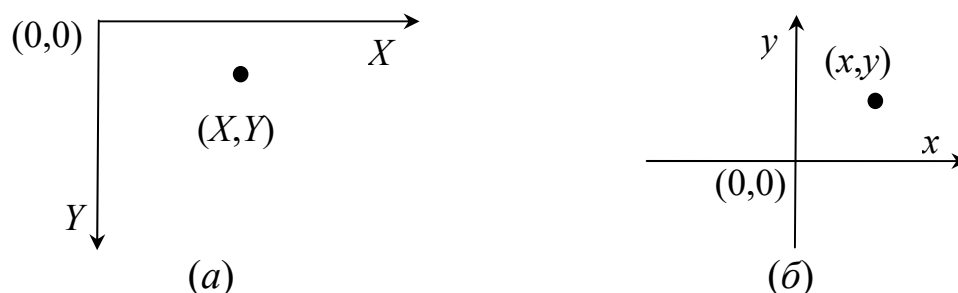


Рисунок 6.1 – Система координат графического экрана (a) и декартова система координат (б)

Для преобразования координат точки (x, y) декартовой системы координат в координаты точки (X, Y) системы координат графического режима можно использовать преобразования:

$$\begin{cases} X = w_x + x \cdot w_m, \\ Y = w_y - y \cdot w_n. \end{cases}$$

где w_m, w_n – масштабные коэффициенты по осям,
 (w_x, w_n) – координаты центра начала декартовой системы координат на графическом экране.

Практическое задание 1

Построить график функции $y = f(x)$ согласно варианту на отрезке $[a, b]$ (a, b – задаются произвольно).

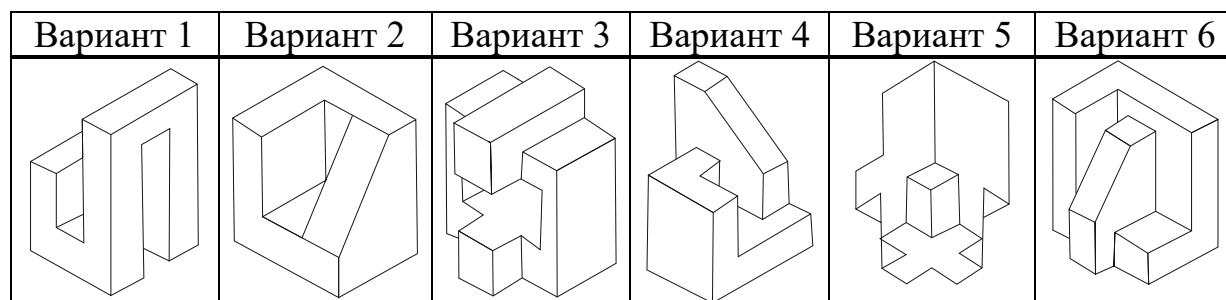
Варианты

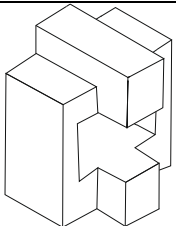
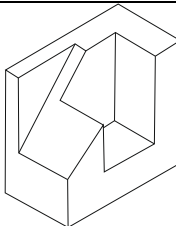
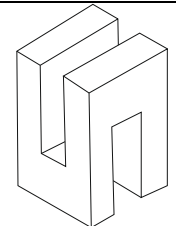
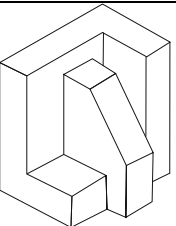
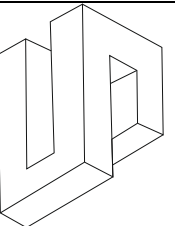
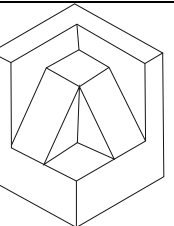
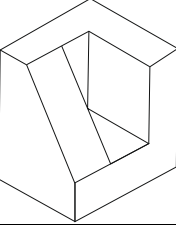
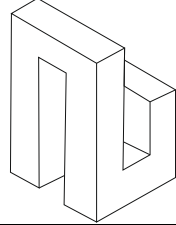
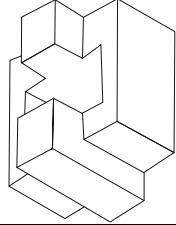
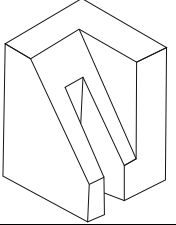
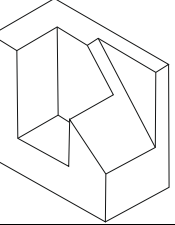
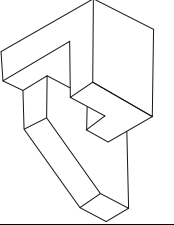
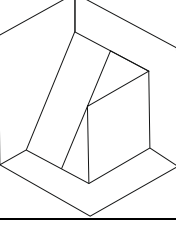
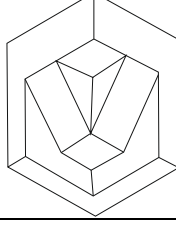
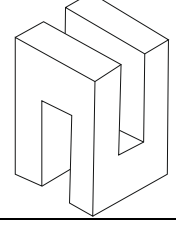
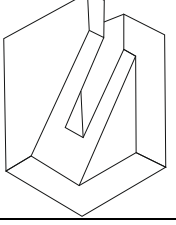
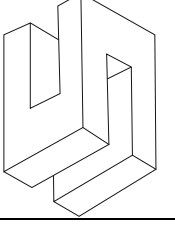
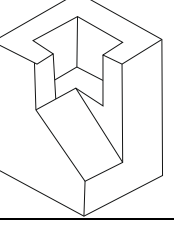
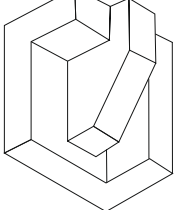
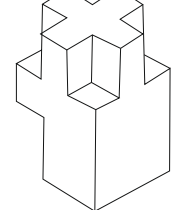
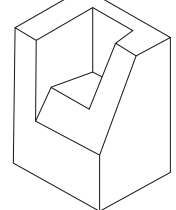
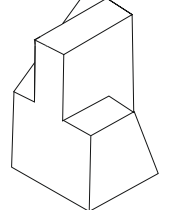
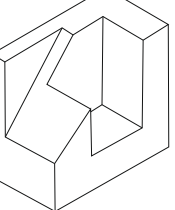
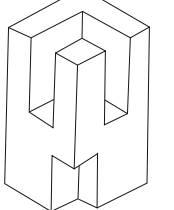
№	Функция	№	Функция	№	Функция
1	$y = \sin 2x + \cos 3x$	11	$y = \operatorname{tg} x + \sin(3 + x)$	21	$y = -\sin x + 2x$
2	$y = 5 \sin 4x + 4 \operatorname{tg} x + x $	12	$y = \sin 4x + \ln x - 2$	22	$y = \ln x + 2x - 1$
3	$y = 2 \cos x + 2 \ln 2x - 4 $	13	$y = x - 4 + 2x - 5 $	23	$y = \operatorname{ctg} x + 2 \sin x - 4$
4	$y = 2x + \ln x - \cos x$	14	$y = \sqrt{ x + \cos x }$	24	$y = 3 \cos 4x + 2 x $
5	$y = \cos x + \ln x $	15	$y = \sqrt{x + 3 x }$	25	$y = \sin(2 x - 3)$
6	$y = \sin x + x^2 + 3$	16	$y = \cos 6x + \sqrt{x^2 + 5}$	26	$y = 2x^2 + 3x + 1$
7	$y = \frac{ \cos x }{1 - \sin 2x } + 2x - 3$	17	$y = \frac{\sin 4x}{ x + 4 + 4} + 2x$	27	$y = \frac{1}{x^3 + 4x} + \cos 2x$
8	$y = 5 \cos 6x + \sqrt{2x - 3}$	18	$y = \cos 2x + 3x \ln x $	28	$y = 3 \operatorname{tg} 2x - \sin x $
9	$y = \cos 6x - x - 5 $	19	$y = \sin 7x + 2 x - 4 $	29	$y = x + \sin 2x$
10	$y = 4 \ln 2x + 3 + 2x + 2$	20	$y = \operatorname{tg} 6x - 3 \operatorname{ctg} x + 2$	30	$y = \operatorname{ctg} x + 2 \sin x - 1$

Практическое задание 2

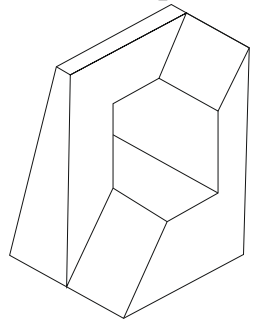
Написать программу для рисования заданного изображения, согласно варианту, с заливкой каждой области разным цветом.

Варианты



Вариант 7	Вариант 8	Вариант 9	Вариант 10	Вариант 11	Вариант 12
					
Вариант 13	Вариант 14	Вариант 15	Вариант 16	Вариант 17	Вариант 18
					
Вариант 19	Вариант 20	Вариант 21	Вариант 22	Вариант 23	Вариант 24
					
Вариант 25	Вариант 26	Вариант 27	Вариант 28	Вариант 29	Вариант 30
					

Примеры выполнения заданий

Задание 1	Задание 2
<p>Построить график функции $y = -\sin x$ на отрезке $[-7,7]$.</p> <pre> program uvp6_1; uses graph; const a=-7; b=7; wm=100; wn=100; var driver,mode: Integer; i,wx,wy: Integer; x,y: Real; begin driver:=Detect; initgraph(driver,mode,''); </pre>	<p>Написать программу для рисования заданного изображения:</p>  <pre> program uvp6_2; uses graph; var driver, mode: Integer; </pre>

<pre> wx:=getmaxX div 2; wy:=getmaxY div 2; setcolor(red); setlinestyle(1,0,3); line(0,wy,getmaxX,wy); line(wx,0,wx,getmaxY); setcolor(white); outtextXY(wx+7,15,'Y'); outtextXY(getmaxX-25, wy+6,'X'); setcolor(green); setlinestyle(0,0,1); i:=1; while (wx-i*wm>0) or (wx+i*wm <getmaxX) do begin line(wx-i*wm,0, wx-i*wm,getmaxY); line(wx+i*wm,0,wx+i*wm, getmaxY); inc(i); end; i:=1; while (wy-i*wn>0) or (wy+i*wn <getmaxY) do begin line(0,wy-i*wn,getmaxX, wy-i*wn); line(0,wy+i*wn,getmaxX, wy+i*wn); inc(i); end; i:=a*wm; while i<=b*wm do begin x:=i/wm; y:=-sin(abs(x)); putpixel(round(wx+wm*x), round(wy-wn*y),white); i:=i+1; end; repeat until KeyPressed; closegraph; end. </pre>	<pre> begin driver:=Detect; initgraph(driver,mode,''); setcolor(white); setlinestyle(0,0,1); line(60,120,220,20); line(120,140,280,40); line(220,20,360,60); line(280,37,240,120); line(360,60,320,140); line(240,120,320,140); line(240,120,160,160); line(160,160,160,280); line(320,140,320,260); line(160,280,240,300); line(160,220,320,260); line(240,300,320,260); line(120,400,160,280); line(180,420,240,300); line(360,60,360,330); line(180,420,360,330); line(20,370,180,420); line(60,120,20,370); line(120,400,120,140); line(60,120,120,140); setfillstyle(1,yellow); floodfill(160,140,15); setfillstyle(5,green); floodfill(160,80,15); setfillstyle(8,blue); floodfill(300,80,15); setfillstyle(1,red); floodfill(80,200,15); setfillstyle(1,brown); floodfill(260,160,15); setfillstyle(1,9); floodfill(300,300,15); setfillstyle(2,5); floodfill(160,300,15); repeat until KeyPressed; closegraph; end. </pre>
--	---

Требования к оформлению отчета

Содержание отчета:

1. Постановка задачи. Записывается условие задачи согласно варианта.
2. Изученные теоретические вопросы.
3. Код программы.
4. Результаты работы программы. Показываются скриншоты результатов работы программы.

Образец оформления титульного листа отчета:

Гомельский государственный университет им. Ф. Скорины Факультет математики и технологий программирования Кафедра вычислительной математики и программирования
ОТЧЕТ по учебной вычислительной практике Вариант ____
Выполнил: студент гр. ПМ-12 Иванов И.И.
Проверил: _____
Гомель 20__

Образец оформления титульного листа лабораторной работы:

Лабораторная работа №__ Тема: «Тема лабораторной работы» Вариант ____
Выполнил: студент гр. ПМ-12 Иванов И.И.
Проверил: _____

Литература

1. Бородич, Ю. С. Паскаль для персональных компьютеров : справочное пособие / Ю. С. Бородич, А. Н. Вальвачев, А. И. Кузьмич. – Минск : Выш. шк. : БФ ГИТМП «НИКА», 1991. – 365 с.
2. Долинский, М. С. Решение сложных и олимпиадных задач по программированию : учебное пособие / М. С. Долинский. – СПб. : Питер, 2006. – 366 с.
3. Зуев Е. А. Программирование на языке TURBO PASCAL 6.0, 7.0 / Е. А. Зуев. – М. : Радио и связь, 1993. – 384 с.
4. Лорин, Г. Сортировка и системы сортировки / Г. Лорин. – М. : Наука, 1983. – 384 с.
5. Немнюгин, С. А. Turbo Pascal : практикум / С. А. Немнюгин. – СПб. : Питер, 2000. – 256 с.
6. Программирование на языке Pascal : практическое пособие для студентов математических специальностей университета : в 2 ч. Ч. 1 / Е. А. Ружицкая [и др.]. – Гомель : ГГУ им. Ф. Скорины, 2005. – 108 с.
7. Программирование на языке Pascal : практическое пособие для студентов математических специальностей университета : в 2 ч. Ч. 2 / Е. А. Ружицкая [и др.]. – Гомель : ГГУ им. Ф. Скорины, 2005. – 92 с.
8. Турбо Паскаль 7.0. – К. : Торгово-издательское бюро ВНУ, 1996. – 448 с.
9. Фаронов, В. В. Turbo Pascal 7.0. Начальный курс : учебное пособие / В. В. Фаронов. – М. : «Нолидж», 1997. – 616 с.
10. Фаронов, В. В. Turbo Pascal 7.0. Практика программирования : учебное пособие / В. В. Фаронов. – М. : «Нолидж», 2001. – 416 с.

Производственно-практическое издание

Ружицкая Елена Адольфовна,
Москалева Марина Валерьевна

УЧЕБНАЯ ВЫЧИСЛИТЕЛЬНАЯ ПРАКТИКА

Язык программирования Pascal

Практическое пособие

Редактор *В. И. Шкредова*
Корректор *В. В. Калугина*

Подписано в печать 27.08.2019. Формат 60x84 1/16.
Бумага офсетная. Ризография. Усл. печ. л. 2,8.
Уч-изд. л. 3,1. Тираж 25 экз. Заказ 561.

Издатель и полиграфическое исполнение:
учреждение образования
«Гомельский государственный университет
имени Франциска Скорины».

Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий № 3/1452 от 17.04.2017.
Специальное разрешение (лицензия) № 02330/450 от 18.12.2013.
Ул. Советская, 104, 246019, Гомель.