

МИНИСТЕРСТВО ОБРАЗОВАНИЯ УКРАИНЫ
ДОНЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

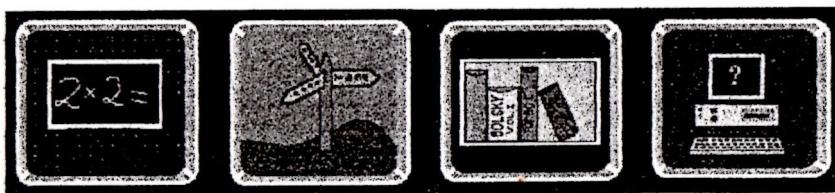
МЕТОДИЧЕСКИЕ УКАЗАНИЯ И ЗАДАНИЯ
К ЛАБОРАТОРНЫМ РАБОТАМ ПО КУРСУ

"ОСНОВЫ ПРОГРАММИРОВАНИЯ
И АЛГОРИТМИЧЕСКИЕ ЯЗЫКИ"

(для студентов специальности "Программное обеспечение"
и "ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ")

(для студентов специальности "Информационные системы в
менеджменте")

Часть 2



ДОНЕЦК ДонГТУ 1998

МИНИСТЕРСТВО ОБРАЗОВАНИЯ УКРАИНЫ
ДОНЕЦКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ И ЗАДАНИЯ
К ЛАБОРАТОРНЫМ РАБОТАМ ПО КУРСУ
**"ОСНОВЫ ПРОГРАММИРОВАНИЯ
И АЛГОРИТМИЧЕСКИЕ ЯЗЫКИ"**

(для студентов специальности "Программное обеспечение
автоматизированных систем")

и "ТЕХНОЛОГИЯ ПРОГРАММИРОВАНИЯ"

(для студентов специальности "Информационные системы
в менеджменте")

Часть 2

Утверждено на заседании
кафедры прикладной математики
и информатики

Протокол N 4 от 21.12.1997

Утверждено на заседании
методической комиссии
специальности "ПО"

Протокол N 3 от 25.12.1997

Утверждено на заседании
методической комиссии
специальности "МИ"
Протокол N 3 от 29.12.1997

ДОНЕЦК ДонГТУ 1998

Методические указания и задания к лабораторным работам по курсу "Основы программирования и алгоритмические языки" (для студентов специальности "Программное обеспечение автоматизированных систем") и "Технология программирования" (для студентов специальности "Информационные системы в менеджменте"). Ч.2/ Сост. Н.Н. Дацун, И.П. Суворова, И.А. Коломойцева. - Донецк, ДонГТУ, 1998 - 136 с.

Даны задания и указания к их выполнению по следующим темам: рекурсия; работа с файлами последовательного доступа; графические средства Turbo-C; работа с динамическими структурами данных; работа с файлами прямого доступа. Для каждой темы рассмотрены приемы алгоритмизации и программирования при решении задач данного типа, приведены примеры и программы на языке Си.

В подготовке материалов методических указаний принимали участие студенты групп МИ9баб, ПО9баб.

Составители

Н.Н. Дацун, доц.
И.П. Суворова, асс.
И.А. Коломойцева, асс.

Отв. за выпуск

Е.А. Башков, проф.

Рецензент

В.Н. Павлыш, доц.

7 ЛАБОРАТОРНАЯ РАБОТА № 6. РЕКУРСИЯ В ЯЗЫКЕ СИ

Цель работы: освоить приемы составления рекурсивных алгоритмов и программ на языке Си.

Контрольные вопросы:

- 1) Что такое рекурсия ?
- 2) Назовите виды рекурсии.
- 3) Назовите основные этапы проектирования рекурсивных алгоритмов.
- 4) Что такое "условие завершения рекурсии" ?
- 5) Назовите достоинства и недостатки рекурсивных алгоритмов.

Рекурсия как вычислительный процесс является альтернативой методу итераций (последовательных приближений).

7.1 Организация рекурсии при решении вычислительных задач по методу последовательных приближений

Расчетные формулы заданий (номера 1 - 35) представляют собой итерационные алгоритмы метода последовательных приближений. Выделяют следующие частные случаи итерационных процессов.

7.1.1 Алгоритм вычисления по явно заданной рекуррентной формуле

Например, вычисление квадратного корня

$$y = \sqrt{x} = y_{n+1} = 1/2 (y_n + x/y_n), \quad n=0,1,2,\dots \quad (7.1)$$

с погрешностью $|y_{n+1} - y_n| < \epsilon$. (7.2)

Начальное приближение вычисляется таким образом:

$$y_0 = \begin{cases} x/2, & \text{если } x > 1 \\ 2x, & \text{если } x \leq 1 \end{cases} \quad (7.3)$$

7.1.2 Алгоритм вычисления значения функции по формуле разложения функции в ряд

$$y = e^x = 1 + x + \frac{1}{2!}x^2 + \dots + \frac{1}{n!}x^n + \dots \quad (7.4)$$

с точностью очередного члена ряда $|\frac{1}{n!}x^n| < \epsilon$.

При суммировании подобных степенных рядов для сокращения объема вычислений необходимо вывести рекуррентные формулы. В данном случае они присутствуют неявно и позволяют выразить последующий член ряда через предыдущий:

$$a_n = \frac{1}{n!} x^n; \quad a_{n+1} = \frac{1}{(n+1)!} x^{n+1}; \quad (7.5)$$

Имеем $\frac{a_{n+1}}{a_n} = \frac{x}{n+1}$,

откуда получаем искомую рекуррентную формулу:

$$a_{n+1} = \frac{x}{n+1} * a_n; \quad (7.6)$$

7.1.3 Общий член ряда имеет более общий вид

Например, если общий член имеет вид

$$a_n = \frac{x^{2n+1}}{4n^2 - 1},$$

то его целесообразно представить в виде двух сомножителей, один из которых вычисляется по рекуррентному соотношению, а другой - непосредственно.

Пусть $C_n = x^{2n+1}$ и вычисляем рекуррентно $C_{n+1} = C_n * x$.

Тогда

$$a_{n+1} = \frac{C_n}{4n^2 - 1} \quad (7.7)$$

Управление итерационными циклами осуществляется по заданному значению погрешности вычислений.

Описанные итерационные процессы необходимо свести к рекурсивному вызову функций в алгоритмическом языке.

7.1.4 Пример 1

Определить функцию $y = \sqrt{x}$.

Строим иерархию функций, позволяющую определить целевую.

Вводим определения функций $y0$, $next_y$ и $body_sqrt$. Из (7.3) получаем функцию начального приближения:

```
float y0 (float x)
{ return (x>1.0 ? x/2 : x*x); }
```

Значение последующего члена ряда через предыдущий вычисляется по формуле (7.1):

```
float next_y(float x, float yn)
{ return((yn+x/yn)/2); }
```

Итерационный процесс до достижения заданной точности вычислений по формуле (7.2) определяем:

```
float body_sqrt (float x, float yn, float ynplus1, float eps)
{ if(fabs(ynplus1-yn)<eps)
    return(ynplus1);
else
    return (body_sqrt (x, ynplus1,
                      next_y(x, ynplus1), eps)
           ); }
```

На основе полученной иерархии определений функций составляем определение целевой функции:

```
float main_sqrt (float x, float eps)
{ float yn, ynplus1;
  yn= y0(x);
  ynplus1= next_y(x, y0(x));
  return (body_sqrt (x,yn, ynplus1, eps)); }
```

Это определение может быть записано короче с использованием композиции функций:

```
float main_sqrt (float x, float eps)
{ return (body_sqrt
          (x, y0(x),
           next_y(x, y0(x)),
           eps
          )
         ); }
```

Вызов функции $main_sqrt$ для значения $x=32$ и $\epsilon=0.01$ имеет вид:

```

main()
{float y; y=main_sqrt(32.0, 0.01); }

```

7.1.5 Программа на языке Си

```

#include <stdio.h>
#include <math.h>
/* ===== ФУНКЦИЯ НАЧАЛЬНОГО ПРИБЛИЖЕНИЯ ===== */
float y0(float x)
{ return(x>1.0 ? x/2 : x*x); }
/* ===== ФУНКЦИЯ ВЫЧИСЛЕНИЯ СЛЕДУЮЩЕГО ЧЛЕНА РЯДА ===== */
/* ===== ЧЕРЕЗ ПРЕДЫДУЩИЙ ===== */
float next_y(float x, float yn)
{ return((yn+x/yn)/2); }
/* ===== РЕКУРСИВНАЯ ФУНКЦИЯ ВЫЧИСЛЕНИЯ ЗНАЧЕНИЯ Y ===== */
float body_sqrt (float x, float yn, float ynplus1, float eps)
{ if(fabs(ynplus1-yn)<eps)
    return(ynplus1);
else
    return (body_sqrt (x, ynplus1, next_y(x, ynplus1), eps )
           );
}
/* ===== ФУНКЦИЯ ВЫЧИСЛЕНИЯ ЗНАЧЕНИЯ Y ПО ===== */
/* ===== ЗНАЧЕНИЯМ X И EPS ===== */
float main_sqrt (float x, float eps)
{ return
    (body_sqrt (x, y0(x), next_y(x, y0(x)), eps )
     );
}
/* ===== ГЛАВНАЯ ФУНКЦИЯ ===== */
main()
{ float x, eps, y;
printf("Введите x=");
scanf("%f", &x);
printf("и eps=");
scanf("%f", &eps);
y=main_sqrt(x, eps);

```

```

/* == СРАВНЕНИЕ ЗНАЧЕНИЙ ФУНКЦИИ, ОПРЕДЕЛЕННОЙ В ==
/* == MAIN_SQRT, И БИБЛИОТЕЧНОЙ ФУНКЦИИ (SQRT) ==
printf("Значения:\n");
printf("main_sqrt(%f, %f)=%f\n", x, eps, y);
printf("sqrt(%f, %f)=%f\n", x, eps, y);
getch();
}

```

7.2 Организация рекурсии при решении задач обработки данных
Рассмотрим организацию таких рекурсивных вычислений на примере решения следующей задачи:

Определить, является ли строка *st* симметричной.

7.2.1 Постановка задачи

7.2.1.1 Исходные данные

n: целое;
st: строка[1..*n*]; {длина строки *st*}
{исходная строка}

7.2.1.2 Ограничения

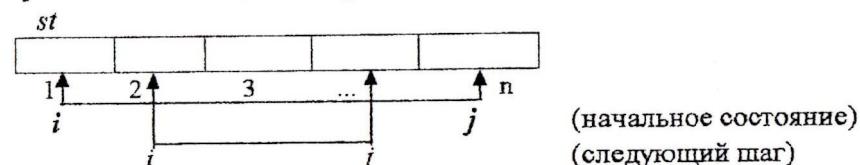
длина (*st*) ≠ 0 (*st* - непустая)

7.2.1.3 Результат

сообщение: строка[1..20]; {строка-сообщение}

7.2.1.4 Связь

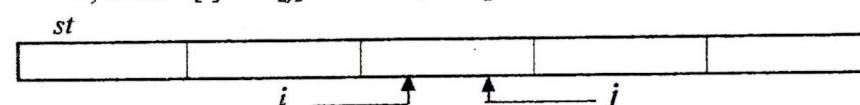
Сравниваем попарно символы строки *st* с индексами *i* и *j*, расположеннымими симметрично относительно ее середины:



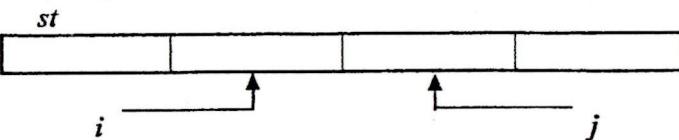
Если эти символы (*st[i]* и *st[j]*) равны, то "сокращаем" размер строки на один символ слева и справа, перемещая индексы *i* и *j* навстречу друг другу.

Условия завершения рекурсии:

- 1) Если символы с индексами *i* и *j* различны, то просмотр завершен неуспешно (*st* не является симметричной);
- 2) Если *st[i]* и *st[j]* совпали, и строка *st* нечетной длины



или строка *st* четной длины



и все предыдущие пары символов совпадали, то просмотр завершен успешно (*st* является симметричной).

7.2.2 Метод решения

```
j := j - 1 |  
i := i + 1 |  
не_конец := истина | st_i = st_j | пока не ( i = j или i = j - 1 )  
не_конец := ложь | st_i ≠ st_j | и не_конец )  
не_конец := истина  
i := 1  
j := n
```

Промежуточные данные:

не_конец: логическое; {признак: равен истина, если предыдущие симметричные относительно середины *st* пары символов совпадают}
i : целое; {индекс левого сравниваемого символа}
j : целое; {индекс правого сравниваемого символа}

7.2.3 Текст программы на языке Си

```
#include <stdio.h>  
#include <conio.h>  
  
#define N 20  
#define ESC 27  
  
main ()  
/* ===== ИСХОДНЫЕ ДАННЫЕ ===== */  
char s[N];  
/* ===== ПРОМЕЖУТОЧНЫЕ ДАННЫЕ ===== */  
int i, /* порядковый номер введенного в строку символа */  
m = 0; /* собственно введенный в строку символ */  
/* ===== прототип вспомогательной функции sim ===== */  
int sim(char *st,int i,int j);
```

```
/* ===== ИСПОЛНИМАЯ ЧАСТЬ АЛГОРИТМА =====*/  
/* ввод исходной строки посимвольно и оформление строки */  
while (m != ESC)  
{ printf("\nВведите строку :\n ");  
i=0;  
do  
    s[i++] = getchar();  
while (s[i-1]!='\n');  
s[i-1]='\0'; /* оформление массива символов как строки */  
if (sim(s, 0, strlen(s)-1))  
    printf("Эта строка симметрична\n");  
else  
    printf("Эта строка несимметрична\n");  
printf("Для выхода нажмите ESC, для продолжения - ENTER.");  
m=getch(); /* продолжение ввода строк */  
}  
}  
/* = Вспомогательная функция проверки строки st на симметричность */  
int sim(char *st, int i, int j)  
{ if ((i==j)||((i==j-1)&&(st[i]==st[j])))  
    return 1;  
else  
    if ((st[i]==st[j])&&(sim(st, i+1, j-1)==1))  
        return 1;  
    else return 0;  
}
```

7.3 Организация рекурсии при решении задач, использующих имя функции как аргумент другой функции

Рассмотрим организацию таких рекурсивных вычислений на примере решения следующей задачи:

Методом деления отрезка пополам найти с точностью *eps* корень уравнения $f(x)=0$ на отрезке $[a,b]$. Решить этим методом уравнения $x^2-2x-5=0$ и $\sin(2x-3)=0$.

Если для уравнения $f(x)=0$ определен интервал изоляции корня (некоторый интервал, на котором ровно один корень этого уравнения), то нахождение корня сводится к следующему алгоритму.

Шаг 1. Отрезок $[a, b]$ делится пополам.

Шаг 2. Если значения функции $f(x)$ на концах этого отрезка различаются меньше, чем на eps , то в качестве корня исходного уравнения принимается вычисленная середина отрезка $[a, b]$.

Шаг 3. В противном случае анализируется, в какой половине этого отрезка (левой или правой) после деления остается корень.

Шаг 4. Поиск корня продолжается соответственно в той половине отрезка, на концах которого функция $f(x)$ имеет различные знаки.

Этот алгоритм применим к решению алгебраических и трансцендентных уравнений различной сложности. Поэтому необходимо выбрать структуру данных для представления вида соответствующего уравнения.

Такой структурой данных в алгоритмических языках выступает функция, аргумент которой соответствует переменной уравнения, а выражение, определяющее возвращаемое функцией значение, соответствует левой части уравнения $f(x)=0$.

При решении различных уравнений нужно передавать имена функций, соответствующих этим уравнениям, аргументом в функцию нахождения корня методом половинного деления.

7.3.1. Постановка задачи

7.3.1.1 Исходные данные

a, b: вещественные; {интервал изоляции корня уравнения $f(x)=0$ }
 f1, f2.. fn: функции; {функции, соответствующие вычислению
значения f по значению аргумента x }
 eps: вещественное; {точность вычисления корня уравнения}

7.3.1.2 Ограничения

$f_1(a)*f_1(b) < 0$ $\{[a, b] \text{ - это интервал изоляции корней}$
 $f_2(a)*f_2(b) < 0 \dots$ $\text{уравнений } f_1(x)=0, \dots f_n(x)=0\}$

$$f_n(a) * f_n(b) < 0$$

7.3.1.3 Результаты
x: вещественное;
 сообщение: строка[1, 20]; {корень уравнения $f(x)=0$ }
{строка-сообщение}

7.3.3 Такие программы на языке C

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
```

```

/* ===== ИСХОДНЫЕ ДАННЫЕ ===== */
float eps; /* точность вычисления корня уравнения */
/* ===== РЕЗУЛЬТАТ ===== */
float xx; /* корень уравнения  $f(x)=0$  */
/* исходные данные: функции, соответствующие решаемым уравнениям*/
float f1(float x)
{ return(x*x-2*x-5);}
float f2(float x)
{ return(sin(2*x-3));}
/* == функция решения уравнения методом половинного деления == */
void root(float a, float b, float f(float))
{ float c; /* середина текущего исследуемого интервала */
/* ===== ИСПОЛНИМАЯ ЧАСТЬ АЛГОРИТМА ===== */
if(f(a)*f(b) <=0) /* проверка ограничений */
{ c=(a+b)/2;
  if (fabs ((double)(f(a)-f(b))) < (double)eps)
    xx = c;
  else
    if(f(c)*f(b)>0.0)
      root(a, c, f ); /* поиск в правой половине интервала */
    else
      if(f(a)*f(c)>0.0)
        root(c, b, f ); /* поиск в левой половине интервала */
}
else
{ printf("Корней на этом интервале нет !!!\n");
  exit(-1);
}
}
main()
{float a, /* == ИСХОДНЫЕ ДАННЫЕ: левая граница */
 b; /* и правая граница интервала изоляции корня*/
/* = прототип вспомогательной функции нахождения корня == */
void root (float a, float b, float (*f)(float x));
clrscr ();
/* ===== ВВОД ИСХОДНЫХ ДАННЫХ ===== */
printf("Введите интервал изоляции корня уравнения:\n");

```

```

printf("левая граница = ");      scanf("%f", &a);
printf("правая граница = ");      scanf("%f", &b);
printf("Введите точность нахождения корня уравнения = ");
scanf("%f", &eps);
/* ===== РЕШЕНИЕ УРАВНЕНИЯ  $x^2 - 2x - 5 = 0$  =====*/
root(a, b, f1 );
printf("Корень 1-го уравнения есть \"%f\n", xx);
getch();
/* ===== РЕШЕНИЕ УРАВНЕНИЯ  $\sin(2x-3)=0$  =====*/
root(a, b, f2 );
printf("Корень 2-го уравнения есть \"%f\n", xx);
getch();
}

```

7.4 Задания

1. Определить функцию (-ии), реализующую (-ие) заданный алгоритм.
 2. Определить количество шагов рекурсии, необходимых для решения задачи.
- Дополнительные задания для вариантов 1 - 35:
1. При необходимости вывести рекуррентные соотношения, связывающие последующий и предыдущий элементы программируемой формулы.
 2. При программировании алгебраических выражений воспользоваться определениями более примитивных функций и на их основе получить иерархию определений. Конструирование целевой функции выполнить методом композиции ранее определенных функций. Варианты заданий приведены в таблице 7.1 [1].
 3. Записать вызов определенной функции для заданных значений аргументов.
 4. Провести тестирование программы для заданных значений аргументов; определить количество шагов рекурсии, необходимых для достижения заданной точности.
 5. Сравнить работу определенной Вами функции и соответствующей библиотечной.

7.4.1 Варианты заданий 1-35

Таблица 7.1 - Варианты заданий

№	Вычисляемая функция	Вычислительные формулы	Значения аргументов
1	e^x	$y = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k}}{k!}$	$x = 2$ $\epsilon = 0.01$
2	$\operatorname{th} x$	$y = 1 + 2 * \sum_{k=1}^{\infty} (-1)^k \frac{(e^{-2kx} - 1)}{k}$ [$x > 0$]	$x = 3$ $\epsilon = 0.01$
3	$\operatorname{sech} x$	$y = 2 * \sum_{k=0}^{\infty} (-1)^k \frac{(e^{-2(k+1)x} - 1)}{k}$ [$x > 0$]	$x = 1.5$ $\epsilon = 0.1$
4	$\operatorname{cosech} x$	$y = 2 * \sum_{k=0}^{\infty} \frac{e^{-2(k+1)x}}{k}$ [$x > 0$]	$x = 2.5$ $\epsilon = 0.2$
5	$\sin x$	$y = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!}$	$x = 2$ $\epsilon = 0.01$
6	$\cos x$	$y = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k}}{(2k)!}$	$x = 1.2$ $\epsilon = 0.01$
7	$\operatorname{sh} x$	$y = \sum_{k=0}^{\infty} \frac{x^{2k+1}}{(2k+1)!}$	$x = 1.3$ $\epsilon = 0.01$
8	$\operatorname{ch} x$	$y = \sum_{k=0}^{\infty} \frac{x^{2k}}{(2k)!}$	$x = 2$ $\epsilon = 0.01$
9	$\sin^2 x$	$y = \sum_{k=1}^{\infty} (-1)^k \frac{(2^{k+1} - x^{2k})}{(2k)!}$	$x = 3$ $\epsilon = 0.1$
10	$\cos^2 x$	$y = 1 - \sum_{k=1}^{\infty} (-1)^k \frac{(2^{k+1} - x^{2k})}{(2k)!}$	$x = 2$ $\epsilon = 0.01$
11	$\sin^3 x$	$y = 1/4 \sum_{k=1}^{\infty} (-1)^k \frac{(3^{k+1} - 3x^{2k+1})}{(2k+1)!}$	$x = 1.5$ $\epsilon = 0.1$

Продолжение таблицы 7.1

№	Вычисляемая функция	Вычислительные формулы	Значения аргументов
12	$\cos^3 x$	$y = \frac{1}{4} \sum_{k=0}^{\infty} (-1)^k * (3 + 3) * x^{2k} / (2k)!)$	$x = 1.7$ $\epsilon = 0.1$
13	$\ln(1+x)$	$y = \sum_{k=1}^{\infty} (-1)^{k+1} * (x^k / k)$ [$-1 < x \leq 1$]	$x = 0.8$ $\epsilon = 0.01$
14	$\ln x$	$y = \sum_{k=1}^{\infty} (-1)^k * ((x-1)/k)$ [$0 < x \leq 2$]	$x = 1.1$ $\epsilon = 0.1$
15	$\ln x$	$y = 2 \sum_{k=1}^{\infty} (1/(2k-1)) * ((x-1)/(x+1))$ [$0 < x$]	$x = -0.3$ $\epsilon = 0.1$
16	$\ln x$	$y = \sum_{k=1}^{\infty} (1/k) * ((x-1)/x)$ [$x \geq 1/2$]	$x = 0.7$ $\epsilon = 0.1$
17	$\ln((1+x)/(1-x))$	$y = 2 \sum_{k=1}^{\infty} (1/(2k-1)) * x^{2k-1}$ [$x^2 < 1$]	$x = 0.09$ $\epsilon = 0.01$
18	$\ln((x+1)/(x-1))$	$y = 2 \sum_{k=1}^{\infty} 1 / ((2k-1)*x^{2k-1})$ [$x^2 > 1$]	$x = 1.44$ $\epsilon = 0.01$
19	$\ln(x/(x-1))$	$y = \sum_{k=1}^{\infty} 1 / (kx)$ [$x^2 > 1$]	$x = 2$ $\epsilon = 0.01$
20	$\ln(1/(1-x))$	$y = \sum_{k=1}^{\infty} (x^k / k)$ [$x^2 < 1$]	$x = 0.7$ $\epsilon = 0.01$

Продолжение таблицы 7.1

№	Вычисляемая функция	Вычислительные формулы	Значения аргументов
21	$(1-x) / x * \ln(1/(1-x))$	$y = 1 - \sum_{k=1}^{\infty} x^k / (k * (k+1))$ [$x^2 < 1$]	$x = 0.8$ $\epsilon = 0.01$
22	$\operatorname{arctg} x$	$y = \sum_{k=0}^{\infty} (-1)^k * x^{2k+1} / (2k+1)$	$x = 0.9$ $\epsilon = 0.01$
23	$\operatorname{Arth} x$	$y = \sum_{k=0}^{\infty} x^{2k+1} / (2k+1)$ [$x^2 < 1$]	$x = 0.75$ $\epsilon = 0.01$
24	$\operatorname{Arth}(1/x)$	$y = \sum_{k=0}^{\infty} x^{-(2k+1)} / (2k+1)$ [$x^2 > 1$]	$x = 2$ $\epsilon = 0.01$
25	$\ln(x + \sqrt{1+x^2}) / \sqrt{1+x^2}$	$y = \sum_{k=0}^{\infty} (-1)^k * 2^k * (k!) * x^{2k+1} / ((2k+1)!)$ [$x^2 < 1$]	$x = 0.8$ $\epsilon = 0.01$
26	$\operatorname{sh} t / (\operatorname{ch} t - \cos x)$ [$t > 0$]	$y = 1 + 2 \sum_{k=1}^{\infty} e^{-kt} \cos kx$ [$t > 0$]	$x = 0.1$ $\epsilon = 0.01$ $t = 1.0$
27	$1/2 \ln \operatorname{tg}(\pi/4 + x/2)$	$y = \sum_{k=1}^{\infty} (-1)^{k-1} \sin(2k-1)x / (2k-1)$ [- $\pi/2 < x < \pi/2$]	$x = 0.8$ $\epsilon = 0.01$
28	$1/2 \ln \operatorname{ctg}(x/2)$	$y = \sum_{k=1}^{\infty} \cos(2k-1)x / (2k-1)$ [$0 < x < \pi$]	$x = 0.9$ $\epsilon = 0.01$
29	$\ln(2 \cos(x/2))$	$y = \sum_{k=1}^{\infty} (-1)^{k-1} \cos(kx) / k$ [- $\pi < x < \pi$]	$x = 1$ $\epsilon = 0.01$
30	$1/2 \ln(1 / (2 * (1 - \cos x)))$	$y = \sum_{k=1}^{\infty} \cos(kx) / k$ [$0 < x < 2\pi$]	$x = 1$ $\epsilon = 0.01$

Окончание таблицы 7.1

№ варианта	Вычисляемая функция	Вычислительные формулы	Значения аргументов
32	$\text{Arctg } x$	$y = \sum_{k=0}^{\infty} x^{-(2k+1)} / (2k+1) \quad [x^2 > 1]$	$x = 1.5$ $\varepsilon = 0.01$
33	$e^x (1+x)$	$y = \sum_{k=0}^{\infty} x^k (k+1) / (k!)$	$x = 3$ $\varepsilon = 0.01$
34	$\sin x$	$y = x \prod_{k=1}^{\infty} \cos(x/2^k) \quad [x < 1]$	$x = 0.7$ $\varepsilon = 0.01$
35	$\text{th}(\pi x / 2)$	$y = 4x/\pi \sum_{k=1}^{\infty} 1 / ((2k-1)^2 + x^2)$	$x = 2.5$ $\varepsilon = 0.01$

7.4.2 Варианты заданий 36 - 97

36. Описать рекурсивную функцию $sum(x, n)$, которая находит сумму элементов вещественного вектора x длины n .

37. Дано n различных натуральных чисел. Сгенерировать все перестановки этих чисел (см. пример в [2], с.265-266).

38. Правильное скобочное выражение получается из некоторого математического выражения, содержащего круглые скобки, вычеркиванием всех знаков, кроме круглых скобок. Например, из выражения $x-y(a+3(b+c(d-5)))+x(a+b)$ получается правильное скобочное выражение $((()))()$.

Описание синтаксиса для правильных скобочных выражений:

- а) $()$ - правильное скобочное выражение;
- б) если P - правильное скобочное выражение, то (P) - правильное скобочное выражение;

в) если P и Q - правильные скобочные выражения, то PQ - правильное скобочное выражение.

Дано натуральное число n и последовательность символов c_1, c_2, \dots, c_{2n} , каждый из которых - круглая скобка. Определить

функцию для ответа на вопрос: "Является ли последовательность c_1, c_2, \dots, c_{2n} правильным скобочным выражением?"

39. Подсчитать количество различных представлений заданного натурального числа n в виде суммы двух попарно различных положительных слагаемых. Представления, различающиеся лишь порядком слагаемых, различными не считаются.

40. Последовательность чисел Фибоначчи образуется по закону $u_0 = 0; u_1 = 1; u_i = u_{i-1} + u_{i-2}; i = 2, 3, \dots$. Определить функцию вычисления i -го числа Фибоначчи.

41. Во входной строке задан текст, за которым следует точка. Проверить, удовлетворяет ли его структура следующему определению:

$<\text{текст}> ::= <\text{элемент}> | <\text{элемент}> <\text{текст}>$

$<\text{элемент}> ::= a | b | (<\text{текст}>) | [<\text{текст}>] | \{<\text{текст}>\}$

42. Задача о "восьми ферзях": на шахматной доске расставить 8 ферзей так, чтобы они не "били" друг друга. Описать функцию, которая получает одну такую расстановку.

43. Определить функцию вычисления значения

$$1 - \frac{1}{2} + \frac{1}{3} - \dots + \frac{1}{9999} - \frac{1}{10000}$$

последовательно слева направо. Сравнить результат ее работы с работой аналогичных функций из вариантов 53, 63, 73. Объяснить, почему при вычислении на ЭВМ каждым из этих способов получаются такие результаты.

44. Реализовать для стека операцию "в". Разместить в стеке n элементов (первоначально стек пуст). Стек промоделировать массивом, а "вершину стека" - указателем занятости этого массива.

45. Напечатать в обратном порядке заданный во входной строке текст. Текст завершается точкой, восклицательным или вопросительным знаками.

46. Описать рекурсивную функцию $min_pol(x, n)$, которая находит минимальный из положительных элементов целочисленного вектора x длины n .

47. Найти среднее арифметическое положительных и отрицательных элементов целочисленного вектора заданного размера.

Операции определения знака числа реализовать как функции, передаваемые аргументом в определяемую рекурсивную.

48. Во входной строке записана (без ошибок) формула следующего вида:

```
<формула> ::= <цифра> | ( <формула> <знак> <формула> )
<знак> ::= + | - | *
<цифра> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

Ввести эту формулу и вычислить ее значение. Например, по строке "5" вычисляется значение 5, а по строке "2-4*6" - значение -22.

49. Подсчитать количество различных представлений заданного натурального числа n в виде произведения двух попарно различных положительных сомножителей. Представления различающиеся лишь порядком сомножителей считаются различными.

50. Биномиальный коэффициент: $C_n^m = \binom{n}{m} = \frac{n!}{m!(n-m)!}$;

$C_0^n = 1$. Определить функцию $C(n, m)$ для вычисления биномиальных коэффициентов.

51. Во входной строке задан текст, за которым следует точка. Проверить, удовлетворяет ли его структура следующему определению:

```
<идентификатор> ::= <буква>
<идентификатор> ::= <буква> <продолжение идентификатора>
<продолжение идентификатора> ::= <буква> <продолжение идентификатора>
<продолжение идентификатора> ::= <цифра> <продолжение идентификатора>
<буква> ::= a | b | c | ... | z
<цифра> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
```

52. Описать рекурсивную функцию $\max_sim(x, n)$, которая находит максимум из разностей элементов являющихся симметричными относительно его середины (x - целочисленный вектор длины n).

53. Определить функцию вычисления значения:

$$1 - \frac{1}{2} + \frac{1}{3} - \dots + \frac{1}{9999} - \frac{1}{10000}$$

сначала последовательно слева направо вычисляется сумма $1 + \frac{1}{3} + \dots + \frac{1}{9999}$, затем слева направо $\frac{1}{2} + \frac{1}{4} + \dots + \frac{1}{10000}$, затем из первой суммы вычитается вторая. Сравнить результат ее работы с работой аналогичных функций из вариантов 43, 63, 73. Объяснить, почему при вычислении на ЭВМ каждым из этих способов получаются такие результаты.

54. Реализовать для очереди операцию "в". Разместить в очереди n элементов (первоначально очередь пуста). Очередь промоделировать массивом, а "начало" и "конец" очереди - указателями занятости этого массива.

55. Описать функцию $digits$, которая подсчитывает количество цифр в тексте, заданном во входной строке. Текст завершается точкой, запятой или точкой с запятой.

56. Описать рекурсивную функцию $\max_otr(x, n)$, которая находит максимальный из отрицательных элементов вещественного вектора x длины n .

57. Найти среднее геометрическое четных и нечетных элементов целочисленного вектора заданного размера. Операции определения четности числа реализовать как функции, передаваемые аргументом в определяемую рекурсивную.

58. Вычислить

$$\cfrac{1}{1 + \cfrac{1}{3 + \cfrac{1}{5 + \cfrac{1}{\dots + \cfrac{1}{51 + \cfrac{1}{53}}}}}}$$

59. Дано натуральное число n . Вычислить произведение первых n сомножителей: $\frac{1}{2} \cdot \frac{3}{4} \cdot \frac{5}{6} \dots$

60. Определить функцию нахождения

$$\binom{N+1}{n+1} = \sum_{j=n}^N \binom{j}{n},$$

используя формулу для биномиальных коэффициентов (вариант 50).

61. Преобразовать выражение (т.е. текст специального вида), составленное из цифр и знаков арифметических операций сложения и вычитания, в постфиксную форму. В постфиксной форме сначала записываются операнды, а затем знак операции. Например,

обычная (инфиксная) запись	постфиксная запись
$3+4$	$34+$
$(5-4)+2$	$54-2+$
$2-(3+4)-6$	$234+-6-$

62. Описать рекурсивную функцию $\min_el(x, n)$, которая находит минимальный из ненулевых элементов, имеющих нечетные номера (x - вещественный вектор длины n).

63. Определить функцию вычисления значения

$$1 - \frac{1}{2} + \frac{1}{3} - \dots + \frac{1}{9999} - \frac{1}{10000}$$

последовательно справа налево. Сравнить результат ее работы с работой аналогичных функций из вариантов 43, 53, 73. Объяснить, почему при вычислении на ЭВМ каждым из этих способов получаются такие результаты.

64. Реализовать для стека операцию "из". Взять из стека n элементов (первоначально в стеке m элементов). Стек промоделировать массивом, а "вершину стека" - указателем занятости этого массива.

65. Данна последовательность ненулевых целых чисел, за которыми следует 0. Напечатать сначала все отрицательные числа этой последовательности, а затем все положительные (в порядке слева направо). Операции сравнения на неравенство реализовать как функции, передаваемые аргументом в определяемую рекурсивную.

66. Описать рекурсивную функцию $proiz(x, n)$, которая находит произведение тех элементов целочисленного вектора x длины n , которые кратны 3.

67. Найти количество четных и нечетных элементов целочисленного вектора заданного размера. Операции определения четности числа реализовать как функции, передаваемые аргу-

ментом в определяемую рекурсивную.

68. Дано действительное число $x \neq 0$. Вычислить

$$\begin{aligned} & \frac{x^2}{x^2 + \frac{2}{x^2}} \\ & x^2 + \frac{8}{x^2 + \dots} \\ & x^2 + \frac{256}{x^2 + \dots} \end{aligned}$$

69. Дано натуральное число n . Вычислить произведение первых n сомножителей: $\frac{1}{1} \cdot \frac{3}{2} \cdot \frac{5}{3} \cdots$

70. Имеется n населенных пунктов, перенумерованных от 1 до n ($n=10$). Некоторые пары пунктов соединены дорогами. Определить, можно ли попасть по этим дорогам из 1-го пункта в n -ый. Информация о дорогах задается в виде последовательности пар чисел i и j ($i < j$), указывающих, что i -й и j -й пункты соединены дорогой; признак конца этой последовательности - пара нулей.

71. Преобразовать выражение (т.е. текст специального вида), составленное из цифр и знаков арифметических операций умножения и деления, в префиксную форму. В префиксной форме сначала записывается знак операции, а затем операнды. Например,

обычная (инфиксная) запись	префиксная запись
$3*4$	$*34$
$(5/4)*2$	$*/542$
$2/(3*4)/6$	$//2*346$

72. Задача о "восьми ферзях": на шахматной доске расставить 8 ферзей так, чтобы они не "били" друг друга. Определить функцию, которая генерирует все такие расстановки.

73. Определить функцию вычисления значения:

$1 - \frac{1}{2} + \frac{1}{3} - \dots + \frac{1}{9999} - \frac{1}{10000}$
 сначала последовательно справа налево вычисляется сумма
 $1 + \frac{1}{3} + \dots + \frac{1}{9999}$, затем справа налево $\frac{1}{2} + \frac{1}{4} + \dots + \frac{1}{10000}$,
 затем из первой суммы вычитается вторая. Сравнить результат ее

работы с работой аналогичных функций из вариантов 43, 53, 63. Объяснить, почему при вычислении на ЭВМ каждым из этих способов получаются такие результаты.

74. Найти такую расстановку двенадцати коней на шахматной доске, при которой каждое поле будет находиться под ударом одного из них.

75. Получить все перестановки элементов 1, 2 ... 6.

76. Реализовать для очереди операцию "из". Взять из счёреди n элементов (первоначально в очереди m элементов). Очередь промоделировать массивом, а "начало" и "конец" очереди - указателями занятости этого массива.

77. Указать маршрут коня, начинаящийся на одном заданном поле шахматной доски и оканчивающийся на другом заданном. Никакое поле не должно встречаться в маршруте дважды.

78. Лабиринт может быть задан матрицей соединений, в котором для каждой пары комнат указано, соединены ли они коридором. Данна матрица соединений лабиринта из n комнат и номера комнат i, j ($1 \leq i \leq n, 1 \leq j \leq n$). Построить путь из комнаты с номером i в комнату с номером j .

79. Имеется n городов. Некоторые из них соединены дорогами известной длины. Вся система дорог задана квадратной матрицей порядка n , элемент A_{ij} которой равен некоторому отрицательному числу, если город i не соединен напрямую дорогой с городом j и равен длине дороги в противном случае ($i, j = 1, 2 \dots n$). Для первого города найти кратчайшие маршруты в остальные города.

80. Получить все полные перестановки 10 элементов 1, 2 ... 10, (перестановка P_1, \dots, P_n элементов 1, 2, ... n называется полной, если $P_i \neq i$ для $i=1, 2 \dots n$).

81. Найти такую расстановку пяти ферзей на шахматной доске, при которой каждое поле будет находиться под ударом одного из них.

82. Дано натуральное число n ($n \leq 99$). Получить все способы выплаты суммы n с помощью монет достоинством 1, 5, 10 и 20 коп.

83. Построить все правильные скобочные выражения длины 10, т.е. те, которые содержат по пять левых и по пять правых круглых скобок.

84. Получить все размещения из 9 элементов 1, 2 ... 9 по пять элементов в каждом.

85. Назовем натуральное число палиндромом, если его запись читается с начала и с конца (как, например, 4884, 393, 1). Найти все меньшие 100 натуральные числа, которые при возведении в квадрат дают палиндром.

86. В данной последовательности действительных чисел a_1, a_2, \dots, a_{20} выбрать возрастающую последовательность наибольшей длины.

87. Имеется n предметов, веса которых равны A_1, A_2, \dots, A_n . Разделить эти предметы на две группы так, чтобы общие веса двух групп были максимально близки.

88. Получить последовательность A_1, A_2, \dots, A_n из цифр 0, 1, 2, в которых нет смежных одинаковых участков (например последовательность 2, 0, 1, 1, ... не годится, так как рядом расположены две одинаковых цифры 1; последовательность 2, 1, 0, 1, 2, 1, 0, 1 не годится, так как рядом расположены два одинаковых участка 2, 1, 0, 1 и т.д.)

89. Два натуральных числа называют дружественными, если каждое из них равно сумме всех делителей другого, кроме самого этого числа. Найти все пары дружественных чисел лежащих в диапазоне от 200 до 300.

90. Найти такую расстановку восьми слонов на шахматной доске, при которой каждое поле будет находиться под ударом одного из них.

91. Дано натуральное число n . Среди чисел 1, 2 .. n найти все такие, запись которых совпадает с последними цифрами записи их квадрата (например $6^2 = 36, 25^2 = 625$ и т.д.).

92. Назовем натуральное число палиндромом, если его запись читается с начала и с конца (например, 4884, 393, 1). Найти все меньшие 100 числа-палиндромы, которые при возведении в квадрат дают палиндромы.

93. Дано натуральное число n . Выяснить, можно ли представить $n!$ в виде произведения трех последовательных чисел.

94. Найти все простые нескратимые дроби, заключенные между 0 и 1, знаменатели которых не превышают 7 (дробь задается двумя натуральными числами - числителем и знаменателем).

95. Дано натуральное число m . Вставить между некоторыми цифрами 1, 2, 3, 4, 5, 6, 7, 8, 9, записанными именно в таком порядке, знаки “+” и “-” так, чтобы значением получившегося выражения было число m . Например, если $m=122$, то подойдет следующая расстановка знаков: $12+34-5-6+78+9$.

96. Дано натуральное число n . Получить в порядке возрастания n первых натуральных чисел, которые не делятся ни на какие простые числа, кроме 2, 3 и 5.

97. Получить все сочетания из 10 элементов 1, 2... 10 по четыре элемента в каждом.

7.5 Контрольные вопросы к лабораторной работе

- 1) Что такое “условие завершения рекурсии”?
- 2) Как реализовать решение Вашего варианта задания с помощью итерационного процесса?
- 3) Реализовать вычисление x^n в виде рекурсивной функции.

7.6 Содержание отчета

1. Для вариантов 1 - 35:

- Условие задачи и вывод рекуррентных соотношений, связывающих последующий и предыдущий элементы программируемой формулы.
- Ручное тестирование программы для заданных значений аргументов.

Для остальных вариантов:

- Условие задачи.
- Математические расчеты и вывод формул для раздела “Связь” постановки задачи.
- Постановка задачи.
- Метод решения задачи.
- Алгоритмы (вспомогательные и основной) и их иерархия.
- Таблица трассировки программы.
- Графическая интерпретация тестовых примеров (с изображением формируемых массивов, строк, их указателей).

2. Для всех вариантов:

- Текст программы на языке Си.
- Выводы.
- Программный документ “Описание программы”.

8 ЛАБОРАТОРНАЯ РАБОТА № 7. РАБОТА С ФАЙЛАМИ ПОСЛЕДОВАТЕЛЬНОГО ДОСТУПА НА ЯЗЫКЕ СИ

Цель работы: освоить работу с файлами последовательного доступа в языке Си и основные приемы программирования задач, использующих такие файлы; освоить передачу аргумента функции *main* в языке Си.

Контрольные вопросы:

- 1) Понятие “файл” в языках программирования.
- 2) Что такое “файл” в языке Си?
- 3) Операции доступа к файлам в языке Си.
- 4) Стандартные устройства ввода-вывода, используемые как файлы в языке Си.
- 5) Переназначение стандартных устройств ввода-вывода в языке Си.
- 6) Операции форматированного ввода-вывода в языке Си.
- 7) Назначение спецификации преобразования данных при форматированном вводе-выводе в языке Си.
- 8) Неформатированный ввод-вывод в языке Си.
- 9) Особенности неформатированного ввода-вывода строк в языке Си
- 9) Ввод-вывод в языке Си со стандартных устройств.

8.1 Приемы алгоритмизации и программирования при работе с последовательными файлами

8.1.1 Открытие файлов и проверка ограничений

Открытие файла в языке Си выполняется с помощью функции *fopen*. Первый аргумент этой функции является именем файла в соответствующей операционной системе. Это имя может быть задано в программе одним из трех способов:

- 1) как строковая константа;
- 2) как значение строки (например, задаваемое пользователем при вводе или формируемое в программе);
- 3) как значение аргумента функции *main*.

В любом из этих случаев максимальная длина имени файла равна:

8 + 1 + 3 + 1 = 13 [байтов]

↓ ↓ ↓ ↓ ↓
 собственно имя точка расширение признак
 конца строки

Таким образом, объявление переменной для хранения имени файла имеет вид:

```
char name_file[13];
```

В зависимости от способа задания имени открываемого файла следует организовать и проверку ограничений.

1) Если имя файла задано как строковая константа в тексте программы, то в случае неуспеха при открытии файла после сообщения пользователю необходимо:

- прекратить выполнение программы;

```
FILE *file_in;  
if((file_in=fopen("name.txt", "r"))==NULL)  
    {printf("Файл name.txt не открыт!!!\n");  
     exit(-1);  
}
```

- исправить в тексте программы имя файла;
 - оттранслировать программу после исправлений;
 - повторить ее выполнение.

2) Если имя файла задается пользователем при вводе как значение строки, то в случае неуспеха при открытии файла после сообщения пользователю возможно повторение ввода имени файла:

```
char name_file[13];
FILE *file_f;
int pr;
do
{
    printf("Введите имя файла\n");
    scanf("%s", name_file);
    if((file_f=fopen(name_file, "r"))==NULL)
```

```
{printf("Файл %s не открылся!!!\n", name_file);
pr=1;
}
else pr=0;

(pr);
```

3) Если имя файла задано как значение аргумента функции `main`, то предварительно необходимо:

- проверить наличие этого аргумента;
 - в случае его отсутствия после сообщения пользователю прекратить выполнение программы

```
main(int argc, char * argv[])
{... if (argc == 0)          /* main вызвана без аргументов */
    {printf("При вызове укажите имя файла !!!\n");
     exit(-1);
    }
}
```

- повторить ее выполнение с заданием имени файла в виде аргумента функции *main*.

В случае неуспеха при открытии файла после сообщения пользователю необходимо (далее предполагается, что имя файла передается первым аргументом функции *main*):

- прекратить выполнение программы

```
main (int argc, char * argv[])
{FILE * file_f; ...
 if((file_f=fopen(argv[1], "r"))==NULL)
    {printf("Файл %s не открыт!!!\n", argv[1]);
     exit(-1);
    }
}
```

- повторить ее выполнение с корректным именем файла в виде аргумента функции *main*.

8.1.2 Изменение статуса файла

Статус файла определяется вторым аргументом функции

fopen. Он задает способ обработки этого файла и определяет положение текущей позиции в файле после его открытия:

В том случае, когда некоторый файл формируется в программе, т.е. является "выходным" и открыт со строкой статуса "w", завершить его формирование необходимо вызовом функции закрытия файла *fclose*. В этом случае содержимое буфера ввода-вывода (в т.ч. и последние записи файла) будет перемещено в файл на внешнем носителе.

Если теперь этот сформированный файл необходимо просмотреть (прочесть), изменяют его статус, открывая файл на чтение (со строкой статуса "r"):

```
FILE *f; char file_name[13]; ....  
/* Открытие формируемого файла на запись */  
if((f=fopen(file_name, "w"))==NULL)  
{printf("Файл %s не открыт!!!\n", file_name);  
 exit(-1);  
}  
...  
/* формирование файла */  
fclose(f); /* закрытие выходного файла */  
/* Открытие сформированного файла на чтение */  
if((f=fopen(file_name, "r"))==NULL)  
{printf("Файл %s не открыт!!!\n", file_name);  
 exit(-1);  
}
```

Внимание: Если действия по изменению статуса файла выполняются в программе в различных функциях, то имя файла в операционной системе (первый аргумент функции *fopen*) необходимо передавать из функции в функцию в соответствии со способом задания имени файла.

1) Если имя файла задано как строковая константа в тексте программы, то при открытии файла с различным статусом необходимо повторять значение этой константы, а передачу этого имени между функциями можно опустить.

```
void write_file(FILE* file_out) /* функция создания файла */  
{ if((file_out=fopen("icx.dat", "w"))==NULL) .... }
```

```
void read_file(FILE* file_in) /* функция чтения файла */  
{ if((file_in=fopen("icx.dat", "r"))==NULL) .... }
```

2) Если имя файла задается пользователем при вводе как значение строки, то это имя необходимо передавать из функции в функцию. Здесь возможны два варианта декларации переменной для хранения имени файла:

Вариант 1: имя файла объявлено в виде локальной переменной.

Это имя между функциями передается как аргумент:

```
/* функция задания пользователем имени файла */  
void n_file()  
{ FILE* file_f;  
 char name_f[13]; /* объявление переменной для имени файла */  
 ... /* задание имени файла */  
 write_file(file_f, name_f); /* вызов функции создания файла */  
 read_file(file_f, name_f); /* вызов функции чтения файла */  
 }
```

/* функция создания файла */

```
void write_file(FILE* file_out, char * name_file)  
{ if((file_out=fopen(name_file, "w"))==NULL) .... }  
/* функция чтения файла */  
void read_file(FILE* file_in, char * name_file)  
{ if((file_in=fopen(name_file, "r"))==NULL) .... }
```

Вариант 2: имя файла объявлено в виде внешней переменной.

Это имя используется несколькими функциями:

```
char name_f[13]; /* объявление переменной для имени файла */  
/* функция задания пользователем имени файла */  
void n_file()  
{ FILE* file_f;  
 ... /* задание имени файла */  
 write_file(FILE* file_f); /* вызов функции создания файла */  
 read_file(FILE* file_f); /* вызов функции чтения файла */  
 }  
/* функция создания файла */  
void write_file(FILE* file_out)  
{ if((file_out=fopen(n_file, "w"))==NULL) .... }
```

```

/* функция чтения файла */
void read_file(FILE* file_in)
{
    if((file_in = fopen(name_f, "r"))==NULL) .... }

3) Если имя файла задано как значение аргумента функции
main, то это имя передается между функциями как аргумент:
main (int argc, char * argv[])
{
FILE *file_f; ...
/* вызов функции создания файла */
write_file(FILE* file_f, argv[1]);
/* вызов функции чтения файла */
read_file(FILE* file_f, argv[1]);
}

```

Функции `write_file` и `read_file` аналогичны варианту локальной переменной из способа 2.

8.1.3 Использование значений, возвращаемых функциями ввода-вывода

До сих пор мы вызывали функции форматированного ввода-вывода как функции, не возвращающие значения. Особенностью обработки такой структуры данных как *файл* является то, что:

- количество “записей” в нем заранее неизвестно;
- метод доступа позволяет обратиться к “записи” с номером *n* только после того, как будут обработаны предыдущие (*n-1*) “запись”.

Поэтому при обработке файлов последовательного доступа (его чтении) возможно использование только цикла с неизвестным числом повторений до достижения ситуации “конец файла”. Функция форматированного ввода из файла `fscanf` возвращает значение EOF, когда текущая позиция в файле будет установлена за последней “записью”. Это значение следует использовать как условие завершения цикла `while`. Так распечатывается содержимое файла-результата из примера 8.2:

```

while (fscanf(fileout, "%d%s%s",
&str.year, str.name, str.author)!=EOF)
printf("%d%20s%25s\n", str.year, str.name, str.author);

```

8.1.4 Особенности обработки файлов последовательного доступа

Рассмотрим приемы алгоритмизации и программирования для решения задачи о нахождении в файле “записи”, соответствующей минимальному (максимальному) значению некоторого из членов “записи”. В отличие от структуры данных “таблица” (см. тема 6) необходимо воспользоваться промежуточной переменной для хранения значения “записи”, а не ее порядкового номера в файле. Для примера 8.2 поиск самого раннего из географических открытых выглядит следующим образом.

```

/* описания структурных типов для “записей” исходного файла
и файла-результата:*/
struct icx
{
char name [20]; char author[25]; int year1; int year2; };
struct vix
{
int year; char name [20]; char author[25]; };
struct icx t; /* Промежуточная переменная-структура для
обработки текущей “записи” исходного файла */
struct vix vmin; /* Промежуточная переменная-структура для
определения самого раннего из географических открытых */
int min; /* переменная для хранения минимального года */
/* читаем первый элемент (min равен первому элементу): */
fscanf(filein, "%s%s%d%d", t.name, t.author, &t.year1, &t.year2);
/* копируем значение года year1 в переменную */
/* для поиска минимума по времени: */
min=t.year1;
/* заполняем структуру для самой ранней даты */
vmin.year = t.year1;
strcpy(vmin.name,t.name);
strcpy(vmin.author,t.author);
/* Просмотр остальных “записей” файла: */
while (fscanf(filein, "%s%s%d%d",
t.name, t.author, &t.year1, &t.year2) != EOF)
{ /* min - определение самого раннего года */

```

```

if(t.year1<min)
{ min=t.year1; /* Запоминаем самый ранний год */
/* Заполняем структуру для самой ранней даты */
vmin.year=t.year1;
strcpy(vmin.name,t.name);
strcpy(vmin.author,t.author);
}
}
}

```

8.2 Пример

Известны названия и даты великих географических открытий, а также имена авторов этих открытий. Исходные данные хранятся в файле последовательного доступа в следующем виде:

Открытие	Авторы	Год начала путешествия	Год окончания путешествия
Багамы, Куба, Гаити	Колумб	1492	1493
Антарктида	Лазарев, Беллинсгаузен	1819	1821
Бразилия	Педру Алвариш Кабрал	1500	1500
Австралия	Джеймс Кук	1768	1771
Магелланов пр.	Магеллан	1519	1521
Алеутские о-ва	Витус Беринг	1733	1741

Сформировать файл, который содержит информацию о самом раннем и самом позднем из географических открытий в следующем виде: год, название географического открытия; его автор. Имя исходного файла данных задается пользователем в диалоге с программой, а имя файла-результата передается аргументом в командной строке при вызове исполнимого файла программы.

8.2.1 Постановка задачи

8.2.1.1 Исходные данные

nameicx: строка[1..13];	{имя исходного файла}
namevix: строка[1..13];	{имя выходного файла}

filein : файл; {входной файл со следующей структурой:
name: строка [1..20]; {название}
author: строка[1..25]; {автор}
year1: целое; {год начала}
year2: целое; {год окончания} }

8.2.1.2 Ограничения

длина(nameicx) ≠ 0 {длина имени входного файла ≠ 0}
длина(namevix) ≠ 0 {длина имени выходного файла ≠ 0}
∃ filein {существует входной файл и он не пуст}

8.2.1.3 Результаты

fileout: файл; {выходной файл со следующей структурой:
информация о самом раннем открытии
year_min: целое; {год открытия}
name_min: строка [1..20]; {название}
author_min: строка[1..25]; {автор}
информация о самом позднем открытии
year_max: целое; {год открытия}
name_max: строка [1..20]; {название}
author_max: строка[1..25]; {автор} }

8.2.1.4 Связь

∃ i ∈ 1,N (fileout.year_min = filein.year_i,
fileout.name_min = filein.name_i,
fileout.author_min = filein.author_i)
min(filein.year_i)
∃ i ∈ 1,N (fileout.year_max = filein.year_i,
fileout.name_max = filein.name_i,
fileout.author_max = filein.author_i)
max(filein.year_i)

где N - количество записей в файле filein.

8.2.2 Программа на языке Си

```

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
struct icx /* СТРУКТУРНЫЙ ТИП ДЛЯ ДАННЫХ ВХОДНОГО ФАЙЛА*/
{
    char name [20]; /*название*/    char author[25]; /*автор*/
    int year1; /*год начала*/    int year2; /*год окончания*/
};

```

```

struct vix      /* СТРУКТУРНЫЙ ТИП ДЛЯ ДАННЫХ ВЫХОДНОГО ФАЙЛА*/
{
    int year;          /* год начала */
    char name [20];   /* название */
    char author[25];  /* автор */
}
/* строка для отчеркивания строки таблицы при выводе входного файла */
char ss1[59]={"-----";
/* строка для отчеркивания строки таблицы при выводе выходного файла*/
char ss2[60]={"-----";
main(int argc, char * argv[])
{/* ====== прототипы функций, вызываемых из main ====== */
void read_print_icx(FILE* );
void read_print_vix(FILE* );
void job(FILE*, FILE* );
/* ====== ИСХОДНЫЕ ДАННЫЕ ====== */
FILE * filein;
char nameicx[13];
/* ====== ВЫХОДНЫЕ ДАННЫЕ ====== */
FILE *fileout;
/* ====== ПРОВЕРКА ОГРАНИЧЕНИЙ: задано ли в командной ======
строке имя выходного файла ? */
if (argc == 0)
    {printf("При вызове укажите имя выходного файла !!!\n");
     exit(-1);
    };
/* ====== ОТКРЫТИЕ ФАЙЛОВ ====== */
if((fileout=fopen(argv[1],"w"))==NULL) /*Выходной файл - на запись */
    {printf("Файл %s не открыт!!!\n", argv[1]);
     exit(-1);
    }
/* ====== Входной файл - на чтение ====== */
printf("Введите имя входного файла\n");
scanf("%s", nameicx);
if((filein=fopen(nameicx, "r"))==NULL)
    {printf("Файл %s не открыт!!!\n", nameicx);
     exit(-1);
    }
/* ====== ИСПОЛНИМАЯ ЧАСТЬ АЛГОРИТМА ====== */
read_print_icx(filein); /* чтение и печать исходного файла */
fclose(filein);
getch();

```

```

/* ====== Повторное открытие входного файла на чтение =====*/
if((filein=fopen(nameicx,"r"))==NULL) /*текущая позиция - снова в начале*/
    {printf("Файл %s не открыт!!!\n", nameicx);
     exit(-1);
    }
job(filein, fileout);           /* решение задачи*/
fclose(filein);                /* закрытие входного файла */
fclose(fileout);               /* закрытие выходного файла */
/* ====== Открытие сформированного выходного файла на чтение =====*/
if((fileout=fopen(argv[1], "r"))==NULL)
    {printf("Файл %s не открыт!!!\n", argv[1]);
     exit(-1);
    }
/* ====== вывод на экран сформированного файла-результата */
read_print_vix(fileout);
fclose(fileout);                /* закрытие выходного файла */
getch();
/* ====== вывод выходного файла на экран ===== */
void read_print_vix (FILE* fileout)
{struct vix str;
 void shapka2();
 printf("ВЫХОДНОЙ ФАЙЛ:\n");
 shapka2(); /* вывод шапки таблицы выходного файла*/
 while (fscanf(fileout, "%d%s%s",
 &str.year, str.name, str.author)!= EOF)
     printf("%d %s %s\n", str.year, str.name, str.author);
     printf("%s\n",ss2);
}
/* ====== вывод шапки таблицы для выходного файла ===== */
void shapka2()
{printf("%s\n",ss2);
 printf("!! Год!    Открытие !    Автор    !\n");
 printf("%s\n",ss2);
}
/* ====== Вывод входного файла на экран ===== */
void read_print_icx(FILE* filein)
{struct icx str;
 void shapka1();
 printf("ВХОДНОЙ ФАЙЛ:\n");

```

```

shapka1();
while (fscanf(filein, "%s%s%d%d",
    str.name, str.author, &str.year1, &str.year2)!= EOF)
    printf("%-20s%-25s%ld%ld\n",
        str.name, str.author, str.year1, str.year2);
printf("%s\n",ss1);
}
/* ===== вывод шапки таблицы для входного файла ===== */
void shapka1()
{printf("%s\n",ss1);
printf("! Открытие ! Автор ! Год! Год!\n");
printf("%s\n",ss1);
}
/* Функция нахождения самого раннего и самого позднего из
географических открытий */
void job(FILE* filein, FILE* fileout)
{ struct icx t;
/* Структуры для определения самого раннего (vmin) и самого позднего
(vmax) из географических открытий */
    struct vix vmin, vmax;
    int min, max;           /* годы им соответствующие */
/* ===== min и max равны первому элементу:===== */
/* читаем первый элемент */
fscanf(filein, "%s%s%d%d", t.name, t.author, &t.year1, &t.year2);
min=max=t.year1;
/* == заполняем структуру для самой ранней даты открытия ==*/
vmin.year=t.year1;
strcpy(vmin.name,t.name);
strcpy(vmin.author,t.author);
/*== заполняем структуру для самой поздней даты открытия* ==
vmax.year=t.year1;
strcpy(vmax.name,t.name);
strcpy(vmax.author,t.author);
/* Просмотр остальных */
while (fscanf(filein, "%s%s%d%d",
    t.name, t.author, &t.year1, &t.year2)!= EOF)
    { /* min - определение самого раннего года */
if (t.year1<min)
    { /* Запоминаем самый ранний год */
min=t.year1;
/* Заполняем структуру для самой ранней даты */
vmin.year=t.year1;
}
}
}

```

```

strcpy(vmin.name,t.name);
strcpy(vmin.author,t.author);
}
/* max - определение самого позднего года */
if (t.year1>max)
    { /* Запоминаем самый поздний год*/
max=t.year1;
/* Заполняем структуру для самой поздней даты */
vmax.year=t.year1;
strcpy(vmax.name,t.name);
strcpy(vmax.author,t.author);
}
}
/* Все найдено => формируем выходной файл */
sprintf(fileout, "%d %-20s %-25s\n",
    vmin.year, vmin.name, vmin.author);
sprintf(fileout, "%d %-20s %-25s\n",
    vmax.year, vmax.name, vmax.author);
fclose(fileout);
}

```

8.2.3 Исходные данные для примера

Пусть данные для примера подготовлены в файле с именем *geogr*.

Багамы_Куба_Гаити	Колумб	1492	1493
Антарктида	Лазарев_Беллинсгаузен	1819	1821
Бразилия	Педру_Алвариш_Кабрал	1500	1500
Австралия	Джеймс_Кук	1768	1771
Магелланов_пр.	Магеллан	1519	1521
Алеутские_о-ва	Витус_Беринг	1733	1741

файл
geogr

Внимание: Так как информация читается из файла с помощью функций форматированного ввода, символ пробела служит признаком конца ввода значений для отдельных объектов программы. Поэтому в строках, содержащих несколько слов текста естественного языка, последние разделяются символом подчеркивания ('_').

8.2.4 Вызов программы на исполнение

Пусть исполняемый модуль программы получен в файле с именем *Lab7.exe*, а файл результатов имеет имя *result*. Тогда вызов этой программы на исполнение выполняется в командной строке:

Lab7.exe result

8.2.5 Выходные данные для примера

8.2.5.1 Файл результатов

Файл результатов имеет вид:

1492 Багамы, Куба, Гаити Колумб	файл
1819 Антарктида	<i>result</i>

файл
result

8.2.5.2 Диалог с пользователем

Сообщение программы	Ответ пользователя
Введите имя исходного файла	<i>geogr</i>
ИСХОДНЫЙ ФАЙЛ:	
Открытие	Автор
Багамы, Куба, Гаити	Колумб
Антарктида	Лазарев, Беллинсгаузен
...	
Алеутские о-ва	Витус Беринг
	1733
	1741

<нажатие любой клавиши>

ВЫХОДНОЙ ФАЙЛ:

Год	Автор	Открытие
1492	Колумб	Багамы, Куба, Гаити
1819	Лазарев, Беллинсгаузен	Антарктида

8.3 Задания

Для студентов специальности МИ исходный файл данных подготовить любым текстовым редактором.

Для студентов специальности ПО исходный файл создавать в программе, вводя данные с клавиатуры. Для этого предусмотреть в программе несколько режимов ее работы:

1. Создание исходного файла.
2. Вывод на экран содержимого исходного файла.
3. Обработка исходного файла.
4. Вывод на экран содержимого выходного файла.
5. Конец работы.

и организовать ввод пользователем номера выбранного режима.

Варианты заданий

1. Производство продукции лесохимии в СССР (в тыс. т).

Годы	Живица	Канифоль	Скипидар	Уксусная кислота
1940	61.8	46.2	16.6	11.9
1945	24.8	16.2	3.7	4.3
1950	101.3	67.3	22.1	15.1
1955	129.4	104.9	33.2	29.3

Найти вид продукции, производство которой увеличилось в 1950 г. по сравнению с 1940 г. в наибольшее количество раз. О результате сообщить его название и разницу показателей продукции, произведенной в 1945, 1950 и 1955 гг. по сравнению с 1940 г.

2. Производство видов продукции машиностроения (средства потребления) в СССР.

Виды продукции	Единицы измерения	1913	1928	1940	1955
Часы	млн. штук	0.7	0.9	2.8	10.7
Радиоприемники и телевизоры	тыс. штук	-	-	160.8	4925
Холодильники	тыс. штук	-	-	3.5	151.4
Стиральные машины	тыс. штук	-	-	-	87

Среди видов продукции с единицей измерения “тыс. штук” найти тот, который имеет наибольший рост в 1955 г. по сравнению с 1913 г. (1913 г.=100%). О результате сообщить его название и процент прироста продукции в 1940 г. и 1955 г. по сравнению с 1928 г.

3. Темпы роста продукции машиностроения (1913 г.=1).

Год	США	Англия	Франция	СССР
1913	1	1	1	1
1917	3	1.3	0.6	1.3
1940	5.6	2.3	1	35
1956	17	5	2.4	1844

Среди стран, которые не снизили темпы роста за весь период 1913-1956 гг., найти ту, у которой наименьший рост продукции в 1940 г. О результате сообщить его название и разницу показателей за 1917 г., 1940 г. и 1956 г. по сравнению с 1913 г.

4. Развитие радио и телевидения в СССР (тыс.).

Средства массовой информации	1937	1940	1950	1958
Трансляционные радиоприемники, в т. ч. в сельской местности	2482	4231	7056	16637
	548	917	1645	7463
Радиоприемники, в т. ч. в сельской местности	548	917	1645	7463
	232	410	1049	5625
Телевизоры	-	0.4	15	1992

Найти вид средства массовой информации, у которого процент распространения в сельской местности (по отношению к общему количеству) за все периоды 1937-1958 гг. был минимальным. О результате сообщить его название и найденные проценты роста по всем годам.

5. Темпы роста национального дохода по расчету на душу населения (%).

Годы	СССР	США	Англия	Франция
1913	100	100	100	100
1929	124	116	111	139
1932	188	70	110	120
1937	380	107	123	117
1940	448	117	137	106
1950	775	165	149	134
1955	1210	183	162	159

Найти страну, у которой прирост национального дохода после Второй Мировой войны был самым низким. О результате сообщить название страны и разницу в росте национального дохода по сравнению с СССР после Второй Мировой войны.

6. Смертность на 1 тыс. населения по возрастным группам в СССР.

Возрастная группа	1897	1928	1938	1958
20 - 24	7.6	5.5	4.4	1.8
25 - 29	8.2	6.1	4.7	2.2
30 - 34	8.7	6.3	5.4	2.6
35 - 39	10.3	7.5	6.8	3.1
40 - 44	11.8	9	8.1	4.1
45 - 49	15.7	10.9	10.2	5.4

Найти год, в котором смертность по всем возрастным группам была минимальной. О результате сообщить названия возрастных групп и разницу показателей по возрастным группам по отношению к 1897 г.

7. Добыча полезных ископаемых в Мексике (в тыс. т.).

Вид продукции	1937	1950	1956
Уголь	1242	912	1408
Нефть	7159	10382	13000
Медь	976	1754	3528
Цинк	160	240	248
Серебро	2633	1529	1339
Сера	-	1.1	748

Среди видов полезных ископаемых, добываемых до Второй Мировой войны, найти те, прирост добычи которых за весь период 1937-1956 гг. не снижался. О результате сообщить название и нарастание показателей добычи в 1950 г. и 1956 г. по сравнению с 1937 г.

8. Выпуск продукции машиностроения (1957 г. в % к 1937 г.).

Страны	1937	1950	1955
СССР	100	377	819
США	100	227	323
Англия	100	159	213
ФРГ	100	100	226
Франция	100	112	171
Италия	100	131	200

Среди стран, имевших прирост продукции в 1950 г. по сравнению с 1937 г., найти те, выпуск продукции в которых в 1955 г. превысил соответствующий показатель Италии в наименьшее число раз. О результате сообщить название страны и показатели за 1937 г. и 1955 г.

9. Уровень механизации в промышленности СССР (в %).

Вид работы	1940	1950	1955
Доставка железной руды из забоя	-	79	93
Отбойка угля	94.8	98.7	98.7
Откатка угля и породы	75.2	97.1	99
Валка леса	-	39	85
Вывозка леса	32	56.7	78.4

Для лет, в которых все виды работ были механизированы, найти год с наименьшим показателем. О результате сообщить названия видов работы и прирост показателей в 1955 г. по сравнению с 1950 г.

10. Парк тракторов и сельхозмашин в СССР (в тыс.).

Машины	1929	1941	1957
Тракторы	18	684	1542
Комбайны зерновые	0.002	182	375
Автомобили грузовые	0.7	228	631
Плуги конные	14900	4500	1900

Найти тип машин, парк которых в 1957 г. увеличился по сравнению с 1937 г. в максимальное число раз. О результате сообщить его название и разницу показателей в 1941 г. и 1957 г. по сравнению с 1929 г.

11. Производство важнейших видов продукции промышленности Великобритании.

Вид	Единица измерения	1929	1937	1956
Каменный уголь	млн. тонн	262	244	262
Железная руда	млн. тонн	14	14.4	16.6
Чугун	млн. тонн	7.7	8.6	13.4
Сталь	млн. тонн	10.1	13.2	20.6
Хлопчатобумажные ткани	млн. кв. м	3	3.5	1.5
Серная кислота	млн. тонн	0.9	1	2.3

Среди видов продукции с единицей измерения "млн. тонн" найти тот, который имеет наименьший прирост в 1956 г. по сравнению с 1929 г. О результате сообщить его название и процент прироста продукции в 1937 г. и 1956 г. по сравнению 1929 г.

12. Мировое производство текстильных волокон (в млн. тонн).

Волокно	1940	1957
Хлопок-волокно	6.7	8.13
Шерсть мягкая	1.07	1.32
Лен	0.78	1.43
Джут	2.22	1.82
Шелк-сырец	0.06	0.03
Химическое волокно	1.12	2.9

Среди видов волокон, производство которых не снизилось после Второй Мировой войны по сравнению с 1940 г., найти тот, у которого минимальный прирост продукции в 1957 г. О результате сообщить его название и разницу показателей в 1940 г. и 1957 г.

13. Производство видов промышленной продукции в ГДР.

Продукция	Единица измерения	1935	1955	1957
Электроэнергия	млрд. квт-ч	14	28.7	32.7
Каменный уголь	тыс. тонн	3523	2692	2453
Бурый уголь	млн. тонн	101.1	200.6	212.3
Чугун	тыс. тонн	201.8	1517	1663
Сталь	тыс. тонн	1198.6	2507.5	2894.5
Прокат	тыс. тонн	808.7	1996.7	2446.4

Среди видов продукции с единицей измерения "тыс. тонн", найти тот, который имеет наибольшее падение производства в 1955 г. по сравнению с 1935 г. О результате сообщить его название и разницу показателей в 1955 г. и 1957 г. по сравнению с 1935 г.

14. Доходы государственного бюджета в СССР (млрд. руб.).

	1940	1950	1956
Налог с оборота	105.9	236.1	258.1
Отчисления от прибыли предприятий	21.7	40.4	101.2
Взносы на социальные страхование	8.5	19.6	28.1
Налоги и сборы с населения	9.4	35.8	50.7

Найти вид доходов бюджета, показатель которого увеличился в 1950 г. по сравнению с 1940 г. в наименьшее число раз. О результате сообщить его название и разницу показателей в 1940 г. и 1956 г. по сравнению с 1950 г.

15. Запас газоносных районов СССР (в % к общесоюзовым запасам).

Район	1940	1958
Северный Кавказ	1.5	43.7
Поволжье	14.3	15.9
Украина	58.6	24.1
Азербайджан	-	7.8
Средняя Азия	-	4.3
Сибирь	-	1.9
Коми АССР	24.6	1.8

Среди газоносных районов, разрабатываемых до Второй Мировой войны, найти те, процент запасов которых не снизился в 1958 г. О результате сообщить название и нарастание показателей в 1958 г. по сравнению с 1940 г.

16. Поголовье скота в СССР (млн. голов).

Годы	Крупный рогатый скот	Свиньи	Овцы и куры	Лошади
1916	58.4	23.8	96.3	38.2
1928	66.8	27.2	114.6	36.1
1933	33.5	9.9	37.3	17.3
1938	50.9	25.7	66.6	16.2
1946	47.6	10.6	70	10.7

Среди видов скота, не снизивших поголовье в период 1933-1938 гг., найти тот, у которого показатель сократился в 1946 г. в наибольшее число раз по сравнению с 1928 г. О результате сообщить его название и разницу показателей в период 1916-1928 гг.

17. Охрана материнства и детства в СССР.

Объекты здравоохранения	1941	1958
Койки для детей (тыс.)	85.6	203.7
Консультации детские	5341	7235
Койки для беременных (тыс.)	17.8	149.1
Консультации женские	4557	7287
Места в детских яслях	859	1946

Найти вид объектов здравоохранения, показатель которого увеличился в 1958 г. по сравнению с 1941 г. в наибольшее число раз. О результате сообщить его название, разницу в показателях и процент прироста за период 1941-1958 гг.

18. Производство текстильных волокон в СССР (в тыс. тонн).

Вид волокна	1913	1940	1957
Хлопок-сырец	744	2254	4353
Шерсть грязная	77	120	223
Лен	330	349	381
Химическое волокно	0.1	11	149

Среди видов волокна, увеличение производства которых в 1940 г. по сравнению с 1913 г. составило более 50 %, найти тот, у которого прирост производства за период 1940-1957 гг. наибольший. О результате сообщить его название и прирост показателей в 1940 г. и 1957 г. по сравнению с 1913 г.

19. Грамотность в СССР городского населения (%).

Годы	Мужчины	Женщины	Оба пола
1897	65.5	43.1	55.6
1926	88	73.9	80.9
1939	97.6	91	94.2
1959	99.5	98.1	98.7

Найти год, в котором разница показателей грамотности мужчин и женщин минимальна, а разница между показателями грамотности обоих полов и женщин превысила 3 %. О результате сообщить его название и разницу всех его показателей по отношению к 1897 г.

20. Производство промышленных товаров в СССР.

	Единица измерения	1913	1928	1940	1958
Пассажирские вагоны	тыс. штук	1.065	0.387	1.051	1.782
Вывоз древесины	млн. м ³	27.2	36	117.9	251
Цемент	млн. тонн	1.5	1.8	5.7	33.3
Часы	тыс.штук	700	900	2800	24800
Мыло	тыс. тонн	168	311	700	1365

Среди видов товаров с единицей измерения "тыс.штук" найти тот, производство которого не снизилось за весь период 1913-1958 гг. О результате сообщить его название и процент прироста продукции в 1940 г. и 1958 г. по сравнению с 1928 г.

21. Производство видов промышленной продукции в ФРГ.

	Единица измерения	1936	1955	1957
Каменный уголь	млн. тонн	117.6	130.7	133.2
Чугун	млн. тонн	12.6	16.5	18.4
Сталь	млн. тонн	14.8	21.4	24.5
Алюминий	тыс. тонн	48.7	137.6	154
Азотные удобрения	тыс. тонн	-	763	872
Калийные удобрения	тыс. тонн	-	1697	1642

Среди видов продукции с единицей измерения "тыс.тонн", производство которых началось после Второй Мировой войны, найти вид с наибольшим ростом показателей в период с 1955-1957 гг. О результате сообщить его название и процент изменения показателя в 1957 г. по сравнению с 1955 г.

22. Показатели развития внутренней торговли СССР.

	1940	1950	1955	1958
Объем розничного товарооборота (млрд. руб.)	175.1	359.6	501.9	677.2
Товарные запасы (млрд. руб.)	18.4	64.1	98.9	148.9
Число предприятий розничной торговли (тыс.)	407.2	415.8	487	519.3
Число предприятий общественного питания (тыс.)	87.6	95.4	118.2	130.9
Численность торговых работников (тыс.)	2166	1967	2490	2847

Найти показатель торговли, прирост которого в 1955 г. по сравнению с 1940 г. (в процентах) минимален. О результате сообщить его название, единицу измерения и абсолютный прирост показателя за период 1940-1950 гг.

23. Парк сельскохозяйственных машин (тыс. шт.).

Страны	Тракторы		Комбайны	
	1950	1957	1950	1957
Англия	309	436	10.5	33
Нидерланды	18	52	1.2	1.9
Дания	18	77	0.4	3.4
Италия	57	188	-	1.9
Канада	400	500	90.5	136.9

Найти страну, у которой процент прироста парка комбайнов в 1957 г. по отношению к 1950 г. максимальный. О результате сообщить название страны и абсолютный прирост показателей парка тракторов за период 1950-1957 гг.

24. Возрастная структура населения СССР.

Возрастная группа	Число лиц (тыс.)	В % ко всему населению
1939	1959	1939
20 - 24	15786	20343
25 - 29	18520	18190
30 - 34	15598	18999
35 - 39	12958	11590
40 - 44	9603	10408
45 - 49	7776	12264
		4.1
		5.9

Найти возрастную группу, в которой нет прироста числа лиц за период 1939-1959 гг., а уменьшение процента ко всему населению за этот же период максимально. О результате сообщить название группы и показатели за 1959 г.

25. Производство сельскохозяйственных машин в СССР.

Вид	1928	1940	1945	1958
Тракторы	1.3	31.6	7.7	219.7
Комбайны зерновые	-	12.8	0.3	65
Комбайны свеклоуборочные	-	-	-	7.3
Комбайны силосоуборочные	-	-	-	38.1
Косилки	-	3.3	-	73.5
Плуги	0.5	38.4	8	164

Среди видов машин, производимых с 1940 г., найти те, которые производились и в 1945 г. О результате сообщить название и нарастание показателей в период с 1945 - 1958 гг.

26. Площадь и сбор сельскохозяйственных культур в США.

Культуры	Площадь (млн. га)		Сбор (млн. га)	
	1938	1957	1938	1957
Кукуруза	37.8	30.8	53.1	85.4
Пшеница	22.4	18.2	19.5	26.1
Овес	14.1	14.5	14	10.2
Хлопок	11.5	6.2	2.7	2.8
Табак	0.6	0.6	0.6	1
Соя	1	8.1	1.2	11.9

Найти культуру, у которой рост сбора за период 1938-1957 гг. был минимальным. О результате сообщить название культуры и процент изменения площади в 1957 г. по отношению к 1938 г.

27. Количество строительных машин в СССР (шт.).

Наименование	1940	1950	1955	1958
Экскаваторы	2086	5870	17471	28500
Краны передвижные	1135	5642	28900	41743
Скреперы	1100	3000	9200	10500
Бульдозеры	750	3000	16100	29162

Найти вид машин, количество которых увеличилось в 1955 г. по сравнению с 1940 г. в минимальное количество раз. О результате сообщить его название и разницу в показателях в 1955 г. по сравнению с 1950 г.

28. Производство сельскохозяйственных продуктов (млн. тонн).

Продукция	Европа		Азия	
	1953	1957	1953	1957
Зерно	120.5	139.5	284.3	346
Картофель	132.2	143	11.1	30.1
Сахарная свекла	69.1	75.4	2.5	5.6
Хлопок	0.1	0.1	2.1	3.3
Льноволокно	0.2	0.2	-	-

Найти вид продукции, производимой в Азии, у которой процент прироста в 1957 г. по отношению к 1953 г. минимальный. О результате сообщить его название и абсолютный прирост показателей в Европе за период 1953 - 1957 гг.

29. Производство строительных материалов в СССР.

Вид	Единица измерения	1913	1928	1940	1950	1955
Цемент	млн. тонн	1.5	1.8	5.7	10.2	22.5
Гипс	млн. тонн	-	0.23	0.2	1.7	2.9
Кирпич	млрд. штук	2.9	2.8	7.5	10.2	20.8
Стекло	млн. м ²	23.7	34.2	44.7	76.9	99.8
Шифер	млн. плит	9	39	206	546	1488

Среди видов продукции с единицей измерения "млн. тонн" найти тот, производство которого не снизилось в период 1913-1955 гг. О результате сообщить его название и проценты прироста продукции в 1940 г. и 1950 г. по сравнению с 1928 г.

30. Число торговых судов, спущенных на воду.

Страна	1929	1937	1948	1957
Англия	536	309	342	260
Дания	34	26	20	34
Италия	32	6	47	57
Нидерланды	77	112	97	199
Норвегия	57	38	49	83
США	63	123	49	54

Среди стран, увеличивших спуск судов в 1948 г. по сравнению с 1937 г., найти ту, число судов которой в 1957 г. превысило соответствующий показатель США в максимальное число раз. О результате сообщить название страны и показатели за 1929 г. и 1957 г.

31. Численность лиц с образованием (на 1 тыс. человек) в СССР.

Республика	Высшее образование		Среднее образование	
	1939	1959	1939	1959
СССР	6	18	17	263
Украина	7	17	90	286
Белоруссия	4	12	67	229
Грузия	11	38	113	315
Латвия	7	21	140	344
Эстония	8	21	129	304

Найти республику, у которой абсолютный прирост численности лиц с высшим образованием за период 1939-1959 гг. был максимальным. О результате сообщить название республики и процент изменения численности лиц со средним образованием в 1959 г. по отношению к 1939 г.

32. Грамотность в СССР сельского населения (%).

Годы	Мужчины	Женщины	Оба пола
1897	34,3	9,6	21,7
1926	67,3	35,4	50,6
1939	93,7	79,2	86,3
1959	99,1	97,5	98,2

Найти год, в котором разница показателей грамотности мужчин и лиц обоего пола максимальна, а разница между показателями грамотности мужчин и женщин не превысила 20%. О результате сообщить его название и процент прироста его показателей в 1939 г. и 1959 г. по отношению к 1926 г.

33. Медицинские учреждения СССР.

Учреждение	1941	1951	1958
Женские консультации	4557	6500	7207
Детские консультации	5341	6705	7235
Диспансеры противотуберкулезные	554	709	1214
Диспансеры кожно-венерологические	609	644	627
Диспансеры онкологические	28	142	219
Диспансеры психоневрологические	56	71	128

Найти виды медицинских учреждений, количество которых снизилось за весь период 1941-1958 гг. О результате сообщить название этих медицинских учреждений и процент прироста в 1951 г. и 1958 г. по сравнению с 1941 г.

34. Товарная продукция совхозов СССР.

Продукция	1940	1953	1958
Зерно	3674	3677	22460
Хлопок-сырец	131	168	5022
Молоко и молочные продукты	1013	1855	5022
Скот и птица	338	637	1284

Найти вид продукции, производство которой увеличилось в 1953 г. по сравнению с 1940 г. в наибольшее количество раз. О результате сообщить его название и процент прироста в 1958 г. по сравнению с 1953 г.

35. Численность мужчин и женщин на 1 тыс. населения по возрастным группам на 1959 г.

Возраст	Мужчины	Женщины
20 - 24	494	506
25 - 29	490	510
30 - 34	453	547
35 - 39	391	609
40 - 44	384	616
45 - 49	384	616

Найти возрастную группу, в которой процент мужчин в общем составе населения максимальен. О результате сообщить абсолютную и относительную разность превышения женского населения над мужским.

36. Удельный вес видов транспорта в общем грузообороте СССР (в %).

Годы	Железно-дорожный	Морской	Речной	Автомобильный	Нефтепроводный
1913	57,4	17,4	24,8	0,8	0,3
1940	85,1	4,9	7,4	1,8	0,8
1950	84,5	5,6	6,4	2,8	0,7
1955	83,4	5,9	5,8	3,6	1,3

Найти годы, в которые отношения показателей "морской"/"речной" и "автомобильный"/"нефтепроводный" были максимальны. О результате сообщить показатели "морской" и автомобильный".

37. Забастовочное движение в 1950-е гг.

Годы	Италия		Япония	
	число		число	
	забастовок	бастующих	забастовок	бастующих
1950	1250	3537104	584	762453
1951	1178	2134735	576	1162583
1952	1558	1441876	590	1623610
1953	1412	4670091	611	1341220
1954	1990	2045268	647	927921
1955	1981	1383417	650	1033346

Найти год, в котором среднее число бастующих, приходящихся на одну забастовку в Италии превысило соответствующий показатель в Японии в максимальное число раз. О результате сообщить вычисленные показатели по странам.

38. Доля (%) участия в производстве продукции секторов экономики ГДР (1956 г.).

	Народные предприятия	Кооперативные предприятия	Частный сектор
Промышленность	85,2	2,6	12,3
Ремесло	-	0,7	99,3
Строительство	55,7	0,4	43,9
Сельское и лесное хозяйство	15,7	13,6	70,7
Торговля	38,8	43,3	17,8
Транспорт	88,7	-	11,3

Найти отрасли производства, в которых частный сектор составляет менее 50 % и среди них ту, у которой отношение “народные предприятия”/“кооперативные предприятия” максимально. О результате сообщить его название, долю частного сектора и вычисленное отношение.

39. Грамотность населения СССР.

Показатель	Годы	Процент грамотности		
		мужчины	женщины	обоего пола
Все население	1926	66.5	37.1	55.1
	1939	90.8	72.6	81.2
Городское население	1926	85.3	67.6	76.3
	1939	95.7	84.6	89.5
Сельское население	1926	61.9	30	45.2
	1939	88.2	66.3	76.8

Найти показатель, у которого в 1926 г. процент грамотности обоего пола не менее 50 и отношение показателей грамотности “мужчины” /“женщины” было минимально. О результате сообщить его название и все показатели в 1939 г.

40. Количество животных по странам света в 1955 г. (млн. голов).

Страны света	Лошади	Мулы	Ослы	Коровы	Буйволы
Европа	14,7	2	2,8	108	0,6
Азия	14	2,9	19	281	90,9
Африка	3	1,6	9,4	103	1,8
Северная Америка	8,9	3,2	3,2	129	-
Южная Америка	17,3	4,5	3,5	149	-
Австралия	1	-	-	23	-

Найти страны света, на которых представлены все виды животных. О результате сообщить его название и процент лошадей и коров от общего количества животных..

41. Урожайность зерновых культур 1956 г. (п/га).

Культуры	Европа	Азия	Африка	Средняя мировая
Пшеница	15,8	8,6	8,2	11,6
Рожь	15,8	8,7	-	14,3
Ячмень	25,5	11,1	7	13,6
Овес	18,4	10,4	6,2	14,4
Кукуруза	15,5	11,8	9,6	17,2
Просо и сорго	9,4	6,5	-	6,8
Рис	42,3	18,5	14,6	18,6

Найти культуру, у которой соотношение урожайности “континент”/“средняя мировая” будет максимальным. О результате сообщить его название, показатели для Европы и Африки и вычисленное отношение

42. Скорострельность оружия.

Оружие	Техническая	Боевая
Автоматы (пистолеты, пулеметы)	400	60
Магазинные винтовки и карабины	-	10
Автоматические винтовки	-	25
Ручные пулеметы	600	60
Станковые пулеметы	400	150
Крупнокалиберные пулеметы	400	125
Зенитные пушки	120	20

Найти вид оружия, имеющего техническую скорострельность, у которого коэффициент скорострельности (техническая/боевая) минимальный. О результате сообщить его название и показатель боевой скорострельности.

43. Показатели слоистых пластмасс.

Показатели	Текстолит	Гетинакс	Древесные пластины
Плотность	1.3	1.3	1.3
Ударная вязкость	30	15	20
Теплостойкость	125	150	140
Водопоглощение	0.4	0.3	1
Пробивное напряжение	4	15	2.2

Найти вид пластмассы, у которого плотность больше 1, ударная вязкость превышает 15, теплостойкость меньше 200, водопоглощение меньше 1 и максимальное пробивное напряжение. О результате сообщить название пластмассы, показатели плотности и пробивного напряжения.

44. Отраслевая структура обрабатывающей промышленности США.

Отрасль	По числу занятых	По стоимости продукции
Металлургия	7.2	8
Металлообработка	35.1	37.1
Химия	7.8	11.9
Лесообработка	9.7	8.3
Швейная	7.6	4.4
Пищевая	10.5	11.3

Среди отраслей, доля числа занятых которых превышает 9.0, найти отрасль с максимальным отношением “стоимость продукции/число занятых”. О результате сообщить название и вычисленное отношение.

45. Элементы орбит больших планет.

Название	Эксцентриситет	Наклон орбиты	Расстояние от Солнца
Меркурий	0.2056	7.0	0.3871
Земля	0.0167	-	1
Юпитер	0.0484	2.29	0.2028
Уран	0.0472	0.46	19.1910
Плутон	0.247	17.06	39.66

Найти планету с эксцентриситетом больше 0.02, наклоном орбиты более 2 и максимальным расстоянием от Солнца. О результате сообщить название, эксцентриситет и расстояние от Солнца.

46. Данные транспортных самолетов большой дальности.

Тип	Количество двигателей	Взлетная тяга (л.с.)	Взлетный вес (т)
Поршневой	4	2500	45
Турбовинтовой	4	4000	70
Турбореактивный	4	5500	100

Найти тип самолета с взлетным весом не меньше 50т. и максимальной тягой. О результате сообщить название, вес и количество двигателей.

47. Химический состав зерен (%).

Зерно	Вода	Сырой белок	Клетчатка	Зола
Пшеница	13.6	16.8	2	1.8
Рожь	13.5	12.2	2	1.6
Ячмень	13	12	5.5	2.8
Овес	14	11.4	11.4	3.5
Кукуруза	12.5	10.6	8.9	1.4
Рис	11.9	7.9	9.9	5.7

Найти вид зерна, у которого воды больше 13%, белка менее 15% и минимальное содержание золы. О результате сообщить название, содержание клетчатки и золы.

48. Состав российских флотов к началу Крымской войны 1853-56г.г.

Флоты и флотилии	Линейные парусники	Фрегаты	Корветы и бриги	Транспорты	Мелкие суда
Балтийский флот	24	9	8	10	146
Черноморский флот	14	6	16	30	106
Архангельская флотилия	-	-	-	-	10
Каспийская флотилия	-	-	-	2	28
Камчатская флотилия	-	-	-	2	6

Найти флот (флотилию), в составе которой имелись транспорты и было максимальное количество судов. О результате сообщить название, количество фрегатов и общее количество судов.

49. Содержание витаминов в растениях (в мг на 100 г).

Название	A	B1	B2	PP	C
Картофель	0,001	0,1	0,05	0,9	10
Капуста цветная	0,001	0,11	0,10	0,6	70
Морковь	9	0,06	0,06	0,4	5
Лук репчатый	0,03	0,06	0,04	0,2	10
Помидор	2	0,03	0,04	0,5	40
Салат	0,12	0,04	0,08	0,2	7

Найти элементы таблицы, у которых содержания витамина А больше 0.001 мг, витамина С больше 5 мг и отношение содержимого витаминов В1/В2 минимальное. О результате сообщить название, содержимое витаминов РР и вычисленное отношение.

50. Даты возникновения математических знаков

Знак	Значение	Автор	Год
log	Логарифм	Б.Кавальери	1632
sin	Синус	Л.Эйлер	1748
cos	Косинус	Л.Эйлер	1748
tg	Тангенс	Л.Эйлер	1753
f(x)	Функция	Л.Эйлер	1734
dx...	Дифференциал	Г.Лейбниц	1684
∫dx	Интеграл	Г.Лейбниц	1686

Найти самый ранний знак, созданный Эйлером. О результате сообщить название знака, его значение и год создания.

51. Производство газа в Европе (1955г.).

Страна	Газовая промышленность	Доменный газ	Природный и попутный газ
Австрия	280	186	860
Франция	2595	-	270
Нидерланды	845	110	98
Румыния	-	-	5657
Швеция	342	4	-
ФРГ	2880	14556	232

Найти страны, у которых производство газа в газовой промышленности более 300 млн.м³ и среди них найти те страны, у которых отношение "доменный газ" / "природный и попутный газ" минимально.

52. Число жителей, на которых приходится один выборщик в Государственную Думу России (начало XX века).

Курии.	По закону 11.12.1905	По закону 3.06.1907
Земледельческая	2000	230
1-я городская	4000	1000
2-я городская	4000	15000
Крестьянская	30000	60000
Рабочая	90000	125000

Найти курии с максимальным увеличением показателя за период 1905 - 1907 гг.

53. Определения размеров земного эллипсоида по градусным измерениям.

Автор	Год	Размер экватора (м)	Полярный радиус (м)	Сжатие
Бессель	1841	6377397	6256079	1:299.15
Кларк	1866	6378206	6356584	1:294.98
Гельмерт	1906	6378200	6356818	1:298.30
Хейрорд	1909	6378388	6356909	1:296.96
Красовский	1940	6378245	6356863	1:298.30

Найти автора, который в XX веке определил размеры земного эллипсоида как максимальные.

8.4 Контрольные вопросы к лабораторной работе

- 1) Организация передачи аргумента в функцию *main*.
- 2) Организация проверки ограничений для имен файлов, задаваемых пользователем.
- 3) Организация изменения статуса файла.
- 4) Особенности организации данных в файлах последовательного доступа, обрабатываемых функциями форматированного ввода-вывода.

8.5 Содержание отчета

1. Условие задачи.
2. Постановка задачи.
3. Метод решения задачи.
4. Алгоритмы (вспомогательные и основной) и их иерархия.
5. Таблица трассировки программы.
6. Текст программы на языке Си.
7. Содержимое исходного файла данных и файла результатов.
8. Выводы (с сравнением решения задачи с помощью таблиц).
9. Программный документ "Описание программы".

9 ЛАБОРАТОРНАЯ РАБОТА N 8. ГРАФИЧЕСКИЕ СРЕДСТВА TURBO-C

Цель работы: освоить приемы работы с графическими средствами системы TURBO-C на примере построения статических изображений, их анимации и вывода текста в графическом режиме.

Контрольные вопросы:

- 1) Назовите режимы работы монитора ПЭВМ.
- 2) Как выполняется в TURBO-C инициализация и завершение работы с графической библиотекой?
- 3) Назовите параметры «пера» и «кисти».
- 4) Какие действия допустимы с точкой (пикселям)?
- 5) Какие типы прямых и кривых можно генерировать средствами графической библиотеки?
- 6) Назовите средства TURBO-C для генерации примитивов с закраской.
- 7) Назовите средства TURBO-C для анимации изображений.

9.1 Примеры алгоритмизации и программирования при работе с графическими средствами

9.1.1 Некоторые понятия относящиеся к графике

Растровая графика - это способ представления графической информации, при котором изображение представлено прямоугольной матрицей точек (пикселей).

Каждая из точек может иметь свой цвет из заданного набора набора цветов, называемого палитрой. Любая графическая операция сводится к работе с отдельными пикселями.

Видеoadаптер - это компонент аппаратуры компьютера, который хранит изображение в своей видеопамяти и отображает ее содержимое на экране монитора.

Разрешение - это размер матрицы пикселей.

Размер палитры - это количество цветов, которые можно одновременно отображать на экране.

Типы видеoadаптеров для ПЭВМ типа IBM PC/AT и PS/2 и их разрешение:

CGA	- 320x200
EGA	- 640x200 или 640x350
VGA	- 640x200 или 640x350 или 640x480
Hercules	- 720x348
SVGA	- 1024x768

Режимы работы видеоадаптера отличаются друг от друга разрешением и размером палитры. Перечисленные выше типы видеоадаптеров поддерживают несколько режимов работы.

Основные группы графических объектов, представляющие собой объединения пикселей:

- линейные изображения (образы прямых и кривых линий);
- сплошные объекты (образы двумерных областей);
- шрифты;
- картинки, англ. *images* (прямоугольные матрицы пикселей).

Графическая библиотека языка программирования - это библиотека функций (процедур), поддерживающая работу с основными группами графических объектов и с основными типами видеоадаптеров.

“Перо” используется для рисования различных линий и имеет следующие атрибуты:

- цвет (устанавливается функцией *setcolor()*);
- толщина линии, ее стиль или образец (устанавливаются функцией *setlinestyle()*).

“Кисть” используется для рисования сплошных объектов и имеет следующие атрибуты, которые устанавливаются функцией *setfillstyle()*:

- цвет;
- образец заполнения.

Окно - это своеобразный маленький экран с собственной локальной системой координат, относительно которой работают все функции рисования.

Видеопамять делится на видеостраницы: видимую (на которой пользователь видит кадр) и активную (невидимую, на которой строится изображение следующего кадра). Видеостраницы используются для организации качественной анимации.

9.1.2 Особенности программирования в графическом режиме

Следует помнить:

- если координаты графических объектов выходят за границы установленного размера экрана, то ошибка не идентифицируется и изображение на экране не визуализируется;
- не все стандартные шрифты поддерживают отображение символов кириллицы;
- драйвер графического адаптера *EGAVGA.BGI* должен размещаться в том же каталоге, что и исполняемый файл либо путь к нему должен быть установлен в *initgraph()*.

9.1.3 Основные примеры работы с графической библиотекой TURBO-C

Графическая библиотека к тексту программы на языке Си подключается директивой препроцессора `#include <graphics.h>`.

9.1.3.1 Настройка среды TURBO-C 2.0 на работу с графической библиотекой

Для работы с графической библиотекой необходимо произвести настройку среды TURBO-C 2.0. Для этого в главном меню среды выбирают опцию «*Options*» и нажимают клавишу *ENTER*. В появившемся вертикальном меню первого уровня выбирают опцию «*Linker*» и снова нажимают клавишу *ENTER*. В следующем появившемся вертикальном меню второго уровня выбирают опцию «*Graphics library*». Нажатием клавиши *ENTER* устанавливают значение «*On*» (включено).

Затем нажимают клавишу *ESC* и закрывают вертикальное меню второго уровня. В меню первого уровня выбирают опцию «*Save options*» и нажимают клавишу *ENTER*. После открытия окна «*Config File*» нажимают клавишу *ENTER*. После появления окна «*Verify*» с текстом «*Overwrite TCCONFIG.TC (Y/N)*» подтвердите выбор (т.е. сохранение настройки на работу с графической библиотекой) нажатием клавиши *Y*.

После выполнения сохранения выполненной настройки вертикальное меню первого уровня автоматически закрывается и курсор возвращается в главное меню среды Turbo-C.

9.1.3.2 Функции графической библиотеки TURBO-C

Все функции, приведенные ниже, используют именованные константы. Их определения приведены в заголовочном файле *GRAPHICS.H*, а их значения в таблицах приложения Б.

- **ARC** - Начертить дугу.

Синтаксис `void far arc(int x, int y, int stangle, int endangle, int radius);`

Описание Чертит текущим цветом дугу окружности на графическом дисплее.

Параметры

int x, int y - Координата центра.
int stangle - Начальный угол в градусах.
int endangle - Конечный угол в градусах.
int radius - Радиус от центра в пикселях.

- **BAR** - Нарисовать прямоугольник.

Синтаксис `void far bar(int left, int top, int right, int bottom);`

Описание Рисует закрашенный прямоугольник на экране графического дисплея. Прямоугольник закрашивается текущим шаблоном заполнения и цветом.

Параметры

int left, int top - Координата верхнего левого угла прямоугольника.
int right, int bottom - Координата нижнего правого угла прямоугольника.

- **BAR3D** - Нарисовать прямоугольный параллелепипед.

Синтаксис `void far bar3d(int left, int top, int right, int bottom, int depth, int topflag);`

Описание Рисует прямоугольный параллелепипед, аналогичный `bar()`, но имеющий моделируемые глубину, ширину и высоту.

Параметры

int left, int top - Координата верхнего левого угла прямоугольника.
int right, int bottom - Координата нижнего правого угла прямоугольника.
int depth - Управляет глубиной моделируемого трехмерного эффекта.

int topflag - Устанавливается равным нулю для изображения параллелепипеда без верхушки и ненулевому значению, если нужно рисовать верхушку.

- **CIRCLE** - Начертить окружность

Синтаксис `void far circle(int x, int y, int radius);`

Описание Чертит окружность на экране графического дисплея.

Параметры

int x - Центр X-координаты окружности.
int y - Центр Y-координаты окружности.
int radius - Радиус окружности (расстояние в пикселях от центра до окружности).

- **CLEARDEVICE** - Очистить графический экран

Синтаксис `void far cleardevice(void);`

Описание Очищает графический экран, закрашивая его текущим цветом фона и перемещая курсор в позицию (0,0).

- **CLEARVIEWPORT** - Очистить область отображения

Синтаксис `void far clearviewport(void);`

Описание Очищает текущую область отображения, закрашивая ее цветом фона, и перемещает текущий указатель (курсор) в точку с относительной координатой (0,0)

- **CLOSEGRAPH** - Закрыть графическую систему

Синтаксис `void far closegraph(void);`

Описание Закрывает графику BGI. Освобождает все внутренние структуры памяти, память, отведенную под шрифты, драйверы, буфер заполнения заливкой (создаваемый вызовом `setgraphbufsize()`) и т.д. Кроме того, восстанавливает режим работы дисплея, действующий до вызова `initgraph()`.

- **DETECTGRAPH** - Определить графические режимы

Синтаксис `void far detectgraph(int far *graphdriver, int far *graphmode);`

Описание Определяет графические возможности системы. Эта функция всегда должна использоваться перед инициализацией BGI-графики. При заданном выборе графических режимов эта функция выбирает самое высокое допустимое разрешение, которое может дать наилучшие результаты.

Параметры

int far *graphdriver - Указатель на целое. Должен быть равен нулю, если требуется автоматическое определение графических возможностей. Ненулевое значение означает выбор идеального или заданного графического режима.

int far *graphmode - Указатель на целое. Если аргумент graphdriver равен нулю, в режиме *graphmode устанавливается значение самого высокого возможного разрешения для графического дисплея системы. Если graphdriver не равен нулю, *graphmode задаёт требуемый режим для адаптера дисплея, поддерживающего несколько режимов.

• **DRAWPOLY** - Начертить контур многоугольника

Синтаксис `void far drawpoly(int numpoints, int far *polypoints);`

Описание Вычерчивает на экране графического дисплея многоугольник с заданным числом точек, используя текущий стиль линии, цвет и режим. Не замыкает многоугольник автоматически. Чтобы получить замкнутую фигуру, последняя точка должна совпадать с первой.

Параметры

int numpoints - Количество пар координат (x,y) в массиве, адресуемом аргументом polypoints.

int far *polypoints - Указатель на массив пар координат (x,y). Каждое значение в этом массиве должно быть типа int. Весь массив должен иметь размер по крайней мере numpoints *2 * sizeof(int) байтов.

• **ELLIPSE** - Начертить эллипс или окружность

Синтаксис `void far ellipse (int x,int y,int stangle,int endangle, int xradius,int yradius);`

Описание Вычерчивает эллиптическую дугу на экране графического дисплея. Контур дуги вычерчивается толщиной и цветом текущей линии. За исключением толщины, стиль текущей линии игнорируется (т. е. контур - всегда сплошной).

Параметры

int x,int y (x,y) - Координата центра дуги.

int stangle - Начальный угол в градусах, от 0 до 360. Чтобы начертить полный эллипс, установить stangle равным 0, а endangle - равным 360. Углы отсчитываются против часовой стрелки. Нулевые углы соответствуют положению стрелки часов, указывающей на 3 часа.

int endangle - Конечный угол дуги.

int xradius - Горизонтальное расстояние от центра до дуги в пикселях.

int yradius - Вертикальное расстояние от центра до дуги в пикселях.

• **FILLELLIPSE** Начертить закрашенный эллипс

Синтаксис `void far fillellipse(int x,int y,int xradius,int yradius);`

Описание Рисует на экране графического дисплея закрашенный эллипс, используя текущий цвет и шаблон заполнения.

Параметры

int x,int y (x,y) - координата центра.

int xradius - Горизонтальный радиус.

int yradius - Вертикальный радиус.

• **FILLPOLY** - Начертить закрашенный многоугольник

Синтаксис `void far fillpoly(int numpoints,int far *polypoints);`

Описание Рисует на экране графического дисплея закрашенный многоугольник, используя текущий цвет и шаблон заполнения. Для контура многоугольника используется текущий цвет и стиль линий.

Параметры

int numpoints - Число вершин многоугольника.

int far *polypoints - Указатель на массив 16-битовых целых, каждая пара которых представляет (x,y)-координату. В этом

массиве должно быть по крайней мере `numpoints` пар целых чисел.

- **FLOODFILL** - Закрасить графическую область.

Синтаксис `void far floodfill(int x, int y, int border);`

Описание Закрашивает область произвольной формы на графическом экране текущим цветом и шаблоном заполнения.

Параметры

`int x, int y` - Координата начальной точки заполнения.

`int boorder` - Цвет границы заполняемой области. (`x, y`) - координата начальной точки должна находиться внутри этой области.

- **GETBKCOLOR** - Определить цвет графического фона.

Синтаксис `int far getbkcolor(void);`

Описание Возвращает значение цвета графического фона.

- **GETCOLOR** - Определить цвет вычерчивания.

Синтаксис `int far getcolor(void);`

Описание Возвращает текущий цвет вычерчивания графических изображений.

- **GETDRIVENAME** - Получить имя графического драйвера.

Синтаксис `char *far getdrivename(void);`

Описание Возвращает указатель на имя текущего графического драйвера, запомненное в виде ASCII-строки.

- **GETGRAPHMODE** - Получить текущий графический режим.

Синтаксис `char far getgraphmode(void);`

Описание Возвращает текущий графический режим, установленный вызовом функции `initgraph()`.

- **GETIMAGE** - Копировать битовый образ в память.

Синтаксис `void far getimage(int left, int top, int right, int bottom, void far *bitmap);`

Описание Копирует часть графического экрана в буфер, адресуемый аргументом `bitmap`. Копируемый прямоугольный образ определяется четырьмя параметрами типа `int`. Из памяти графического дисплея копируются пиксели вместе с информацией о ширине и высоте.

Параметры

`int left, int top` - Верхняя левая (`X, Y`) - координата образа.

`int right, int bottom` - Нижняя правая (`X, Y`) - координата образа.

`void far *bitmap` - Указатель на буфер достаточно большого размера для запрошенного образа битовой карты.

- **GETLINESETTINGS** - Получить текущие параметры линии.

Синтаксис `void far getlinesettings(struct linesettingstype far *lineinfo);`

Описание Заполняет структуру типа `linesettingstype` текущими значениями параметров графической линии - стилем, шаблоном и толщиной. Эта структура объявлена в следующим образом:

```
struct linesettingstype {  
    int linestyle; // Значение enum line_styles//  
    unsigned upattern; // Шаблон линии, если  
                       // linestyle== USERBIT_LINE  
    int thickness; // 1-NORM_WIDTH,  
                   // 3-THICK_WIDTH (шириной 3  
                   // пикселя)  
};
```

Параметры

`struct linesettingstype far *lineinfo` - Указатель на структуру типа `linesettingstype`.

- **GETMAXCOLOR** - Возвращает максимальное значение цвета.

Синтаксис `int far getmaxcolor(void);`

Описание Возвращает максимальное значение графического цвета.

массиве должно быть по крайней мере `numpoints` пар целых чисел.

- **FLOODFILL** - Закрасить графическую область.

Синтаксис `void far floodfill(int x, int y, int border);`

Описание Закрашивает область произвольной формы на графическом экране текущим цветом и шаблоном заполнения.

Параметры

`int x, int y` - Координата начальной точки заполнения.

`int boorder` - Цвет границы заполняемой области. (`x, y`) - координата начальной точки должна находиться внутри этой области.

- **GETBKCOLOR** - Определить цвет графического фона.

Синтаксис `int far getbkcolor(void);`

Описание Возвращает значение цвета графического фона.

- **GETCOLOR** - Определить цвет вычерчивания.

Синтаксис `int far getcolor(void);`

Описание Возвращает текущий цвет вычерчивания графических изображений.

- **GETDRIVENAME** - Получить имя графического драйвера.

Синтаксис `char *far getdrivename(void);`

Описание Возвращает указатель на имя текущего графического драйвера, запомненное в виде ASCII-строки.

- **GETGRAPHMODE** - Получить текущий графический режим.

Синтаксис `char far getgraphmode(void);`

Описание Возвращает текущий графический режим, установленный вызовом функции `initgraph()`.

- **GETIMAGE** - Копировать битовый образ в память.

Синтаксис `void far getimage(int left, int top, int right, int bottom, void far *bitmap);`

Описание Копирует часть графического экрана в буфер, адресуемый аргументом `bitmap`. Копируемый прямоугольный образ определяется четырьмя параметрами типа `int`. Из памяти графического дисплея копируются пиксели вместе с информацией о ширине и высоте.

Параметры

`int left, int top` - Верхняя левая (X, Y) - координата образа.

`int right, int bottom` - Нижняя правая (X, Y) - координата образа.

`void far *bitmap` - Указатель на буфер достаточно большого размера для запрошенного образа битовой карты.

- **GETLINESETTINGS** - Получить текущие параметры линии.

Синтаксис `void far getlinesettings(struct linesettingstype far *lineinfo);`

Описание Заполняет структуру типа `linesettingstype` текущими значениями параметров графической линии - стилем, шаблоном и толщиной. Эта структура объявлена в следующим образом:

```
struct linesettingstype {  
    int linestyle; // Значение enum line_styles//  
    unsigned upattern; // Шаблон линии, если  
                      // linestyle== USERBIT_LINE  
    int thickness; // 1-NORM_WIDTH,  
                  // 3-THICK_WIDTH (ширина 3  
                  // пикселя)  
};
```

Параметры

`struct linesettingstype far *lineinfo` - Указатель на структуру типа `linesettingstype`.

- **GETMAXCOLOR** - Возвращает максимальное значение цвета.

Синтаксис `int far getmaxcolor(void);`

Описание Возвращает максимальное значение графического цвета.

- **GETMAXX, GETMAXY** - Получить максимальные графические координаты.

Синтаксис `int far getmaxx(void);`
`int far getmaxy(void);`

Описание Возвращают максимальные значения графических координат X (`getmaxx()`) и Y (`getmaxy()`). Значения берутся относительно 0, поэтому если `getmaxx()` возвращает 639, то по горизонтали имеется 640 пикселей.

- **GETPIXEL** - Получить цвет пикселя.

Синтаксис `int far getmaxcolor(int x, int y);`

Описание Возвращает значение, которое представляет цвет пикселя с заданной координатой.

Параметры

`int x, int y` - Координата пикселя.

- **GETTEXTSETTINGS** - Получить информацию о графическом шрифте.

Синтаксис `void far gettextsettings(struct textsettingstype far *texttypeinfo);`

Описание Заполняет структуру типа `textsettingstype` различной информацией о параметрах отображения текстовых строк на графическом дисплее. Эта структура объявлена в следующим образом:

```
struct textsettingstype {
    int font; // Идентификатор шрифта BGI
    int direction; // Ориентация отображения текста
    int charsize; // Размер, переданный settextstyle()
    int horiz; // Горизонтальное выравнивание
    int vert; // Вертикальное выравнивание
};
```

Параметры

struct linesettingstype far *lineinfo - Указатель на структуру типа `textsettingstype`, которая будет содержать результат функции.

- **GETVIEWSETTINGS** - Получить информацию о текущей области вывода.

Синтаксис `void far getviewsettings(struct viewporttype far *viewport);`

Описание Заполняет структуру типа `viewporttype` информацией о текущей области вывода графического дисплея. Эта структура объявлена в GRAPHICS.H следующим образом:

```
struct viewporttype {
    int left, top, right, bottom; // Прямоугольник области вывода
    int clip; // Вертикальное выравнивание
};
```

Параметры

struct viewporttype far *viewport - Указатель на структуру типа `viewporttype`, заполняемую функцией.

- **GETX, GETY** - Получить графическую координату.

Синтаксис `int far getx(void);`
`int far gety(void);`

Описание Взятые вместе эти функции возвращают текущую позицию графического дисплея - (X,Y) - координату изображения для большинства функций графического отображения и вывода графического текста.

- **GRAPHERRORMSG** - Получить сообщение об ошибке графической системы.

Синтаксис `char far grapherrormsg(int ErrorCode);`

Описание Возвращает указатель на строку описания указанного кода ошибки.

Параметры

int ErrorCode - Код ошибки, аналогичный возвращаемому функцией `graphresult()`.

- **GRAPHRESULT** - Получить код ошибки графической системы.

Синтаксис `char far graphresult(void);`

Описание Эту функцию следует вызывать для проверки наличия ошибок после использования одной из функций графической библиотеки. Возвращенное значение нужно запомнить в переменной типа int, так как после вызова graphresult() внутренний код ошибки графической системы будет установлен равным grOK (что означает отсутствие ошибок). Чтобы получить текстовое описание полученного кода ошибки, его можно передать функции grapherrmsg().

- **IMAGESIZE** - Получить размер битовой карты.

Синтаксис `unsigned far imagesize(int left, int top, int right, int bottom);`

Описание Возвращает число байтов, требуемых для запоминания копии отображаемого графического образа, прямоугольный контур которого задан полученными аргументами. Если для запоминания графического образа требуется более одного сегмента памяти размером 64K, imagesize() возвращает -1.

Параметры

- int left, int top - Координата верхнего левого угла образа.
- int right, int bottom - Координата нижнего правого угла образа.

- **INITGRAPH** - Инициализировать графическую систему.

Синтаксис `void far initgraph(int far *graphdriver, , int far *graphode, char far *pathodriver);`

Описание Инициализирует программное обеспечение графического интерфейса Borland (BGI) и подготавливает видеointерфейс для отображения в графическом режиме. Загружает в память графический драйвер BGI в соответствии с значением одной из следующих констант:

Константа	Значение
DETECT	0 (Запросит автоматическое определение графического режима)
CGA	1
MCGA	2
EGA	3
EGA64	4

EGAMONO	5
IBM8514	6
HERCMONO	7
ATT400	8
VGA	9
PC3270	10

Чтобы определить, правильно ли инициализирована графическая система, после вызова initgraph() вызовите функцию graphresult().

Параметры

int far *graphdriver - Указатель на переменную типа int, обычно инициализированную значением DETECT до вызова функции initgraph(), чтобы выбрать наилучший возможный графический режим. Данная функция присваивает этой переменной значение одной из констант драйвера из таблицы, приведенной в секции Описание. Вы можете также запросить явный графический режим, инициализируя параметр graphdriver значением соответствующей константы драйвера, но тогда вы сами несете ответственность за способность системы поддерживать запрошенный режим отображения.

int far *graphmode - Указатель на переменную типа int, которой initgraph() обычно присваивает значение наивысшего разрешения, поддерживаемого автоматически выбранным или явно заданным драйвером. Вы можете также присвоить этой переменной значение явной константы графического режима драйвера.

char far *pathodriver - Указатель на путь каталога, в котором находятся драйверы BGI. Драйверы могут также находиться в текущем каталоге, и в этом случае pathodriver должен быть установлен равным нулю.

- **LINE** - Начертить линию.

Синтаксис `void far line(int x1, int y1, int x2, int y2);`

Описание Вычерчивает линию с координатами концевых точек (x1, y1) и (x2, y2) текущим цветом, стилем и толщиной. Не изменяет текущую позицию курсора.

Параметры

int x1, int y1 - Координаты начала линии.

int x2, int y2 - Координаты конца линии.

- **LINEREL** - Начертить линию относительно текущей позиции.

Синтаксис void far linerel(int dx, int dy);

Описание Вычерчивает линию от текущей точки с координатами (x, y) до точки, заданной относительно текущей, текущим цветом, стилем и толщиной. Один конец линии имеет координаты (x, y), а другой - координаты (x+dx, y+dy).

Параметры

int dx - Горизонтальное расстояние относительно текущей позиции.

int dy - Вертикальное расстояние относительно текущей позиции.

- **LINETO** - Начертить присоединенную линию.

Синтаксис void far lineto(int x, int y);

Описание Вычерчивает линию от текущей позиции до точки с координатами (x, y) текущим цветом, стилем и толщиной. Устанавливает текущую позицию равной (x, y).

Параметры

int x, int y - Координаты конца линии.

- **MOVEREL** - Переместить текущую позицию.

Синтаксис void far moverel(int dx, int dy);

Описание Перемещает текущую позицию (x, y) в точку с координатами (x+dx, y+dy).

Параметры

int dx - Горизонтальное расстояние относительно текущей позиции.

int dy - Вертикальное расстояние относительно текущей позиции.

- **MOVETO** - Переместить графический указатель.

Синтаксис void far moveto(int x, int y);

Описание Перемещает внутреннюю текущую позицию (СР) в точку области вывода, имеющую координаты (x, y), устанавливая начало отображения для графических функций, таких как lineto().

Параметры

int x, int y - Требуемые координаты СР.

- **OUTTEXT** - Отобразить строку.

Синтаксис void far outtext(char far *textstring);

Описание Отображает в графическом режиме текстовую строку в текущей позиции (СР), используя текущий шрифт, направление, размер, выравнивание и цвет.

Параметры

char far *textstring - Дальний указатель на завершающуюся нулем строку.

- **OUTTEXTXY** - Отобразить строку в заданной позиции.

Синтаксис void far outtextxy(int x, int y, char far *textstring);

Описание Аналогична outtext(), но отображает строку в заданной позиции, имеющей координаты (x, y).

Параметры

int x, int y - Начальная координата.

char far *textstring - Дальний указатель на завершающуюся нулем строку.

- **PUTIMAGE** - Отобразить битовый образ.

Синтаксис void far getimage(int left, int top void far *bitmap, int op);

Описание Копирует сохраненный в виде битовой карты графический образ на дисплей.

Параметры

int left, int top - Координаты верхнего левого пикселя прямоугольника, в который вписан графический образ.

void far *bitmap - Указатель на данные графического образа, записанные посредством вызова getimage().

int op - Один из символов перечисления putimage_ops: COPY_PUT, XOR_PUT, OR_PUT, AND_PUT и NOT_PUT, который выбирает логическую операцию,

используемую функцией `putimage()` для комбинации битов записываемого графического образа и соответствующих битов графического образа, который находится в данный момент на экране дисплея.

- **PUTPIXEL** - Отобразить пиксель.

Синтаксис `void far putpixel(int x, int y, int color);`

Описание Отображает один графический пиксель на экране дисплея в позиции с заданными координатами. Цвет пикселя зависит от текущего видеорежима.

Параметры

int x, int y - Координаты позиции, в которой отображается пиксель.

int color - Предполагаемый цвет пикселя.

- **RECTANGLE** - Начертить прямоугольник.

Синтаксис `void far bar(int left, int top, int right, int bottom);`

Описание Рисует прямоугольник на экране графического дисплея, используя текущий цвет линии, толщину и цвет.

Параметры

int left, int top - Координата верхнего левого угла прямоугольника.

int right, int bottom - Координата нижнего правого угла прямоугольника.

- **SECTOR** - Начертить сектор эллипса.

Синтаксис `void far sector (int x, int y, int stangle, int endangle, int xradius, int yradius);`

Описание Действует аналогично функции `ellipse()`, но заполняет сектор эллипса текущим шаблоном и цветом заполнения. Чтобы изменить цвет контура сектора, необходимо использовать функцию `setcolor()`.

Параметры

См. `ellipse()`.

- **SETBKCOLOR** - Определить цвет графического фона.

Синтаксис `void far setbkcolor(int color);`

Описание Изменяет цвет фона графического дисплея.

Параметры

int color - Значение цвета, используемое как индекс в массиве значений цветов текущей палитры.

- **SETCOLOR** - Изменить цвет вычерчивания.

Синтаксис `void far setcolor(int color);`

Описание Изменяет графический цвет вычерчивания.

Параметры

int color - Номер цвета, используемый как индекс в текущей палитре для определения цвета.

- **SETFILLSTYLE** - Установить шаблон и цвет заполнения.

Синтаксис `void far setfillstyle(int pattern, int color);`

Описание Изменяет битовый образ шаблона заполнения и цвет, используемые различными функциями, обрабатывающими закрашенные фигуры. Возможные шаблоны заполнения:

Перечислимый символ	Значение	Шаблон заполнения
BKSLASH_FILL	5	(Жирные) линии вида \\\
CLOSE_DOT_FILL	11	Близко расположенные точки
EMPTY_FILL	0	Цвет фона
HATCH_FILL	7	Мелкая прямоугольная сетка
INTERLEAVE_FILL	9	Чередующиеся линии
LINE_FILL	2	Линии вида -----
LTBKSLASH_FILL	6	Линии вида \\\\
LTSLASH_FILL	3	Линии вида ////
SLASH_FILL	4	(Жирные) линии вида ////
SOLID_FILL	1	Сплошной цвет
USER_FILL	12	Шаблон пользователя
WIDE_DOT_FILL	10	Редко расположенные точки
XHATCH_FILL	8	Крупная наклонная сетка

Параметры

int pattern - Константа, определяющая шаблон заполнения.

int color - Цвет заполнения фигур.

- **SETLINESTYLE** - Установить ширину и цвет линии.

Синтаксис `void far setlinestyle(int linestyle, unsigned upattern, int thickness);`

Описание Изменяет толщину и битовый шаблон, используемые графическими функциями, вычерчивающими прямые линии. Не воздействует на контур окружностей и эллипсов.

Параметры

int linestyle - Устанавливается равным SOLID_LINE, DOTTED_LINE, CENTER_LINE, DASHED_LINE и USERBIT_LINE.

unsigned upattern - Если параметр linestyle равен USERBIT_LINE, этот параметр должен содержать 16-битовое значение, в котором биты, равные 1, задают изображаемые точки, а биты, равные 0, - точки, через которые просвечивается фон.

int thickness - Устанавливается равным NORM_WIDTH или THICK_WIDTH.

- **SETTEXTJUSTIFY** - Установить выравнивание текста в графическом режиме.

Синтаксис `void far settextjustify(int horiz, int vert);`

Описание Задает позицию вывода текста в графическом режиме относительно текущей позиции курсора (СР). Если параметр horiz равен LEFT_TEXT и текущее направление равно HORIZ_DIR, перемещает СР в позицию за последним пикселям отображаемой строки текста. Влияет на выравнивание текста, отображаемого функциями outtextxy() и outtext(), вызову которых предшествовал вызов функции moveto().

Параметры

int horiz - Одна из констант LEFT_TEXT, CENTER_TEXT или RIGHT_TEXT. Задает горизонтальную позицию строки относительно СР.

int vert - Одна из констант BOTTOM_TEXT, CENTER_TEXT или TOP_TEXT.

- **SETTEXTSTYLE** - Установить стиль текста в графическом режиме.

Синтаксис `void far settextstyle(int font, int direction, int charsize);`

Описание Устанавливает характеристики текста, используемые в последующих вызовах функций outtextxy() и outtext().

Параметры

int font - Одно из имен встроенных шрифтов BOLD_FONT, COMPLEX_FONT, DEFAULT_FONT, EUROPEAN_FONT, GOTHIC_FONT, SANS_SERIF_FONT, SCRIPT_FONT, SIMPLEX_FONT, SMALL_FONT, TRIPLEX_FONT, TRIPLEX_SCR_FONT или имя пользовательского шрифта.

int direction - Либо HORIZ_DIR - для горизонтального отображения текста, либо VERT_DIR - для вертикального отображения текста.

int charsize - Относительный размер символов. Нулевое значение позволяет выбрать размер по умолчанию шрифта (векторного) шрифта или размер, указанный в предшествующем вызове функции setusercharsize().

- **SETUSERCHARSIZE** - Установить размер текста в графическом режиме.

Синтаксис `void far setusercharsize(int multx, int divx, int multy, int divy);`

Описание Изменяет горизонтальный и вертикальный размер текста в графическом режиме. Чтобы получить размер символов, меньший нормального, используйте значение коэффициента масштабирования меньше единицы. Например, если multx/divx равно 0.25, будут отражаться символы ширины, равной одной четверти их нормальной ширины. Если multy/divy равно 3.5, будут отображаться символы, высота которых приблизительно в 3.5 раз больше их нормальной высоты. Чтобы использовать новое значение коэффициента масштабирования, после вызова функции setusercharsize() вызывайте функцию settextstyle() с значением параметра charsize, равным нулю.

Параметры

int multx, int divx - Коэффициент горизонтального масштабирования, равный multx/divx.

int multy, int divy - Коэффициент вертикального масштабирования, равный multy/divy.

- **SETVIEWPORT** - Изменить графическую область вывода.

Синтаксис `void far setviewport(int left, int top, int right, int bottom, int clip);`

Описание Задает графическую область вывода и определяет, нужно ли производить отсечение по границам области вывода.

Параметры

int left, int top - Абсолютная координата верхнего левого угла прямоугольника области вывода.

int right, int bottom - Абсолютная координата нижнего правого угла прямоугольника области вывода.

int clip - 0 - не производить отсечение, ненулевое значение - производить отсечение.

- **SETWRITEMODE** - Установить режим вывода рисования линий.

Синтаксис `void far setwritemode(int mode);`

Описание Позволяет выбрать, каким образом новые пиксели комбинируются с уже отображаемыми на экране пикселями. Влияет на вывод, производимый функциями drawpoly(), line(), linetl(), lineto() и rectangle().

Параметры

int mode - Либо COPY_PUT (новые пиксели записываются поверх существующих), либо XOR_PUT (новые пиксели комбинируются с существующими с помощью операции "ИСКЛЮЧАЮЩЕЕ ИЛИ").

- **TEXTHEIGHT** - Получить высоту строки текста.

Синтаксис `int far textheight(char far *textstring);`

Описание Возвращает высоту в пикселях строки текста, которая должна отображаться в графическом режиме, используя текущий шрифт, установленный функцией settextstyle().

Параметры

char far *textstring - Указатель на строку, которая будет отображаться в графическом режиме.

Синтаксис `void far settextstyle(int font, int direction, int charsize);`

Описание Устанавливает характеристики текста, используемые в последующих вызовах функций outtextxy() и outtext().

Параметры

int font - Одно из имен встроенных шрифтов BOLD_FONT, COMPLEX_FONT, DEFAULT_FONT, EUROPEAN_FONT, GOTHIC_FONT, SANS_SERIF_FONT, SCRIPT_FONT, SIMPLEX_FONT, SMALL_FONT, TRIPLEX_FONT, TRIPLEX_SCR_FONT или имя пользовательского шрифта.

int direction - Либо HORIZ_DIR - для горизонтального отображения текста, либо VERT_DIR - для вертикального отображения текста.

int direction - Относительный размер символов. Нулевое значение позволяет выбрать размер по умолчанию штрихового (векторного) шрифта или размер, указанный в предшествующем вызове функции setusercharsize().

- **SETUSERCHARSIZE** - Установить размер текста в графическом режиме.

Синтаксис `void far setusercharsize(int multx, int divx, int multy, int divy);`

Описание Изменяет горизонтальный и вертикальный размер текста в графическом режиме. Чтобы получить размер символов, меньший нормального, используйте значение коэффициента масштабирования меньше единицы. Например, если multx/divx равно 0.25, будут отражаться символы ширины, равной одной четверти их нормальной ширины. Если multy/divy равно 3.5, будут отображаться символы, высота которых приблизительно в 3.5 раз больше их нормальной высоты. Чтобы использовать новое значение коэффициента масштабирования, после вызова функции setusercharsize() вызывайте функцию settextstyle() с значением параметра charsize, равным нулю.

Параметры

int multx, int divx - Коэффициент горизонтального масштабирования, равный multx/divx.

int multy, int divy - Коэффициент вертикального масштабирования, равный multy/divy.

- **SETVIEWPORT** - Изменить графическую область вывода.

Синтаксис `void far setviewport(int left, int top, int right, int bottom, int clip);`

Описание Задает графическую область вывода и определяет, нужно ли производить отсечение по границам области вывода.

Параметры

int left, int top - Абсолютная координата верхнего левого угла прямоугольника области вывода.

int right, int bottom - Абсолютная координата нижнего правого угла прямоугольника области вывода.

int clip - 0 - не производить отсечение, ненулевое значение - производить отсечение.

- **SETWRITEMODE** - Установить режим вывода рисования линий.

Синтаксис `void far setwritemode(int mode);`

Описание Позволяет выбрать, каким образом новые пиксели комбинируются с уже отображаемыми на экране пикселями. Влияет на вывод, производимый функциями drawpoly(), line(), linerel(), lineto() и rectangle().

Параметры

int mode - Либо COPY_PUT (новые пиксели записываются поверх существующих), либо XOR_PUT (новые пиксели комбинируются с существующими с помощью операции "ИСКЛЮЧАЮЩЕЕ ИЛИ").

- **TEXTHEIGHT** - Получить высоту строки текста.

Синтаксис `int far textheight(char far *textstring);`

Описание Возвращает высоту в пикселях строки текста, которая должна отображаться в графическом режиме, используя текущий шрифт, установленный функцией setTextStyle().

Параметры

char far *textstring - Указатель на строку, которая будет отображаться в графическом режиме.

- **TEXTWIDTH** - Получить ширину строки текста.

Синтаксис `int far textwidth(char far *textstring);`

Описание Возвращает ширину в пикселях строки текста, которая должна отображаться в графическом режиме, используя текущий шрифт, установленный функцией setTextStyle().

Параметры

char far *textstring - Указатель на строку, которая будет отображаться в графическом режиме.

9.1.3.3 Функции динамического распределения памяти

(Заголовочный файл - *STDLIB.H* или *ALLOC.H*)

- **CALLOC** - Выделить и очистить память.

Синтаксис `void calloc(size_t nitems, size_t size);`

Описание Выделяет в куче блок памяти размером вплоть до 64К и устанавливает каждый байт этого блока равным нулю. В случае ошибки возвращает нуль, а иначе возвращает адрес первого выделенного байта.

Параметры

size t nitems - Количество выделяемых элементов.

size t size - Размер одного элемента в байтах.

- **MALLOC** - Выделить память.

Синтаксис `void malloc(size_t size);`

Описание Выделяет блок памяти из кучи. В случае успеха данная функция возвращает указатель на выделенную память. Если нельзя выделить запрошенное число байтов, функция возвращает нуль.

Параметры

size t size - Число выделяемых байтов памяти. Если значение size равно нулю, malloc() возвращает нуль.

9.1.3.4 Различные функции преобразования числовых данных в строку

- **CPRINTF** - Выполнить форматированный вывод на экран.

Синтаксис `void cprintf(const char *format[, argument,.....]);`

(Заголовочный файл *CONIO.H*)

Описание Отображает форматированный вывод с помощью программ непосредственного видеовывода. Функция sprintf() не распознает управляющий код новой строки “\n” и возврата каретки – перевода строки “\r”. Для позиционирования курсора до и после вызова sprintf() используйте функцию gotoxy().

Параметры

const char *format – Стока формата, содержащая отображаемый текст и управляющие коды вида “%s”. Чтобы начать новую строку, используйте управляющий код “\r”.

Argument... – Список значений-аргументов соответствующий типов данных, разделенных запятыми.

У остальных функций заголовочный файл - STDLIB.H

- **ECVT** - Преобразовать число с плавающей точкой в строку.

Синтаксис `char *ecvt(double value, int ndig, int *dec, int *sign);`

Описание Преобразует заданный аргумент value типа double в завершающуюся нулем строку. Возвращает адрес статистического буфера, который перезаписывается при каждом вызове этой функции.

Параметры

double value – Преобразуемое значение с плавающей точкой.

int ndig – Число требуемых символов цифр результата.

int *dec – Указатель на целое, в котором ecvt() запоминает относительную позицию десятичной точки. (Результирующая строка не содержит символ десятичной точки.) Отрицательное значение означает, что десятичная точка находится слева от первой цифры строки.

int *sign – Функция устанавливает это целое равным нулю, если значение аргумента value больше или равно нулю. *sign устанавливается равным ненулевому значению, если значение аргумента value меньше нуля.

- **FCVT** - Преобразовать число с плавающей точкой в строку.

Синтаксис `char *fcvt(double value, int ndig, int *dec, int *sign);`

Описание Аналогична функции ecvt(), за исключением того, что округление выполняется в соответствии со спецификациями FORTRAN-F.

Параметры

Такие же, как и у ecvt().

- **GCVT** - Преобразовать число с плавающей точкой в строку.

Синтаксис `char *gcvt(double value, int ndec, char *buf);`

Описание Преобразует значение типа double в завершающуюся нулем ASCII-строку и возвращает buf. Результат, запоминаемый в области памяти, адресуемой аргументом buf, имеет ndec десятичных разрядов в формате языка FORTRAN:

[+/-] [нули] [цифра...][‘.’][цифра...]

или, если этот формат не может точно представить значение value, результат выводится в формате E функции printf():

[+/-] цифра [‘.’] ‘E’ знак [цифра...]

Параметры

double value – Значение с плавающей точкой, преобразуемое в ASCII.

int ndec – Число требуемых десятичных разрядов. Может быть задано максимум 18 десятичных разрядов.

char *buf – Указатель на символьный буфер результата.

- **ITOА** - Преобразовать целое в строку.

Синтаксис `char *itoa(int value, char *string, int radix);`

Описание Преобрназует значение типа int в завершающуюся нулем строку. Возвращает указатель на результирующую строку.

Параметры

int value – Преобразуемое значение. Если аргумент value – отрицательный и аргумент radix равен 10, результат выводится с предшествующим знаком минуса.

char *string – Указатель на буфер типа char достаточного для размещения результата размера. В зависимости от значения radix функция itoa() может возвращать вплоть до 17 байтов.

int radix – Используемое при преобразовании основание системы счисления: от 2 до 36.

- **LTOA** - Преобразовать длинное целое в строку.

Синтаксис `char *ltoa(long value,char *string, int radix);`

Описание Преобразует значение типа long в завершающуюся нулем строку. Возвращает указатель на результирующую строку.

Параметры

int value – Преобразуемое значение. Если аргумент value – отрицательный и аргумент radix равен 10, результат выводится с предшествующим знаком минуса.

char *string – Указатель на буфер типа char достаточного для размещения результата размера. В зависимости от значения radix функция ltoa() может возвращать вплоть до 33 байтов.

int radix – Используемое при преобразовании основание системы счисления: от 2 до 36.

9.2 Пример

```
/* ===== ГРАФИЧЕСКИЕ СРЕДСТВА TURBO-C ===== */
/* Вывод и масштабирование изображения государственного флага */
/* Украины и текстов различными шрифтами */
#include <stdio.h>
#include <conio.h>
#include <graphics.h>

/* ===== ВНЕШНИЕ ПЕРЕМЕННЫЕ ===== */
int /* координаты точки левого верхнего угла области рисунка */
x, /* координата по x */
y; /* координата по y */
int m, /* масштаб */
gh=DETECT, /* параметры для инициализации графики */
gm;
int maxx, /* размер экрана в данном графическом режиме по x */
maxy; /* размер экрана в данном графическом режиме по y */
int ErrC; /* код ошибки, формируемый графическими примитивами */
main()
{ /* прототипы функций-сообщений об ошибках: */
void error_text(int); /* в текстовом режиме */
void error_gr(int); /* в графическом режиме */
int i,
j;
```

```
int rs=40, /* размер стороны квадрата, в который вписывается рисунок */
rv=20, /* размер флага по вертикали */
rh=30; /* размер флага по горизонтали */
int xc, /* координата x центра рисунка */
yc; /* координата y центра рисунка */
int xrl, /* координата x левого верхнего угла рисунка*/
yrl, /* координата y левого верхнего угла рисунка*/
xrp, /* координата x правого нижнего угла рисунка*/
yrp; /* координата y правого нижнего угла рисунка*/
int dd=10; /* величина задержки при анимации */
int color_ekr, /* цвет экрана */
color_bar; /* цвет области рисунка */
int size; /* размер области памяти для хранения изображения при
анимации */
int code_err; /* код ошибки для функций печати сообщений */
int ii; /* промежуточная переменная для организации анимации */
void *image; /* область памяти для хранения анимированного
изображения */

char tt[]="Ukraine"; /* строка для пробы текста */
```

```
clrscr();
printf("Введите точку с координатами x y, разделяя их пробелом :\n");
scanf("%d%d",&x,&y);
if(x<0 || y<0)
{code_err=0;
error_text(code_err);
}
printf("\nВведите масштаб в виде целого числа: 1, 2, 3 ...\n");
scanf("%d",&m);
if(m<=0 )
{code_err=1;
error_text(code_err);
}
/* инициализация графического режима */
initgraph(&gh,&gm,"");
ErrC=graphresult();
if(ErrC != grOk)
```

```

{printf("\nОШИБКА: %s", graphererrmsg(ErrC));
 getch();
 exit(-1);
}
/* проверка ограничений по результатам открытия графики */
maxx=getmaxx();
maxy=getmaxy();
color_ekr=GREEN,
color_bar=WHITE,
/* очистка экрана цветом фона */
setfillstyle(SOLID_FILL, color_ekr); /* параметры кисти */
bar(0, 0, maxx, maxy);
/* проверка ограничений на введенные значения области рисунка */
if(x>maxx || y>maxy /* проверка левого верхнего угла */
|| x+rs>maxx || y+rs>maxy /* проверка левого верхнего угла */
)
{code_err=0;
 error_gr(code_err);
}
if(x+rs*m>maxx || y+rs*m>maxy)
{code_err=1;
 error_gr(code_err);
}
setfillstyle(SOLID_FILL, color_bar); /* параметры кисти */
bar(x, y, x+rs*m,y+rs*m);
/* ===== масштабирование ===== */
xc=x+(rs*m)/2;
yc=y+(rs*m)/2;
rv*=m;
rh*=m;

xrl=xc-rh/2;
yrl=yc-rv/2;
xrp=xc+rh/2;
урp=yc+rv/2;
/* ===== рисование флага ===== */
setfillstyle(SOLID_FILL, LIGHTBLUE); /* параметры кисти */
bar(xrl, yrl, xrp, yc);
setfillstyle(SOLID_FILL, YELLOW); /* параметры кисти */
bar(xrl, yc, xrp, урp);

```

```

/* ===== анимация ===== */
/* определение размера картинки с изображением флага */
size = imagesize(xrl, yrl, xrp, урp);
ErrC=graphresult();
if(ErrC != grOk)
{printf("\nОШИБКА: %s", graphererrmsg(ErrC));
 getch();
 exit(-1);
}
/* чтение картинки из видеопамяти в оперативную */
getImage (xrl, yrl, xrp, урp, image);
ErrC=graphresult();
if(ErrC != grOk)
{printf("\nОШИБКА: %s", graphererrmsg(ErrC));
 getch();
 exit(-1);
}
/* перемещение картинки по периметру экрана */
for(ii=0; ii<=maxx-rh; ii++) /* верхняя горизонталь -> вправо*/
{ putimage (ii, 0, image, COPY_PUT);
 delay(dd);
}
for(ii=0; ii<=maxy-rv; ii++) /* правая вертикаль -> вниз */
{ putimage (maxx-rh, ii, image, COPY_PUT);
 delay(dd);
}
for(ii=maxx-rh; ii>=0; ii--) /* нижняя горизонталь -> влево */
{ putimage (ii, maxy-rv, image, COPY_PUT);
 delay(dd);
}
for(ii=maxy-rs*m+rv; ii>=0; ii--) /* левая вертикаль -> вверх */
{ putimage (0, ii, image, COPY_PUT);
 delay(dd);
}
getch();
/* вывод текста различными шрифтами */
setcolor(15);

```

```

/* по горизонтали*/
settextstyle(GOTHIC_FONT,HORIZ_DIR,6);
outtextxy(220,330,tt);
settextstyle(TRIPLEX_FONT,HORIZ_DIR,3);
outtextxy(340,120,tt);
settextstyle(DEFAULT_FONT,HORIZ_DIR,2);
outtextxy(200,200,tt);
settextstyle(DEFAULT_FONT,HORIZ_DIR,3);
outtextxy(10,100,"ТИМиИ");
/* по вертикали*/
settextstyle(DEFAULT_FONT,VERT_DIR,4);
outtextxy(320,0,tt);
getch();
closegraph();
ErrC=graphresult();
if(ErrC != grOk)
{printf("\nОШИБКА: %s", grapherrmsg(ErrC));
 getch();
 exit(-1);
}
getch();
}
/* === функция вывода сообщений об ошибке в текстовом режиме === */
void error_text(int n)
{char st_err[10][80]=
 {
 "Введенные координаты точки ошибочны",
 "Введенный масштаб для изображения ошибочен"
 };
 puts(st_err[n]);
 getch();
 exit(-1);
}
/* === функция вывода сообщений об ошибке в графическом режиме === */
void error_gr(int n)
{char st_err[10][80]=
 {
 "Введенные координаты выходят за пределы экрана",
 "Введенный масштаб для изображения слишком велик"
 };
 int color_text_err= WHITE;

```

```

char sx[20],
sy[20],
sm[20];
settextjustify(LEFT_TEXT, BOTTOM_TEXT);
settextstyle(DEFAULT_FONT, HORIZ_DIR, 1);
setcolor(color_text_err);
switch(n)
{
 case 0: /* перевод числового значения координат в строки */
 sprintf(sx,"%d",x);
 sprintf(sy,"%d",y);
 /* вывод сообщения об ошибке */
 outtextxy(100, 100, st_err[n]);
 color_text_err= RED;
 setcolor(color_text_err);
 outtextxy(480, 100, sx);
 outtextxy(520, 100, sy);
 break;
 case 1: /* перевод числового значения масштаба в строку */
 sprintf(sm,"%d",m);
 /* вывод сообщения об ошибке */
 outtextxy(100, 100, st_err[n]);
 color_text_err= RED;
 setcolor(color_text_err);
 outtextxy(490, 100, sm);
 break;
}
getch();
closegraph();
exit(-1);
}

```

9.3 Задание

Создать статическое цветное изображение заставки для своего варианта курсовой работы. Анимировать это изображение.

Для студентов специальности ПО обеспечить масштабирование изображения пользователем.

Заветка должна содержать следующий текст, выводимый различными шрифтами и размерами:

- название темы курсовой работы;
- фамилия и инициалы студента;
- название группы.

Расчеты координат графических объектов должны быть выполнены только в относительных координатах, например, относительно центра экрана; для этого использовать переменные, выражения, а не числовые константы.

Обязательной является проверка результата выполнения всех функций графики.

9.4 Контрольные вопросы по лабораторной работе

- 1) Настроить среду TURBO-C на работу с графической библиотекой.
- 2) Записать на языке Си вызовы функций графических примитивов для рисования заданного изображения.
- 3) Записать на языке Си программу анимации этого изображения.

9.5. Содержание отчета.

1. Условие задачи.
2. Виды экрана с расположенными на нем графическими изображениями и текстом (начальное и конечное состояния при анимации).
3. Текст программы на языке Си.
4. Выводы.
5. Программный документ "Описание программы".

10 ЛАБОРАТОРНАЯ РАБОТА № 9. ДИНАМИЧЕСКИЕ СТРУКТУРЫ ДАННЫХ В ЯЗЫКЕ СИ

Цель работы: освоить приемы алгоритмизации при организации работы с динамическими структурами данных (на примере решения задач обработки односвязанных списков).

Контрольные вопросы:

- 1) Как выполняется в языке Си динамическое распределение памяти?
- 2) Как выполняется в языке Си освобождение памяти, выделенной динамически?
- 3) Какие динамические структуры данных могут быть промоделированы в языке Си односвязанными списками?
- 4) Что такое "стек"? Назовите операции, определенные для этой структуры данных.
- 5) Что такое "очередь"? Назовите операции, определенные для этой структуры данных.

10.1 Приемы алгоритмизации и программирования при решении задач обработки динамических структур данных

10.1.1 Выбор способа представления и обработки списка

Условие решаемой задачи определяет выбор способа представления списка (дисциплину обслуживания).

Организация списка как стека подходит для решения задач о разбиении исходного списка на подсписки, для определения признака состояния списка (характеристики длины и т.д.). Следует помнить, что в этом случае элементы списка будут доступны в порядке, обратном их поступлению.

Если же изменяется порядок следования элементов списка или выполняется модификация его состояния (удаление/добавление элементов с произвольным номером в списке и т.п.), то предпочтительнее организация списка как очереди.

10.1.2 Способы передачи списков в функции

Так как список доступен в алгоритмических языках программирования через указатель на его "голову", следует правильно организовать передачу списка в функции.

10.1.2.1 Формируемые списки

1) Если формируемый список возвращается из функции как ее аргумент, то соответствующий аргумент представляет собой указатель на указатель “головы” списка:

```
/* структурный тип данных для элементов списка */
#define STACK struct stack
STACK { int info; STACK *next; };
void push(STACK **s, int item); /* прототип функции “добавить в стек”*/
В теле этой функции доступ к содержимому элементов такого списка осуществляется только через указатели, а к содержимому членов элемента - через операцию “->”.
void push(STACK **s, int item)           /* определение функции */
{
    STACK *new_item;
    new_item=(STACK*)malloc(sizeof(STACK));
    ...
    new_item->next=*s;
    *s=new_item;
}
```

При вызове такой функции в качестве фактического аргумента передается адрес списка:

```
... STACK * list3; /* указатель на “голову” списка */
push (&list3, error);
```

Эта функция применима для формирования нескольких списков.

2) Если формируемый список возвращается из функции как ее результат, то он оформляется как указатель “головы” списка:

```
STACK * create_list(int n_list); /* прототип функции создания списка*/
В теле этой функции список формируется в промежуточной переменной, локальной внутри функции. При завершении формирования списка значение этой промежуточной переменной возвращается в качестве результата функции:
```

```
STACK * create_list(int n_list) /* определение функции создания списка */
{
    STACK * list=NULL;          /* объявление локальной переменной*/
    ...
    push(&list,info);           /* работа с этой локальной переменной*/
    return(list);                /* возврат значения функции */
}
```

При вызове такой функции ее результат и есть указатель на голову сформированного списка.

```
STACK * list1;                  /* указатель на “голову” списка */
list1 = create_list(1);          /* создание списка list1 */
```

Эта функция применима для формирования нескольких списков.

3) Если список формируется в функции как внешняя переменная, то данная функция не возвращает результата (имеет тип void), а в ее теле при формировании списка используется имя этой внешней переменной:

```
STACK * list1; /* указатель на “голову” списка - внешняя переменная*/
void create_list ()           /* определение функции создания списка*/
{
    ... push(&list1, info);...
} /* работа с этой внешней переменной*/
При вызове такой функции она формирует требуемый список:
create_list();                 /* создание списка list1 */
```

Эта функция применима для формирования только одного списка.

10.1.2.2 Обрабатываемые списки

Следует помнить, что непосредственное использование значения указателя на “голову” списка в качестве текущего указателя приводит к тому, что его значение при завершении обработки списка отличается от начального. Все дальнейшие действия со списком в этом случае не являются корректными и во многих случаях приводят к программному прерыванию. Для исключения подобной ситуации необходимо в качестве текущего указателя списка использовать копию значения указателя на “голову” списка.

1) Если указатель на “голову” списка передается аргументом в функцию и копия создается в самой функции:

```
#define STACK struct stack
STACK * list1, * list2; /* указатели на “голову” списков */
/* функция вывода на экран содержимого списка */
void display (STACK *lst) /* аргумент - указатель на “голову” списка */
{
    STACK * current = lst; /* копия указателя на “голову” списка*/
    while (current)        /* current - текущий указатель списка*/
    {
        printf("%d-->", current->info);
        current=current->next;
    }
}
```

то перед вызовом такой функции нет необходимости копировать значение указателя на список, так как созданная копия локальна внутри функции:

```
... display (list1); display (list2); /* вызовы функции */
```

Эта функция применима для вывода нескольких списков.

2) Если указатель на “голову” списка передается аргументом в функцию, но копия внутри функции не создается:

```
#define STACK struct stack  
STACK * list1; /* указатель на “голову” списка */  
/* функция вывода на экран содержимого списка */  
void display1 (STACK ** lst) /* указатель на указатель “головы” списка */  
{ while (* lst) /* lst - текущий указатель списка */  
    { printf("%d->", (* lst)->info);  
     * lst = (* lst)->next;  
    }  
}
```

то копию необходимо создавать перед вызовом такой функции в вызывающей ее и использовать эту копию как аргумент:

```
STACK * list1; /* указатель на “голову” списка */  
STACK * list11 /* переменная для копии указателя на список */  
... list11=list1; /* создание копии */  
display1(&list11); /* использование копии */
```

Далее *list1* снова указывает на “голову” списка.

Эта функция применима для вывода нескольких списков.

3) Если указатель на “голову” списка является внешней переменной по отношению к функции:

```
STACK * list2; /* указатель на “голову” списка */  
void display2 ()  
{ while (list2) /* list2 - текущий указатель в списке */  
    { printf("%d->", list2->info);  
     list2 = list2->next;  
    }  
}
```

то эта функция применима для вывода только данного списка (*list2*):

```
display2();
```

10.2 Пример

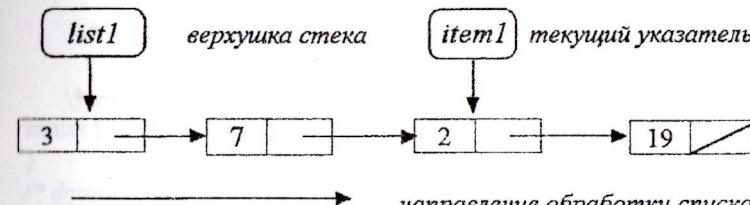
Определить функцию РАЗНОСТЬ для нахождения разности двух множеств.

10.2.1 Выбор способа представления данных

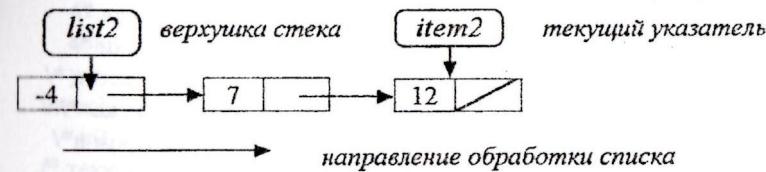
Множества естественным образом моделируются списком своих элементов. Так как при нахождении разности множеств не требуется перестроения исходных списков, а только их просмотр,

анализ и копирование, то для обработки списков-множеств достаточно операций для стека: “в”, “из” и “просмотреть” (дисциплина обслуживания LIFO).

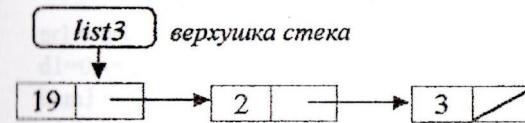
Пусть первое исходное множество $A = \{19, 2, 7, 3\}$. В списке *list1* оно представляется так:



Второе множество $B = \{12, 7, -4\}$ представляется в списке *list2*:



$C = A - B = \{19, 2, 3\}$ моделируется списком *list3*:



Для реализации операции РАЗНОСТЬ необходимо организовать поэлементный просмотр списка *list1*. Для каждого элемента этого списка необходим поэлементный просмотр списка *list2*. Элемент из *list1* переносится в список *list3* только в том случае, когда он не встретится в *list2*.

Для перемещения по спискам *list1* и *list2* требуются свои текущие указатели, причем указатель в *list2* должен быть установлен на начало этого списка при переходе к каждому следующему элементу в *list1*.

Признак пустого результирующего множества ($C = \emptyset$) может быть промоделирован нулевым состоянием счетчика записанных в *list3* элементов.

10.2.2 Текст программы на языке Си

```

#include <alloc.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <ctype.h>
#define N 20
/* ===== структурный тип данных для элементов списка ===== */
#define STACK struct stack
STACK           /* список с дисциплиной обслуживания LIFO = стек */
{ int info;
  STACK *next;
};
/* ===== ПРОТОТИПЫ ФУНКЦИЙ: операций со стеком: ===== */
void push(STACK **s, int item);           /* втолкнуть в стек*/
int pop  (STACK **s, int *err);           /* вытолкнуть из стека */
int peek (STACK *s, int *err);           /* выбрать с верхушки */
void difference(STACK *stack1, STACK *stack2); /* разности множеств */
void display(STACK *lst);                /* вывода на экран содержимого списка */
STACK * create_list (int n);             /* создания списка с порядковым номером n*/
/* ===== ИСХОДНЫЕ ДАННЫЕ ===== */
STACK *list1,          /* список - модель первого множества */
      *list2;          /* список - модель второго множества */
/* ===== РЕЗУЛЬТАТЫ ===== */
STACK *list3;          /* список - модель множества-результата */
/* функция вывода на экран содержимого списка */
void display(STACK *lst)
{ STACK *current = lst;
  while(current)
  { printf("%d-->", current->info);
    current=current->next;
  }
}
/* функция создания списка с порядковым номером n_list*/
STACK * create_list(int n_list)
{STACK * list = NULL;
 int done=1,        /* признак окончания ввода элементов списка */
   info,            /* вводимый элемент списка */
   c;               /* символ вводимого пользователем ответа в диалоге: (Y/N)*/
  printf("\n      Заполнение %d-го списка:\n ", n_list);
  while(done)
  { printf("Добавить новый элемент(Y/N)");

```

```

c=getch(); c=toupper(c);
switch (c)
{ case 'Y': printf("\n Элемент= ");
  scanf("%d", &info);
  push(&list, info);
  break;
  case 'N': done=0;           /* конец формирования списка */
  break;
}
return (list);
}
/* функция нахождения разности двух множеств (в виде списков) */
void difference (STACK *stack1, STACK *stack2)
{int d1,           /* считываемый элемент 1-го списка */
 d2,           /* считываемый элемент 2-го списка */
 count=0,         /* количество элементов в списке-результате*/
 yes=0,          /* количество совпавших элементов в обоих списках */
 int er1,          /* код ошибки, возвращаемый функцией peek в первом списке */
 er2,          /* код ошибки, возвращаемый функцией peek во втором списке */
 /* текущие указатели для перемещения по исходным спискам: */
 STACK *item1,          /*по первому */
 *item2;          /*по второму */
 er1 = er2 =0;
 d1=0;
 item1 = stack1;           /* внешний цикл - по 1-му списку */
 while( er1 == 0)
 {d1= peek (item1,&er1); /* считали текущий элемент 1-го списка*/
 if((d1==0)&&( er1 == 1)) /* 1-й список пуст */
  break;
 /* 1-й не пуст: начинаем просмотр 2-го для текущего элемента из 1-го */
 yes = 0;           /* сброс счетчика совпавших элементов */
 item2 = stack2;           /* внутренний цикл - по 2-му списку */
 er2 = 0;           /* сброс кода результата при чтении из 2-го */
 while(er2 == 0)           /* пока 2-й не пуст */
 {d2 = peek (item2,&er2); /* считали текущий элемент 2-го списка */
 if((d2==0)&&( er2 == 1)) /* 2-й список пуст */
  break;
 if(d1==d2)           /* 2-й список не пуст: есть совпавшие */
  yes++;           /* их счет */
 item2=item2->next;           /* перемещение по 2-му */
}
}

```

```

if (yes==0)      /* для текущего из 1-го во 2-м нет совпадающего */
{ push (&list3, d1);           /* в разность заносим элемент 1-го */
  count++;                /* счет элементов, записанных в результат */
}
item1=item1->next;          /* перемещение по 1-му */
}
if ( !count )               /* список-результат пуст */
{ puts("\nРазностью этих множеств будет пустое множество.");
  getch();
  exit(-1);
}
/* иначе найдена разность в виде списка list3 */
}

void main ()
{
clrscr();
list1 = create_list(1);      /* создать 1-й список*/
printf ("\nИсходный 1-й список имеет вид:\n\"a\"");
display (list1);
list2 = create_list(2);      /* создать 2-й список*/
printf ("\nИсходный 2-й список имеет вид:\n\"a\"");
display (list2);
difference (list1,list2);   /* найти разность */
printf ("\n\nВыходной список разности множеств имеет вид:\n");
display(list3);
getch();
}

```

Тексты функций “втолкнуть в стек” (*push*), “вытолкнуть из стека” (*pop*) и “выбрать с верхушки” (*peek*) рассмотрены в лекционном материале.

10.3 Задания

Определить функцию для работы с односвязанными списками. Для студентов специальности ПО заданные преобразования списков выполнять на месте (в исходном списке, если не оговорено иначе). Если в задании предусмотрено формирование более одного списка-результата, один из них получать в исходном списке.

Для студентов специальности МИ заданные преобразования списков выполнять путем формирования новых списков.

Варианты заданий

1. **УДАЛЕНИЕ_ЧЕТНЫХ**, которая формирует новый список по исходному списку путем удаления из него элементов, стоящих на четных местах (элементы списка нумеруются с единицы). Например, L1=(a, b, c, d, e), формируется L2=(a, c, e).
2. **ВСТАВКА**, которая формирует новый список по исходному списку следующим образом: перед каждым элементом со значением X вставляется элемент со значением Y, а после элемента X - элемент со значением Z. Например, L1=(‘a’, ‘#’, ‘b’, ‘#’), X=‘#’, Y=[‘, Z=‘]’, формируется L2=(‘a’, [‘, ‘#’, ‘], ‘b’, [‘, ‘#’, ‘]’).
3. **УДАЛЕНИЕ_ЭЛЕМЕНТОВ**, которая формирует новый список по исходному списку L1 путем удаления из него элементов, которые стоят непосредственно перед элементом со значением X. Например, L1 = (7, 9, 8, 6, 5, 9, 1), X = 9, формируется L2=(9, 8, 6, 9, 1).
4. **ДЕЛИТЬ_ПОПОЛАМ**, которая распределяет элементы списка L между двумя новыми списками L1 и L2 таким образом, чтобы получаемые списки были примерно одинаковой длины. В случае нечетной длины исходного списка L остающийся “непарный” элемент поместить в список L1. Например, L = (7, 9, 8, 6, 5, 9, 1), формируются L1=(7, 8, 5, 1) и L2=(9, 6, 9).
5. **УДАЛЕНИЕ_НЕЧЕТНЫХ_ЧИСЕЛ**, которая формирует список из исходного списка путем удаления из него нечетных чисел. Например, L1=(2, 3, 4, 5, 6, 8, 9), формируется L2=(2, 4, 6, 8).
6. **СДВИГ_ВЛЕВО**, которая получает новый список из исходного списка путем циклического сдвига на один элемент влево. Например, L1=(a, b, c), формируется L2=(b, c, a).
7. **УДАЛЕНИЕ_ЭЛЕМЕНТА**, которая формирует новый список путем удаления из исходного списка всех вхождений элемента X. Например, L1=(2, 4, 5, 6, 4, 7, 2, 4), X=4, формируется L2=(2, 5, 6, 7, 2).
8. **БЕЗ_ДУБЛИРОВАНИЯ**, в которой новый список получается как результат удаления всех повторных вхождений элементов в исходном списке. Например, L1=(b, a, c, a), формируется L2=(b, a, c).

9. ***СДВИГ_ВПРАВО***, которая получает новый список из исходного списка путем циклического сдвига на один элемент вправо. Например, $L1=(a, b, c)$, формируется список $L2=(c, a, b)$.
10. ***ЗАМЕНА_ЧЕТНЫХ***, которая формирует новый список путем замены всех четных элементов исходного списка их модулями. Например, $L1=(90, -1, -72, 0)$, формируется $L2=(90, 1, 72, 0)$.
11. ***ДУБЛЬ***, которая каждый элемент исходного списка удваивает. Например, $L=(3, 2)$, формируется $LL=(3, 3, 2, 2)$.
12. ***УДАЛЕНИЕ_ОТРИЦАТЕЛЬНЫХ_ЧИСЕЛ***, которая формирует новый список из исходного списка путем удаления из него отрицательных чисел. Например, $L1=(-2, 3, -4, 5, 6, 8, -9)$, формируется список $L2=(3, 5, 6, 8)$.
13. ***ПОСЛЕДНИЙ***, утверждающую, что X является последним элементом списка L .
14. ***ДОБАВИТЬ_МОДУЛЬ***, которая формирует новый список из исходного списка путем добавления после каждого отрицательного элемента его абсолютного значения. Например, $L1=(2, -6, 4, -2)$, формируется $L2=(2, -6, 6, 4, -2, 2)$.
15. ***ИНДЕКС***, которая по заданному индексу i определяет значение элемента исходного списка L .
16. ***СУФФИКС***, утверждающую, что S является конечным подсписком списка L . Например, для списков $S=(e, f)$, $L=(c, d, e, f)$ функция возвращает значение “истина”.
17. ***СОСЕДИ***, которая определяет, являются ли соседними элементы X и Y списка L .
18. ***ПОДСТАНОВКА***, в которой новый список получается из исходного списка путем подстановки Y вместо всех вхождений X . Например, $L1=(b, a, c)$, $X=a$, $Y=x$, формируется $L2=(b, x, c)$.
19. ***ЧАСТОТА_X***, которая определяет, с какой частотой встречается элемент X в исходном списке L . Например, $L=(1, 6, 1, 9, 1, 25)$, $X=1$, функция возвращает значение 3.
20. ***УДАЛЕНИЕ***, в которой новый список получается из исходного списка удалением первого вхождения элемента X .
21. ***СУММА***, которая возвращает значение суммы элементов списка L . Например, для списка $L=(1, 2, 3, 4, 5)$ функция возвратит значение 15.

22. ***ПЕРЕСЕЧЕНИЕ***, которая формирует список Z как пересечение (согласно теории множеств) исходных списков X и Y .
23. ***ЧЕТНАЯ_ДЛИНА*** для выяснения четности длины исходного списка L .
24. ***ОБРАТИТЬ***, которая формирует новый список путем изменения порядка следования элементов на противоположный относительно исходного списка. Например, $L1=(b, a, c, a)$, формируется $L2=(a, c, a, b)$.
25. ***УНИКАЛЬНЫЙ***, которая формирует новый список из исходного списка, исключая повторяющиеся элементы. Например, $L1=(b, a, c, a)$, формируется $L2=(b, c)$.
26. ***КОНКАТЕНИРОВАТЬ*** для присоединения к списку $L1$ списка $L2$. Список $L3$ является списком-результатом: $L3=L1+L2$.
27. ***ПРЕФИКС***, утверждающую, что список P является начальным подсписком списка L . Например, для списков $L=(c, d, e, f)$ и $P=(c, d)$ функция возвратит значение “истина”.
28. ***СОРТИРОВКА_ПО_УБЫВАНИЮ***, которая упорядочивает элементы исходного списка по убыванию.
29. ***ДЕЛЕНИЕ_СПИСКА***, которая делит исходный список L арифметических данных на два: $L1$ и $L2$. Если элемент исходного списка L не больше значения $Comp$, он помещается в список $L1$, в противном случае - в список $L2$.
30. ***ДОБАВИТЬ_ДУБЛЬ_K***, которая формирует новый список из исходного списка путем дублирования k -го элемента списка. Например, $L1=(3, 5, 6, 7)$, $k=3$, формируется $L2=(3, 5, 6, 6, 7)$.
31. ***ИСКЛЮЧИТЬ***, которая формирует новый список из исходного списка путем удаления из него k -го элемента. Например, $L1=(a, b, c, d, e)$, $k=2$, формируется $L2=(a, c, d, e)$.
32. ***ПОЗИЦИЯ***, которая определяет номер (позицию) элемента X в исходном списке.
33. ***ДУБЛИРОВАТЬ_X***, которая формирует новый список из исходного списка путем дублирования заданного элемента X . Например, $L1=(c, a, b, c)$, $X=c$, формируется список $L2=(c, c, a, b, c, c)$.
34. ***УДАЛЕНИЕ_НЕЧЕТНЫХ***, которая формирует новый список по исходному списку путем удаления из него элементов,

стоящих на нечетных местах (элементы списка нумеруются с единицы). Например, $L1=(a, b, c, d, e)$, формируется $L2=(b, d)$.

35. **ПРОИЗВЕДЕНИЕ**, которая возвращает значение произведения элементов списка L . Например, для списка $L=(1, 2, 3, 4, 5)$ функция возвратит значение 120.

36. **УДАЛЕНИЕ_ПОЛОЖИТЕЛЬНЫХ_ЧИСЕЛ**, которая формирует новый список из исходного списка путем удаления из него положительных чисел. Например, $L1=(-2, 0, 4, -5, 6, -8, 9)$, формируется список $L2=(-2, 0, -5, -8)$.

37. **УДАЛЕНИЕ_ЭЛЕМЕНТОВ**, которая формирует новый список по исходному списку путем удаления из него элементов, которые стоят непосредственно после элемента со значением X . Например, $L1=(7, 9, 8, 5, 9, 1)$, $X=9$, формируется $L2=(7, 9, 5, 9)$.

38. **МОДУЛЬ**, которая формирует новый список, состоящий из абсолютных значений элементов исходного списка. Например, $L1=(-2, 0, 3, -4)$, формируется список $L2=(2, 0, 3, 4)$.

39. **НЕЧЕТНАЯ_ДЛИНА** для выяснения нечетности длины исходного списка L .

40. **ДЕЛЕНИЕ_ПО_НОМЕРУ**, которая делит исходный список на два по некоторому ключу k (номеру элемента в списке), причем второй список начинается с элемента с номером k . Например, $L=(2, 0, 23, 4, 7)$, $k=3$, формируются $L2=(2, 0)$ и $L2=(23, 4, 7)$.

41. **КРАТНЫЕ_ТРЕМ**, которая формирует новый список, состоящий из элементов исходного списка, кратных трем. Например, $L1=(-3, 10, 23, 0, 7, 54)$, формируется $L2=(-3, 0, 54)$.

42. **ВСТАВИТЬ**, которая формирует новый список путем вставки элемента X в список после k -го элемента. Например, $L1=(-3, 10, 23, 0, 7, 54)$, $X=77$ и $k=5$; формируется список $L2=(-3, 10, 23, 0, 7, 77, 54)$.

43. **ДЛИНА_C**, которая определяет длину списка, начиная с некоторого элемента X . Например, $L1=(26, 10, 2, 17, 104)$, $X=2$; функция возвращает значение 3.

44. **ЗАМЕНА_НЕЧЕТНЫХ**, которая формирует новый список путем замены всех нечетных элементов исходного списка нулями. Например, $L1=(300, 101, 73, 0)$, формируется $L2=(300, 0, 0, 0)$.

45. **ПОМЕНЯТЬ_МЕСТАМИ**, которая формирует новый список из исходного, поменяв местами k -й и i -й элементы местами. Например, $L1=(3, 1, 7, 0)$, $k=4$, $i=2$, формируется $L2=(3, 0, 7, 1)$.

46. **ПРИНАДЛЕЖИТ**, которая определяет принадлежность элемента X исходному списку L .

47. **ВСТАВКА_5**, которая формирует новый список по исходному списку следующим образом: перед каждым элементом, кратным 5, вставить элемент со значением X . Например, $L1=(10, 6, 9, 25)$, $X=22$, формируется $L2=(22, 10, 6, 9, 22, 25)$.

48. **ПОМЕНЯТЬ**, которая формирует новый список из исходного, поменяв местами первый и последний элементы местами.

49. **ПЕРЕМЕСТИТЬ_МИНИМУМ**, которая формирует новый список по исходному списку следующим образом: находит минимальный элемент исходного списка и перемещает его в начало списка, сделав первым элементом.

50. **ДЛИНА_C_НОМЕРА**, которая определяет длину списка, начиная с элемента с номером k . Например, $L1=(26, 10, 2, 17, 104)$, $k=2$; длина равна 4.

51. **ИСКЛЮЧИТЬ_K_N**, которая формирует новый список из исходного списка путем удаления из него N элементов, начиная с k -го. Например, $L1=(a, b, c, d, e, f)$, $k=2$, $N=3$, формируется $L2=(a, e, f)$.

52. **УДАЛЕНИЕ_ЧЕТНЫХ_ЧИСЕЛ**, которая формирует новый список из исходного списка путем удаления из него четных чисел. Например, $L1=(2, 3, 4, 6, 5, 9)$, формируется $L2=(3, 5, 9)$.

53. **ЧАСТОТА_B_ДИАПАЗОНЕ**, которая определяет, с какой частотой встречается элемент X в исходном списке L в диапазоне элементов, начиная с номера i и заканчивая номером j . Например, $L=(1, 6, 1, 9, 1, 25)$, $X=1$, $i=1$, $j=4$, функция возвращает значение 2.

54. **ПОМЕНЯТЬ_МЕСТАМИ**, которая формирует новый список из исходного, поменяв местами минимальный и максимальный элементы местами. Например, $L1=(3, 1, 7, 0)$, формируется $L2=(3, 1, 0, 7)$.

55. **СОРТИРОВКА_ПО_ВОЗРАСТАНИЮ**, которая упорядочивает элементы исходного списка по возрастанию.

10.4 Контрольные вопросы по лабораторной работе

- 1) Какая системная библиотека языка Си поддерживает работу с динамически распределяемой памятью?
- 2) Организовать на языке Си вывод элементов списка с помощью рекурсивной функции.
- 3) Сравнить организацию работы со списком в виде очереди и в виде стека при определении функций:
 - БЕЗ_ДУБЛИРОВАНИЯ (вариант 8);
 - ЧАСТОТА_X (вариант 19);
 - ОБРАТИТЬ (вариант 24).

10.5 Содержание отчета

1. Условие задачи.
2. Постановка задачи.
3. Обоснование выбора способа представления и обработки динамических структур данных задачи.
4. Метод решения задачи.
5. Алгоритмы (вспомогательные и основной) и их иерархия.
6. Текст программы на языке Си.
7. Таблица трассировки программы.
8. Графическая интерпретация тестовых примеров (с изображением формируемых списков, указателей и т.д.).
9. Выводы.
10. Программный документ "Описание программы".

11 ЛАБОРАТОРНАЯ РАБОТА № 10. РАБОТА С ФАЙЛАМИ ПРЯМОГО ДОСТУПА НА ЯЗЫКЕ СИ

Цель работы: освоить работу с файлами прямого доступа в языке Си и основные приемы программирования задач, использующих такие файлы.

Контрольные вопросы:

- 1) Различия между файлами прямого и последовательного доступа.
- 2) Что такое позиционирование файла?
- 3) К каким файлам может быть применена операция позиционирования?
- 4) Возможно ли чтение файла (запись в файл) прямого доступа в обратном порядке?
- 5) Способы обработки файлов прямого доступа.

11.1 Приемы алгоритмизации и программирования при работе с файлами прямого доступа

Открытие и закрытие файла с прямым доступом в языке Си выполняется также, как и для последовательных файлов.

11.1.1 Особенности обработки файлов прямого доступа

Как и для файла с последовательным доступом в отличие от структуры данных "таблица" для решения задачи о нахождении в файле прямого доступа "записи", соответствующей минимальному (максимальному) значению некоторого из членов записи, либо при сортировке такого файла, необходимо воспользоваться промежуточной переменной для хранения значения "записи", а не ее порядкового номера в файле.

Совершенно недопустимо переписывать содержимое файла в оперативную память для последующей его обработки, так как размеры файла во внешней памяти могут быть гораздо большие, чем объем доступной оперативной памяти.

Для реализации сортировки файла методом "пузырька" необходимо иметь доступ к каждой "записи" файла или ее члену (по аналогии с сортировкой в массиве иметь доступ к элементам массива по их номеру). Перемещение на любую позицию в

файле выполняется с помощью функция *fseek*. Предварительно необходимо вычислить размер блока файла в байтах, который соответствует размеру “записи” файла или ее члена, например, с помощью оператора *sizeof*. Этот размер является аргументом *size* функций *fwrite/fread*. При поэлементной обработке “записей” файла аргумент *number* должен быть равен 1. Если же необходимо “добраться” до записи с конкретным номером *i*, то либо ее позицию необходимо вычислить, либо “пропустить” (*i-1*) предшествующую ей запись. Пропуск записей в этом случае может быть выполнен с помощью цикла чтения пропускаемых записей в оперативную память, однако этот способ крайне неэффективен.

Внимание: При сравнении членов “записей” файла, которые являются строковыми переменными, допустимо использование только функции *strcmp*.

11.1.2 Функции, необходимые для работы с файлами прямого доступа

Прототипы функций *fseek, fwrite, fread* хранятся в файле *stdio.h*:

- **FSEEK** - Позиционирование в файле

Синтаксис *int fseek (File *bp, long offset, int whence)*

Описание Устанавливает указатель в файле, заданном файловой переменной *bp*, на указанную позицию *offset* при заданном способе смещения *whence*.

Параметры

*File *bp* - Файловая переменная.

long offset - Смещение.

int whence - Способ смещения:

SEEK_SET - от начала файла;

SEEK_CUR - от текущей позиции;

SEEK_END - от конца файла.

- **FWRITE** - Запись в файл

Синтаксис *size_t fwrite(const void *ptr, size_t size, size_t n, File *bp)*

Описание Функция *Fwrite()* записывает в файл, который задан файловой переменной, начиная с текущей позиции указателя, в переменную адресуемую *ptr*, *number* блоков по *size* байтов в каждом.

Параметры

*const void *ptr* - Область памяти.

size_t size - Размер блока.

size_t number - Количество блоков.

*File *bp* - Файловая переменная.

- **FREAD** - Чтение из файла

Синтаксис *size_t fread(void *ptr, size_t size, size_t n, File *bp)*

Описание Функция *Fread()* читает из файла, который задан файловой переменной, начиная с текущей позиции указателя, в переменную адресуемую *ptr*, *number* блоков по *size* байтов в каждом.

Параметры

*const void *ptr* - Область памяти.

size_t size - Размер блока.

size_t number - Количество блоков.

*File *bp* - Файловая переменная.

Первым параметром функции *fwrite()* и *fread()* должен быть указатель на область памяти. Поэтому, если необходимо записать или прочитать переменную целого типа, то следует использовать вспомогательную переменную строкового типа размера 2 байта + 1 байт (конец строки) и перевести ее из строкового типа в целый тип с помощью функции *atoi* (прототип этой функции находится в файле *stdlib.h*).

Оператор *sizeof* (*имя_переменной*) применяем для определения размера (количества байт) переменной. Если использовать явную константу, то при изменении типа переменной (например: *int--> long*), придется изменить и всю программу.

11.2 Пример

Сформировать файл, который содержит информацию о людях (фамилия, возраст, адрес). Найти самого старшего

человека. Отсортировать файл в алфавитном порядке по полю "фамилия".

11.2.1. Постановка задачи

11.2.1.1 Исходные данные

```
N : целое ; {количество записей в файле}
A : таблица[1..10](
    fio:массив[1..30]симв;
    age:целый;
    adr:массив[1..40]симв);
Fv : файл;
```

11.2.1.2 Ограничения

N>0

forall i in [1..N] A[i].age > 0;

11.2.1.3 Результаты

```
max: integer{максимальный возраст человека};
str : массив[1..30] симв {ФИО};
str1: массив[1..40] симв {адрес} ;
Fv : файл;
```

11.2.2. Программа на языке Си

```
#include <stdio.h>
#include <string.h>
#include <conio.h>
#include <stdlib.h>
/*
 *      ПРОТОТИПЫ ФУНКЦИЙ
 */
void File_input();
void File_output();
void Older();
void Sort();
/*
 * ИНФОРМАЦИЮ ОБ ОДНОМ ЧЕЛОВЕКЕ ПРЕДСТАВИМ В ВИДЕ *
 *      СТРУКТУРЫ
 */
struct RECORD
{
    char fio[30];
    int age;
    char adr[40];
} a;
FILE *fv; /* файловая переменная */
int N=0; /* количество записей в файле */
```

```
/*
 */
main()
{
    clrscr();
    File_input(); /* создание файла прямого доступа */
    Older(); /* поиск самого старшего человека */
    Sort(); /* сортировка файла методом "пузырька" */
    File_output(); /* функция вывода содержимого файла */
}

/* ===== ФУНКЦИЯ СОЗДАНИЯ ФАЙЛА ПРЯМОГО ДОСТУПА ===== */
void File_input()
{
    char strok[3];
    /* открываем файл на запись */
    if ((fv=fopen("res.dat", "w"))==NULL)
        {printf("Ошибка открытия файла");
         exit(-1);
        }
    strcpy(a.fio,"a ");
    /* ЦИКЛ ВВОДА ДАННЫХ И ЗАПИСИ ИХ В ФАЙЛ*/
    while (strcmp(a.fio,"")!=0)
    {
        printf("ФИО:");
        gets(a.fio);
        if (strcmp(a.fio,"")!=0)
        {
            N++;
            fwrite(a.fio,sizeof(a.fio),1,fv);
            printf("Возраст:");
            gets(strok);
            fwrite(strok,sizeof(a.age)+1,1,fv);
            printf("Адрес:");
            gets(a.adr);
            fwrite(a.adr,sizeof(a.adr),1,fv);
        }
    }
    printf("Все данные записаны в файл res.dat\n");
    fclose(fv);
}

/* ===== ОТКРЫВАЕМ ФАЙЛ НА ЧТЕНИЕ И ЗАПИСЬ ===== */
if ((fv=fopen("res.dat", "r+w"))==NULL)
    {printf("Ошибка открытия файла");
     exit(-1);
    }
}
```

```

/* ===== ФУНКЦИЯ ВЫВОДА СОДЕРЖИМОГО ФАЙЛА ===== */
void File_output()
{
    int uk,i;
    char str[3];
    fseek(fv,0,SEEK_END);
    /* ОПРЕДЕЛЯЕМ ДЛИНУ ФАЙЛА */
    uk=f.tell(fv);
    i=0;
    while(i<uk)
    {
        fseek(fv,i,SEEK_SET);
        fread(a.fio,sizeof(a.fio),1,fv);
        fread(str,sizeof(a.age),1,fv);
        fseek(fv,f.tell(fv)+1,SEEK_SET);
        fread(a.adr,sizeof(a.adr),1,fv);
        printf("\n%s %s %s",a.fio,str,a.adr);
        i+=sizeof(a)+1;
    }
    fclose(fv);
}
/* ===== ПОИСК САМОГО СТАРИШЕГО ЧЕЛОВЕКА ===== */
void Older()
{
    int max=0,i=0,uk;
    char str[30],str1[40],s[10];
    fseek(fv,0,SEEK_END);
    uk=f.tell(fv);
    while(i<uk)
    {
        fseek(fv,i,SEEK_SET);
        fread(a.fio,sizeof(a.fio),1,fv);
        fread(s,sizeof(a.age),1,fv);
        a.age=atoi(s);
        fseek(fv,f.tell(fv)+1,SEEK_SET);
        fread(a.adr,sizeof(a.adr),1,fv);
        if (a.age>max)
        {
            max=a.age;
            strcpy(str,a.fio);
            strcpy(str1,a.adr);
        }
        i+=sizeof(a)+1;
    }
    printf("Самый старший: %s (возраст=%d) проживает по адресу %s", str,
    max, str1);
}

```

```

/* ===== СОРТИРОВКА ФАЙЛА МЕТОДОМ "ПУЗЫРЬКА" ===== */
void Sort()
{
    int i=0, /*НУМЕР ЗАПИСИ*/
        pos, /*НУМЕР ПОЗИЦИИ*/
        j, /*ПЕРЕМЕННАЯ ЦИКЛА*/
        s1[10],s[10];
    struct RECORD b;
    fseek(fv,0,SEEK_SET);
    while(i<((sizeof(a)+1)*(N-1))) /* Читаем первую из сравниваемых записей*/
    {
        fseek(fv,i,SEEK_SET);
        fread(a.fio,sizeof(a.fio),1,fv);
        fread(s,sizeof(a.age),1,fv);
        fseek(fv,f.tell(fv)+1,SEEK_SET);
        fread(a.adr,sizeof(a.adr),1,fv);
        j=i+sizeof(a)+1;
        while (j<((sizeof(a)+1)*N)) /* Читаем вторую из сравниваемых записей */
        {
            fseek(fv,j,SEEK_SET);
            fread(b.fio,sizeof(b.fio),1,fv);
            fread(s1,sizeof(b.age),1,fv);
            fseek(fv,f.tell(fv)+1,SEEK_SET);
            fread(b.adr,sizeof(b.adr),1,fv);
            if (strcmp(a.fio,b.fio)>0) /*Если первая запись больше второй, то ...*/
                /*ОБМЕН ЗАПИСЕЙ*/
                {
                    pos=i;
                    fseek(fv,pos,SEEK_SET);
                    fwrite(b.fio,sizeof(b.fio),1,fv);
                    fwrite(s1,sizeof(b.age)+1,1,fv);
                    fwrite(b.adr,sizeof(b.adr),1,fv);
                    pos=j;
                    fseek(fv,pos,SEEK_SET);
                    fwrite(a.fio,sizeof(a.fio),1,fv);
                    fwrite(s,sizeof(a.age)+1,1,fv);
                    fwrite(a.adr,sizeof(a.adr),1,fv);
                    strcpy(s,s1);
                    strcpy(a.fio,b.fio);
                    strcpy(a.adr,b.adr);
                }
            j+=sizeof(a)+1; } /*if*/
        i+=sizeof(a)+1; } /*while j*/
    }
}
/*while i*/

```

11.3 Варианты заданий

Во всех вариантах результатами являются файлы; в задании 3 сохранить в файле-результате структуру исходного файла.

1. Крупнейшие автомобильные монополии.

Наименование	Год создания	Активы, млн. долл.
Дженерал моторз	1916	12916
Форд мотор	1902	8090
Крайслер	1925	3149
Фиат	1899	1401
Фольксваген	1930	1434
Рено	1895	590
Ниссан мотор	1953	1295
Вольво	1926	379

- 1) найти старейшую компанию;
- 2) найти наименование компаний с активом, не более 1000 млн. долл.;
- 3) найти и упорядочить по алфавиту наименования компаний, созданных после окончания I Мировой войны.

2. Показатели развития банков России к 1914 г.

Наименование	Число филиалов	Капитал (млн. руб.)
Русско-Азиатский	102	78
Азово-Донской	7	92
Русский торгово-промышленный	111	44
Сибирский торговый	57	36
Соединенный	80	35

- 1) найти банк с наибольшим капиталом;
- 2) найти банки с числом филиалов больше 50 млн. рублей;
- 3) найти и упорядочить в порядке, обратном алфавитному, наименования банков с капиталом, меньшим 70 млн. рублей.

3. Экспорт алмазов стран Африки, тыс. карат.

Страна	1965	1966
Конго	10000	12000
Гана	3084	1499
ЮАР	2204	2630
Намибия	1590	1694
Либерия	540	555
Танзания	828	905

- 1) найти страну с минимальным экспортом алмазов в 1965 г.;
- 2) найти страны, экспорт алмазов которых в 1966 году превысил 1000 тыс. каратов;

- 3) найти и упорядочить по алфавиту страны, чей экспорт в 1966 году не превысил 2000 тыс. каратов.

4. Крупнейшие авиа- и моторостроительные монополии.

Наименование монополии	Год создания	Собственный капитал (млн. \$)
Боинг	1916	564
Доннел-Дуглас	1967	323
Локкид эркрафт	1932	318
Бритиш эркрафт	1969	47

- 1) найти компанию с минимальным капиталом;
- 2) найти компании, созданные после II Мировой войны;
- 3) найти и упорядочить в порядке, обратном алфавитному, наименования компаний с капиталом, превышающим 300 млн. \$.

5. Добыча полезных ископаемых в Австралии.

Наименование	Единица измерения	1952	1967
Медь	тыс. тонн	18.9	91.7
Золото	тыс. кг	30.6	25.2
Свинец	тыс. кг	213.9	381.0
Железная руда	тыс. тонн	1913.0	12273.0
Каменный уголь	млн. тонн	19.7	35.3
Бокситы	тыс. тонн	7.4	4244.0

- 1) найти полезное ископаемое, добыча которого в 1952 году была максимальной;
- 2) найти полезные ископаемые, добыча которых в 1967 году превысила 10 тыс. тонн;
- 3) найти и упорядочить по алфавиту полезные ископаемые, добыча которых в 1952 году не превысила 20 тыс. тонн.

6. Виды промышленной продукции Японии.

Вид	Единица измерения	1950	1958
Уголь каменный	млн. т	38.5	49.7
Сырая нефть	тыс. т	328.4	372.0
Железная руда	тыс. т	825.9	2004.0
Медь	тыс. т	84.7	123.7
Серная кислота	млн. т	3.2	3.8
Цемент	млн.т	4.4	14.9

1) найти вид продукции, чье производство было максимальным в 1958 г.;

2) найти виды продукции, чье производство в 1950 году превысило 400 тыс. т;

3) найти и упорядочить в порядке, обратном алфавитному, наименования видов продукции, чье производство в 1958 г. не превысило 4 млн. т.

7. Производство электроэнергии в развитых странах (млрд. кВатт·час).

Страна	1955	1958
США	629.0	724.0
Англия	93.9	112.9
Канада	82.8	96.7
ФРГ	76.5	95.1
Япония	65.2	81.2
Франция	49.6	61.8
Швеция	24.7	30.4

1) найти страну, которая в 1958 году произвела больше всех электроэнергии;

2) найти страны, в которых в 1955 году производство электроэнергии превысило 70 млрд. кВатт·часов;

3) найти и упорядочить по алфавиту страны, в которых в 1958 г. производство электроэнергии не превысило 100 млрд. кВатт·ч.

8. Производство сельхозяйственной продукции в Австралии.

Наименование	Площадь	
	1960	1967
Пшеница	4869	8329
Ячмень	952	999
Сахарный тростник	195	268
Овес	1212	1703
Травы	842	1398

1) найти культуру, которая занимала максимальную посевную площадь в 1960 году;

2) найти культуры, посевная площадь которых в 1967 году превысила 1000 тыс. га;

3) найти и упорядочить в порядке, обратном алфавитному, названия культур, чья посевная площадь в 1960 году не превысила 1300 тыс. га.

9. Доля развитых стран в промышленном производстве.

Страна	1953	1958
США	53.3	46.9
ФРГ	7.8	10.5
Франция	4.0	5.4
Италия	3.1	3.8
Канада	3.6	3.5
Япония	2.2	3.2

1) найти страну, доля в промышленном производстве которой в 1958 году была минимальной;

2) найти страны, доля промышленного производства которых в 1953 году не превысила 6.0 %;

3) найти и упорядочить по алфавиту страны, доля промышленного производства которых в 1958 г. превысила 4.0 %.

10. Производство важнейших видов промышленной продукции в развитых странах (в млн. т.).

Страна	Сталь	Уголь	Нефть
США	102.3	466.60	353.60
ФРГ	24.5	162.20	4.00
Англия	22.0	227.20	0.08
Франция	14.1	58.20	1.40
Италия	6.8	1.15	1.20
Канада	4.6	11.10	24.50
Япония	12.6	52.80	0.22

1) найти страну с максимальным производством стали;

2) найти страны, добыча нефти в которых превысила 1 млн. т;

3) найти и упорядочить в порядке, обратном алфавитному, страны, добыча угля в которых не превысила 200 млн. т.

11. Распределение лесных ресурсов по странам.

	Площадь лесного фонда	
	в млн. га	в % от общей территории
СССР	1131	50
Канада	342	34
Бирма	39	58
Конго	100	43
Япония	22.6	59
Уругвай	0.45	2
Нидерланды	0.25	8

- 1) найти страну с минимальной площадью лесного фонда;
- 2) найти страны с площадями лесного фонда большими 1 млн. га;

3) найти и упорядочить по алфавиту страны, процент площади лесного фонда от общей территории которых не превышает 50.

12. Динамика реального ВНП и реального ВНП на душу населения в некоторых странах - U.S. Department of Commerce, Historical Statistics of the United States: Colonial Times to 1970, Washington, 1975. P.225; Economic Report of the President, 1989.

Страны	Темпы роста реального ВНП (%)		Темпы роста реального ВНП на душу населения (%)	
	1870 - 1969	1950 - 1988	1870 - 1969	1950 - 1988
США	3.7	3.3	2.0	1.9
Япония	4.2	7.1	-	5.9
ФРГ	3.0	5.0	1.9	4.2
Англия	1.9	2.6	1.3	2.2
Франция	2.0	4.1	1.7	3.3
Италия	2.2	4.4	1.5	3.9
Канада	3.6	4.5	1.8	2.7

1) найти страну, темп роста реального ВНП в которой за период 1950-1988 гг. был минимален;

2) найти страны, в которых темп роста реального ВНП на душу населения за период 1950-1988 гг. превысил 3 %;

3) найти и упорядочить в порядке, обратном алфавитному, страны, темп роста реального ВНП которых за период 1870-1969 гг. не превысил 3.5 %.

13. Распределение лесных ресурсов по частям света - МСЭ, т. 5, с. 503.

Площадь лесного фонда		
	В млн га	В % от общей территории
Европа	314	31
Азия	1477	33
Африка	807	27
Америка	1577	37
Австралия	85	10

- 1) найти часть света с минимальной площадью лесного фонда
- 2) найти части света, площади лесного фонда которых превышают 500 млн. га;

3) найти и упорядочить по алфавиту части света, процент площади лесного фонда от общей территории которых не превышает 35.

14. Показатели по Ленинскому районному центру занятости г. Donetsk за 1996 г. (из годового статистического отчета по Ленинскому РЦЗ).

Квартал	Численность населения в районе (тыс. чел.)	Численность работящего населения (тыс. чел.)	Численность предоставляемых вакансий	Число обратившихся в центр занятости (чел.)	Число зарегистрированных безработных (чел.)
1	117,4	55,0	159	269	90
2	117,4	54,1	79	584	85
3	117,4	53,8	73	581	154
4	117,4	52,0	54	1876	203

1) в каком квартале число обратившихся в центр занятости было максимальным;

2) в каких кварталах численность предоставляемых вакансий не превысила 100;

3) в каких кварталах число безработных не превысило 200 человек (результат упорядочить по убыванию).

15. Состояние валют - МСЭ, т. 3, с. 303.

Валюта	Снижение содержимого золотого в г чистого золота		Падение покупательной способности (1957 в % к 1929)
	1929	1957	
Доллар США	1,5	0,89	52,4
Фунт стерлингов	7,32	2,49	31,6
Французский франк	0,059	0,002	3,4
Японская йена	0,75	0,0025	0,76

1) найти валюту, падение покупательной способности которой максимально;

2) найти валюты, снижение содержимого золотого в г чистого золота которых превысило 1.0 в 1929 году;

3) найти и упорядочить в порядке, обратном алфавитному, валюты, снижение содержимого золотого в г чистого золота которых не превысило 1.0 в 1957 году.

16. Объем внешней торговли СССР по некоторым странам Европы в 1984 г.

Страна	Оборот	Экспорт	Импорт
Австрия	1352.8	565.7	787.1
Бельгия	1602.0	990.6	611.4
Болгария	10564.1	5510.8	5053.3
Великобритания	1816.8	1184.8	632.0
Венгрия	8065.0	4058.0	4007.0
ГДР	13393.5	6797.8	6595.7
Греция	687.7	532.2	154.5
Дания	355.3	290.3	65.0
ФРГ	404.7	293.7	111.0
Исландия	105.2	56.3	48.9
Ирландия	103.8	22.9	80.9

- 1) найти страну, оборот торговли с которой был максимальен;
 2) найти страны, экспорт товаров в которую из СССР превысил 1000;

3) основные показатели развития морского транспорта в СССР.

Год	Грузооборот (млрд. т-км)	Перевезено грузов (млн. т.)	Пассажирооборот (млрд. пасс.-км)	Перевезено пассажиров (млн. человек)
1913	19.9	13.9	1.0	3.4
1940	23.8	31.2	0.9	3.1
1945	34.2	20.2	0.6	1.0
1956	82.4	57.7	1.4	8.2

- 1) в каком году пассажирооборот был минимальен;
 2) в каких годах перевезено не более 40 млн. т грузов;
 3) в каких годах перевезено более 2 млн. пассажиров

(результат упорядочить по убыванию).

18. Степень концентрации производства в некоторых олиополистических отраслях США, 1989 г. - Учебник по основам экономической теории под редакцией Камаева В.Д. и др.

Отрасль	Число продавцов	Годовые поставки четырех крупнейших компаний, %
Алюминиевая	15	64
Силиконовая	5	99
Автомобильная	28	92
Холодильники	18	94
Жареный кофе	118	65
Жевательная резинка	9	95
Кукурузный сахар	19	65

- 1) найти отрасль промышленности, в которой годовые поставки четырех крупнейших компаний максимальны;
 2) найти отрасли промышленности, число продавцов в которых превышает 10;

3) найти и упорядочить в порядке, обратном алфавитному, отрасли промышленности, в которых годовые поставки четырех крупнейших компаний не превышают 95 %.

19. Выпуск обуви в 1957 г. - МСЭ, т. 4, с. 922.

Страна	Обувь кожаная (млн. пар)	На душу населения (пар)
СССР	536	1.72
США	593.6	3.5
Англия	142.7	2.71
Франция	79	1.82
Чехословакия	32	2.42
Польша	41.1	1.33
Румыния	15.8	0.89

- 1) найти страну, в которой производилось меньше всего пар обуви;
 2) найти страны, в которых на душу населения производилось более 2 пар обуви;
 3) найти и упорядочить по алфавиту страны, в которых производство обуви не превысило 500 млн. пар.

20. Удельный вес развитых стран в промышленном производстве (%) - МСЭ, т. 6, с. 562.

Страна	1937	1948	1958
США	41.4	56.4	46.9
Англия	12.5	11.7	8.2
Франция	6.0	4.1	5.4
ФРГ	9.0	4.3	10.5
Япония	4.8	1.5	3.2
Италия	3.0	2.1	3.8

- 1) найти страну с минимальным удельным весом в 1958 г.;
- 2) найти страны с удельным весом в 1948 г. большим 4 %;
- 3) найти и упорядочить в порядке, обратном алфавитному, страны с удельным весом в 1937 году не большим 10 %.

21. Движение показателей себестоимости и цена на металлорежущие станки.

Год	Средняя себестоимость 1 штуки изделия	Средняя оптовая цена предприятия	Рентабельность по отношению к себестоимости, %
1966	1.00	1.00	16.8
1969	1.23	1.30	23.1
1970	1.32	1.40	23.5
1972	1.42	1.51	23.8
1974	1.57	1.64	21.8

1) в каком году средняя себестоимость одного металлорежущего станка была максимальной;

2) когда средняя оптовая цена предприятия быланей 1.3;

3) в каких годах рентабельность по отношению к себестоимости превысила 20 % (результат упорядочить по убыванию).

22. Посевные площади и сбор сельскохозяйственных культур в Канаде - МСЭ, т. 4, с. 159.

Наименование культуры	Посевная площадь (тыс. га)		Сбор (тыс. т)	
	1954	1957	1954	1957
Пшеница	10573	8509	14633	10165
Овес	4381	4458	6385	5931
Ячмень	3204	3804	4970	4790
Кормовые травы	4285	4633	16799	16517
Льняное семя	409	411	453	586

- 1) сбор какой культуры был минимален в 1957 г. (в тыс. т);
- 2) сбор каких культур в 1954 году превысил 10000 тыс. тонн;
- 3) найти и упорядочить по алфавиту названия культур, посевная площадь которых в 1957 г. не превысила 5000 тыс. га.

23. Производство масла и сыра (в тыс. т) - МСЭ, т. 6, с. 137.

Страна	Масло	Сыр
США	641	632
Дания	166	84.5
Швеция	82.4	50.6
Нидерланды	76.8	153
ФРГ	301	157
Франция	305	324

- 1) найти страну с минимальным производством масла;
- 2) в каких странах производство масла превысило 200 тыс. т;
- 3) найти и упорядочить в порядке, обратном алфавитному, страны, производство сыра в которых не превысило 200 тыс. т.

24. Производство молочных продуктов в СССР (в тыс. т) - МСЭ, т. 6, с. 137.

Наименование продукта	1913	1932	1945	1957
Масло	104.1	71.6	117.3	635
Сыр	7.9	14.3	25.6	145.1
Сухое молоко	-	0.034	1.1	39.1

1) производство какого молочного продукта (в тыс. тонн) в 1957 году было минимальным;

2) производство каких молочных продуктов в 1945 г. превысило 100 тыс. т;

3) найти и упорядочить по алфавиту названия продуктов, производство которых в 1932 г. превысило 10 тыс. т.

25. Изменения структуры грузооборота по видам транспорта (в %).

Вид транспорта	Грузооборот			Пассажирооборот		
	1960	1970	1975	1960	1970	1975
Железнодорожный	79.8	65.1	62.2	68.5	48.0	41.9
Морской	7.0	17.1	14.2	0.5	0.3	0.3
Речной	5.3	4.5	4.2	1.7	1.0	0.8
Трубопроводный	2.7	7.4	12.8	-	-	-
Автомобильный	5.2	5.8	6.5	24.5	36.6	40.6
Воздушный	0.0	0.1	0.1	4.8	14.1	16.4

1) на каком виде транспорта в 1975 г. грузооборот был максимальен;

2) на каком виде транспорта в 1960 году грузооборот превысил 5%;

3) найти и упорядочить в порядке, обратном алфавитному, названия видов транспорта, на которых в 1970 году пассажирооборот не превысил 40 %.

26. Капитальные вложения государственных и кооперативных организаций СССР (в % к итогу).

	Пятилетки				1-5-ая пятилетки
	1	2	4	5	
Промышленность	42.6	41.8	48.9	51.1	48.1
Сельское хозяйство	13.9	8.1	7.3	9.6	8.7
Транспорт и связь	18.6	21.5	14.2	10.1	12.8
Жилищное строительство	11.8	10.2	12.7	15.5	14.8
Прочее строительство	13.1	18.4	16.9	13.7	15.6

1) в какую отрасль народного хозяйства в 5-ю пятилетку было вложено больше всего средств;

2) в какие отрасли народного хозяйства капитальные вложения во время 2-й пятилетки превысили 15 %;

3) найти и упорядочить по алфавиту наименования отраслей народного хозяйства, капитальные вложения в которые во время 4-й пятилетки не превысили 20 %.

27. Структура основных фондов промышленности СССР на 1.01.1956 (в % к итогу) - МСЭ, т.6, с.1011.

Отрасль промышленности	Здания	Сооружения	Силовое оборудование	Производственное оборудование	Транспортные средства
Черная металлургия	25.7	21.2	7.4	32.8	4.6
Нефтяная	9.6	51.8	3.8	18.9	3.4
Электростанции	15.0	30.8	28.2	7.5	1.3
Легкая	31.4	6.9	6.6	47.5	2.8

1) какая отрасль промышленности имела больше всего транспортных средств;

2) производственное оборудование каких отраслей промышленности превысило 20 %;

3) найти и упорядочить в порядке, обратном алфавитному, наименования отраслей промышленности, которые имели в качестве основных фондов не менее 20 % зданий.

28. Действующие международные картели - МСЭ, т. 4, с. 579-580.

Название товара	Количество стран-участниц	Год образования	Основные условия карельного соглашения
Сталь	4	1926	Регулирование экспорта
Калий	3	1925	Распределение рынков сбыта
Электрические лампы	10	1925	Распределение рынков сбыта
Титановые белила	7	1920	Обмен патентами
Хинин	7	1890	Соглашение с плантаторами
Кабель	12	1907	Распределение рынков сбыта

1) какой вид продукции производит самый старый картель;

2) найти виды продукции, с которыми работают картели, количество стран-участниц в которых превышает 5;

3) найти и упорядочить по алфавиту названия видов продукции, с которыми работают картели, возникшие не ранее 1900 года.

29. Индексы показателей эффективности производства автомобилей ВАЗ-2101 по мере увеличения его выпуска Волжским автозаводом.

Год	Товарный выпуск	Себестоимость автомобиля	Оптовая цена	Рентабельность
1970	1.00	1.00	1.00	2.3
1971	8.00	0.70	0.73	5.4
1972	14.42	0.54	0.65	22.8
1973	17.65	0.49	0.58	19.1
1974	17.14	0.43	0.48	12.7

1) в каком году рентабельность производства автомобилей была максимальной;

2) в каких годах товарный выпуск автомобилей превысил 10.00;

3) в каких годах оптовая цена автомобиля не превысила 0.9, если за 1.00 берется оптовая цена автомобиля в 1970 году (результат упорядочить по возрастанию).

30. Структура оборотных фондов промышленности СССР на 1.1.1955 г. (в %) - МСЭ, т. 6, с. 763.

Отрасли промышленности	Производственные запасы	Незавершенное производство и полуфабрикаты	Прочие
Машиностроение	47.3	43.5	9.2
Металлургия	64.1	19.7	16.2
Электростанции	95.8	1.7	2.5
Легкая и пищевая	79.8	17.4	2.8

1) какая отрасль промышленности имеет самые большие производственные запасы;

2) какие отрасли промышленности в разделе "прочие" имеют оборотные средства, превышающие 5 %;

3) найти и упорядочить в порядке, обратном алфавитному, наименования отраслей промышленности, у которых незавершенное строительство и полуфабрикаты не превышают 40 %.

31. Девять крупнейших банков мира.

Банк	Страна	Активы (млрд. \$)	Вклады служащих (млрд. \$)	Число служащих
"Дайти Канте Бэнк" (Di-Ichi Kangyo Bank)	Япония	288.4	211.1	192293
"Сумитомо бэнк" (Sumitomo Bank)	Япония	270.2	194.1	16610
"Фудзи бэнк" (Fuji Bank)	Япония	263.1	187.0	14805
"Мицубиси бэнк" (Mitsubishi Bank)	Япония	245.3	177.7	14296
"Санва бэнк" (Sanwa Bank)	Япония	238.1	174.8	14827
"Индастриэл бэнк оф Джапан" (Industrial Bank of Japan)	Япония	222.5	175.6	5363222
"Креди агрикол" (Kredit agricole)	Франция	214.7	151.1	73939
"Ситикорп" (Citicorp)	США	203.6	119.6	90000
"Дойче банк" (Deutsche Bank)	ФРГ	170.8	154.0	54579

1) в каком банке больше всего служащих;

2) активы каких банков превышают 240 млрд. \$;

3) найти и упорядочить по алфавиту названия банков, вклады служащих которых не превышают 180 млрд. \$.

32. Производство основных видов продукции пищевой промышленности (в 1951 г.) - МСЭ, т. 7, с. 172.

Страна	Мясо (млн. т)	Консервы (млн. банок)	Рыба (млн. т)	Масло животное (тыс. т)
США	12.7	36122	2.94	707
Англия	1.7	3268	1.05	35
Франция	2.5	1363	0.54	300
ФРГ	2.4	2500	0.77	334
Италия	0.94	1377	0.22	68

1) в какой стране было выпущено больше всего млн. банок консервов;

2) найти страны, в которых производство животного масла превышало 300 тыс. тонн;

3) найти и упорядочить в порядке, обратном алфавитному страны, в которых производство мяса не превысило 10 млн. тонн.

33. Производство основных видов продукции пищевой промышленности в СССР (в тыс. т) - МСЭ, т. 7, с. 171.

Виды продукции	1913	1928	1940	1945	1958
Мясо	1042	678	1561	663	2364
Рыба	1018	846	1404	1125	2931
Масло животное	104	82	226	117	650
Масло растительное	471	448	708	242	1146
Сахар-песок	1347	1293	2165	465	5434

1) найти вид продукции, производство которого было максимальным в 1958 году;

2) найти те виды продукции, производство которых в 1940 году превысило 1000 тыс. тонн;

3) найти и упорядочить по алфавиту наименования видов продукции, производство которых в 1913 году не превысило 1200 тыс. тонн.

34. Административное деление Австралии, 1969г.

Штат	Площадь, тыс. км ²	Население, тыс. чел
Новый Южный Уэльс	801.4	4382.4
Виктория	277.6	3324.2
Квинсленд	1726.7	1732.4
Южная Австралия	984.4	1125.4
Западная Австралия	2527.6	904.4
Тасмания	68.4	382.0

- 1) площадь какой страны минимальна;
- 2) население каких стран не превысило 3000 тыс. чел.;
- 3) найти и упорядочить в порядке, обратном алфавитному, страны с площадью, большей 500 тыс. км².

35. Высокопрочные магниевые сплавы.

Марка	Вид полуфабриката	Удлинение, %
МЛ4	отливки	7.0
МЛ5	отливки	7.0
МЛ6	отливки	1.5
МЛ12	отливки	8.0
МЛ8	листы	12.0
ВМ65-1	прутки	10.0

- 1) найти сплав с минимальным удлинением;
- 2) найти марки сплавов, поставляемых в виде отливок;
- 3) найти и упорядочить по алфавиту марки сплавов с удлинением больше 9%.

36. Нормальный состав крови человека.

	Цельная кровь, %	Плазма, %	Эритроциты, %
Вода	80.0	90.5	62.5
Сухой остаток	21.0	10.0	38.5
Гемоглобин	15.3	0.0	35.5
Сахар	9.5	10.0	60.0
Лецитин	32.5	21.5	47.5
Холестерин	19.0	15.0	20.5
Калий	20.0	16.0	35.0

- 1) найти составную часть крови, в которой содержание цельной крови максимальное;
- 2) найти составную часть крови, в которой содержание эритроцитов больше 50%, и плазмы меньше 95%;
- 3) найти и упорядочить в лексикографическом порядке составляющие крови, у которых цельной крови больше 10%.

37. Режимы проявления кинопленок.

Операция	Раствор	Продолжительность (мин)	Температура (°C)
Первое проявление	Орвоколор-09	32	19
Промывка	Водопроводная вода	25	10–16
Цветное проявление	Орвоколор-13	11	19
Промывка 2	Водопроводная вода	25	10–16
Отбеливание	Орвоколор-57	5	19
Промывка 3	Водопроводная вода	5	10–15

- 1) найти операции, выполняемые при одинаковой t°;
- 2) найти растворы, применяемые в операциях с продолжительностью больше 10 минут и температурой больше 20 градусов;
- 3) найти и упорядочить в порядке, обратном лексикографическому, операции, у которых раствор – водопроводная вода.

38. Пищевая ценность продуктов.

Продукт	Белки (%)	Калорийность (ккал)
Шампиньон	45.0	192
Белый гриб	42.5	227
Горошек зеленый	47.0	65
Морковь	0.6	27
Помидоры	0.4	14
Картофель	1.1	65

- 1) продукты с максимальной калорийностью;
- 2) найти продукт с содержанием белков больше 1% и калорийностью больше 150 ккал;
- 3) отсортировать файл в лексикографическом порядке по полю "продукт".

39. Состав территории Австрии.

Земля	Площадь (тыс. кв. км)	Население (тыс. чел)
Вена	0.4	1627.0
Бургенланд	4.0	271.0
Верхняя Австрия	12.0	1131.0
Зальцбург	7.1	347.3
Каринтия	9.5	495.2
Нижняя Австрия	19.2	1374.0
Тироль	12.6	462.9
Форальберг	2.6	226.4
Штирия	16.4	1137.9

- найти плотности населения земель;
- найти максимальную плотность;
- отсортировать файл в порядке, обратном алфавитному, по полю "земля".

40. Восхождение спортсменов на "семитысячники" Памира и Тянь-Шаня.

Вершина	Количество восходителей	Дата
Хан-Тегри	3	11.09.1931
Пик Коммунизма	1	03.09.1933
Пик Ленина	3	08.09.1934
Пик Победы	3	19.09.1938

- найти восхождения (вершина, дата), совершенные в одиночку;
- найти вершину с самой ранней датой покорения;
- отсортировать файл в лексикографическом порядке по полю "вершина".

41. Спецификации деталей шахтной крепи.

Наименование	Материал	Вес (кг)
Верхняк	Ст.5	44.900
Стойка	Ст.5	93.600
Стятка	Ст.5	33.000
Планка	Ст.3	13.720
Скоба	Ст.3	33.300
Гайка	Ст.3	1.078

- найти детали из Ст.5 и весом более 20 кг;
- найти деталь с минимальным весом;
- отсортировать файл в порядке, обратном алфавитному, по полю "наименование".

42. Эвтектические сплавы с галлием.

Сплав	Температура плавления (°C)	Содержание Ga (%)
I	3.0	61
II	5.0	62
III	13.0	67
IV	16.0	76
V	17.0	82
VI	27.3	99.5

- найти сплавы с температурой плавления больше 15 °C;
- найти сплав с минимальным содержанием Ga;
- отсортировать файл в лексикографическом порядке по полю "сплав".

43. Наиболее крупные острова.

Наименование	Площадь (тыс. км ²)	Принадлежность	Часть света
Гренландия	2176.0	Дания	Европа
Калимантан	734.0	Индонезия	Азия
Мадагаскар	590.0	Мадагаскар	Африка
Баффинова Земля	512.0	Канада	Америка
Суматра	435.0	Индонезия	Азия
Великобритания	230.0	Англия	Европа
Хонсю	230.0	Япония	Азия
Виктория	208.0	Канада	Америка
Элсмир	200.5	Канада	Америка

- найти государства Азии, имеющие острова площадью больше 250 тыс. км²;
- найти остров с минимальной площадью;
- найти и упорядочить в порядке, обратном лексикографическому, страны Европы.

44. Физико-химические показатели основных типов пластмасс.

Показатель	Тип пластмасс		
	целлULOид	этилцеллюлозный этрол	ацетилцеллюлозный этрол
Плотность (г/см ³)	1.33	1.45	1.35
Водопоглощение	1.40	0.60	1.60
Пробивное напряжение	21.00	14.50	25.00
Горючесть	сильно горюч	негорюч	негорюч

- найти негорючие типы пластмасс с пробивным напряжением больше 15;
- найти тип пластмассы с минимальным водопоглощением;
- отсортировать файл в лексикографическом порядке по полю "Показатель".

45. Термопрочные магниевые сплавы.

Марка	Вид полуфабриката	Хим. Состав, % (кроме Mg)
МЛ-7	Отливки	In — 0.5
МЛ11	Отливки	In — 0.6
ВМ17	Прутки	Mn — 1.8

- найти марки сплавов, поставляемых в отливках и содержащих In;
- найти марки сплавов с минимальным содержанием In;
- найти и упорядочить в порядке, обратном лексикографическому, марки сплавов.

46. Таблица элементарных частиц.

Название	Время жизни	Спин
Пи-плюс	$2.56 \cdot 10^{-8}$	0
Пи-ноль	$0.4 \cdot 10^{-15}$	0
Анти-лямбда-ноль	$2.8 \cdot 10^{-10}$	1/2
Анти-кси-минус	$2.0 \cdot 10^{-8}$	1/2
Сигма-минус	$1.6 \cdot 10^{-16}$	1/2
Нейтрон	1040	1/2

- 1) найти время жизни частиц с нулевым спином;
 2) найти частоту с максимальным временем жизни;
 3) найти и упорядочить в порядке, обратном лексикографическому, названия элементарных частиц со спином равным 1/2.

47. Климатические показатели регионов Южной Америки.

Климатический пояс	Высота над уровнем моря (м)	Осадки за год (мм)
Экваториальный	106	2623
Субэкваториальный	60	456
Тропический	61	1102
Субтропический	25	962
Умеренный	274	125

- 1) найти климатический пояс с уровнем осадков больше 200 мм и высотой выше 50 м;
 2) найти пояс, который находится на максимальной высоте над уровнем моря;
 3) найти и упорядочить в порядке, обратном лексикографическому, климатические пояса, у которых осадки меньше 1000 мм.

48. Основные сведения о ядерных электростанциях.

Параметры	Белоярская	Воронежская	Беркли
Давление пара у турбин	90	29	21.8
Температура пара	480	230	318
КПД тепловой части	33	25.8	25
Теплоноситель	вода	вода	графит
Замедлитель	графит	вода	графит
Ядерное топливо	слабо обогащенный уран	слабо обогащенный уран	естественный уран
Форма корпуса реактора	цилиндр	цилиндр	шар

- 1) найти АЭС, у которых замедлителем является графит и форма корпуса реактора - цилиндр;
 2) найти АЭС с максимальной температурой пара;

3) найти и упорядочить в порядке, обратном лексикографическому, параметры;

49. Характеристики некоторых управляемых снарядов.

Название	Тип снаряда (ракеты)	Длина (м)	Дальность полета (м)
Гермес	крылатая	7.6	80
Ф-2	баллистическая	14.3	320
Матадор	крылатая	12.0	885
Атлас	баллистическая	24.4	8800
Матра 510	крылатая	3.2	8

- 1) найти ракеты типа "баллистическая" с длиной больше 10 м;
 2) найти снаряд с минимальной дальностью полета;
 3) найти и упорядочить в лексикографическом порядке названия ракет, у которых тип снаряда - крылатая.

50. Инструментальные средства разработки экспертных систем.

Название	Представление знаний	Средства функционирования
APES	Правила	Prolog
E FOSSI 1	Деревья	Prolog
GURU	Правила	C
KANDOR	Фреймы	Lisp
FIT	Логика	Lisp

- 1) найти наименование инструментальных средств, использующих правила и функционирующих в среде Prolog;
 2) найти количество средств, функционирующих в среде Lisp;
 3) найти и упорядочить в лексикографическом порядке названия систем, у которых среда функционирования - Lisp.

51. Состав и свойства цинковых сплавов.

Сплав	Твердость (кг/мм ²)	Температура кристаллизации (°C)
I	90	390
II	110	419
III	85	424
IV	100	420
V	97	395

- 1) найти сплав с твердостью больше 85 кг/мм² и температурой больше 400 градусов;
 2) найти сплав с максимальной температурой кристаллизации;
 3) найти и упорядочить в лексикографическом порядке названия сплавов, у которых твердость меньше 100 кг/мм².

52. Характеристики современных Лисп-машин.

Модель	Емкость ОЗУ (Мслов)	Емкость ВЗУ (Мб)
LM-2	0.2	80
Lambda	1.0	470
Symbolics	6.0	169
Alpha	2.0	160
Elis	16.0	540

- 1) найти машины с емкостью ВЗУ больше 100 Мб
- 2) найти машину с минимальной емкостью ОЗУ;
- 3) найти и упорядочить в лексикографическом порядке названия моделей, у которых емкость ОЗУ меньше 10 Мслов.

53. Пищевая ценность колбасных изделий.

Группа колбасных изделий	Белки (%)	Калорийность 100г в Ккал
Вареные	20	300
Полукопченые	20	450
Копченые	30	500
Ливерно-паштетные	16	400

- 1) найти группу изделий с калорийностью больше 300 Ккал и белками больше 20%;
- 2) найти группу изделий с минимальным содержанием белков;
- 3) найти и упорядочить в лексикографическом порядке названия групп изделий, у которых содержание белка больше 10 %.

54. Этапы развития языка ПРОЛОГ.

Год	США	Англия	Франция	Швеция	Япония
1979	ALDT-1 (Либерман)	IC Prolog (Кларк)	Argos II (Кейлор)	QLOG (Коморовски)	
1979	ETHER (Цорнфельд)				
1979	OPS (Форджи)				
1981	BRANDX (Шолович)	MICROPROLOG (Мак Кейб)			
1981	PEARL (Диринг)				
1982					FGKL (Мото Ока)

- 1) найти системы, которые разработаны до 1980 года в США;
- 2) в какой из стран разработано наибольшее количество систем;
- 3) найти и упорядочить в лексикографическом порядке названия систем, разработанных в США.

55. Спецификация деталей на одну раму крепления выработок на шахтах Донбасса.

Номер позиции	Наименование деталей	Размер	Количество	Материал
1	Верхняк	Профиль Б	1	Ст.5
2	Стойка	Профиль А	2	Ст.5
3	Стяжка	1/3 профиля А или Б	3	Ст.5
5	Планка	210*70*18	4	Ст.3
6	Болт	M20	11	Ст.3
7	Гайка черная	M20	11	Ст.3
8	Башмак	профиль А или Б	2	Ст.5

- 1) найти детали с профилем А или Б из Ст.5;
- 2) найти номер позиции с наименьшим количеством деталей;
- 3) найти и упорядочить в лексикографическом порядке названия деталей, у которых размер - профиль А.

56. Информация о площади и населении отдельных государств.

Название	Площадь (тыс. км ²)	Население (тыс. чел.)	Столица
Франция	544	55860	Париж
Канада	9976	25880	Оттава
США	9363	246880	Вашингтон
Польша	312.7	37853	Варшава
Болгария	110.9	8986	София
Украина	603.7	51200	Киев

- 1) найти страны с площадью больше 250 тыс.км² и населением больше 3000 тыс.чел;

- 2) найти страну с максимальной площадью;
- 3) найти и упорядочить в порядке, обратном лексикографическому, по полю "столица" страны, у которых площадь больше 500 тыс.км².

57. Принтеры.

Модель	MIN цена, \$	Общ. Цена, \$	Измененис за месяц
HP LASERJET 5L	450	538	-1.30%
HP LASERJET 6P	884	969	-2.30%
HP LASERJET 4V	2250	2579	2.20%
EPSON EPL-9000	1836	1927	-0.60%

1) найти принтеры с минимальной ценой больше 2000\$ и общей ценой больше 1000\$;

2) найти принтер с максимальной общей ценой;

3) найти и упорядочить в лексикографическом порядке названия принтеров, у которых изменение в цене за месяц отрицательное.

58. Калорийность съедобной части некоторых рыб.

Название	Вода, %	Белки, %	Жиры, %	Калорийность(100г, ккал)
Минога	55.1	12.5	27.3	365
Осетр	71.4	15.6	9.8	155
Лещ	74	17.9	5.4	123
Судак	78.9	18	0.7	81

1) найти вид рыбы, у которого содержание воды больше 50%, белков больше 15%, жиров меньше 10%;

2) найти вид рыбы с максимальной калорийностью;

3) найти и упорядочить в лексикографическом порядке названия рыб, у которых калорийность больше 100 ккал.

59. Административно-территориальный состав Украины.

Область	Территория	Население	Центр
Донецкая	26.5	4934	Донецк
Киевская	29	3530	Киев
Одесская	33.3	2417	Одесса
Луганская	26.7	2759	Луганск
Днепропетровская	31.9	3382	Днепропетровск
Запорожская	27.2	1801	Запорожье

1) найти область с территорией больше 27 и населением больше 3000;

2) найти область с минимальным количеством населения;

3) найти и упорядочить в лексикографическом порядке названия областей, у которых территория меньше 30.

60. Кремниевые транзисторы.

Тип прибора	Структура	Класс	$I_{(кв)max}, A$	$U_{(кв)max}, В$
КТ812А	N-P-N	Низкочастотный	8	700
КТ819Г	N-P-N	Низкочастотный	10	100
КТ837В	P-N-P	Низкочастотный	7.5	70
КТ908А	N-P-N	Высокочастотный	10	100
КТ933А	P-N-P	Высокочастотный	0.5	80
КТ919В	N-P-N	Сверхвысокочаст.	0.2	45
КТ970А	N-P-N	Сверхвысокочаст.	13	65

1) найти приборы со структурой N-P-N, низкочастотные и $I_{(кв)max}$ больше 7.5;

2) найти транзисторы с максимальным $U_{(кв)max}$;

3) отсортировать файл в лексикографическом порядке по полю "тип прибора".

61. Вершины на континенте Евразия.

Название	Горная система	Высота над уровнем моря (м)
Джомолумгма	Гималаи	8 848
Чогори	Каракоум	8 611
Нангапарбат	Гималаи	8 126
Тиричмир	Гиндукуш	7 690
Пик Коммунизма	Памир	7 495
Эльбрус	Большой Кавказ	5 642

1) найти вершины горной системы Гималаи с высотой больше 7500м;

2) найти вершину с максимальной высотой на уровне моря;

3) найти и упорядочить в лексикографическом порядке названия вершин горной системы Гималаи.

62. Содержание витаминов в пищевых продуктах.

	B1	B2	PP	C
Капуста свежая	0.06	0.05	0.4	30
Капуста квашеная	0.02	0.02	0.3	20
Капуста квашеная без рассола	0.02	0.03	0.3	----
Дрожжи пивные	5	4	40	----
Дрожжи пекарские прессов.	0.45	2	28	----
Дрожжи пекарские сухие	1.8	8	112	----

1) найти пищевые продукты, у которых содержания витамина C

больше 5 мг и витамина PP больше 0.02;

2) найти пищевой продукт, у которого отношение витаминов B1/B2 максимальное;

3) упорядочить в порядке, обратном лексикографическому, названия продуктов.

11.3 Контрольные вопросы по лабораторной работе

1) Сформировать файл прямого доступа с помощью функций языка Си.

2) Реализовать на языке Си операцию добавления в конец файла прямого доступа.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. Градштейн И.С., Рыжин И.М. Таблица интегралов, сумм рядов и произведений. - М.: Физматгиз, 1963. - 754с.
2. Уинер Р. Язык Турбо Си. - М.: Мир, 1991. - 384с.
3. Абрамов С.А., Гнездилова Г.Г., Капустина Е.Н. и др. Задачи по программированию. - М.: Наука, 1988. - 224с.
4. Болски М.И. Язык программирования Си: Справочник. - М.: Радио и связь, 1988. - 96с.
5. Керниган Б., Ритчи Д. Язык программирования Си. - М.: Финансы и статистика, 1990. - 230с.
6. Керниган Б., Ритчи Д., Фьюэр А. Язык программирования Си. Задачи по языку Си. - М.: Финансы и статистика, 1985. - 279с.
7. Сван Т. Освоение Borland C++ 4.5. Энциклопедия функций. - К.: Диалектика, 1996. - 320с.
8. Уэйт М., Прата С., Мартин Д. Язык Си: Руководство для начинающих. - М.: Мир, 1984. - 512с.
9. Билецкий Я. Энциклопедия языка Си. - М.: Мир, 1992. - 687с.

Приложение А

График выполнения лабораторных работ

Таблица A1.1 - График лабораторных работ во втором семестре

N ра- бо- ты	Срок выполн- ения (нед.)	Тема	дата выдачи (нед.)	дата защиты (нед.)
6	2	Рекурсия в языке Си	1	3
7	2	Работа с файлами последовательного доступа на языке Си	3	5
8	2	Графические средства Turbo-C	5	7
9	2	Работа с динамическими структурами данных на языке Си	7	9
10	2	Работа с файлами прямого доступа на языке Си	9	11
11	2	Календарные задачи на языке Паскаль	11	13
12	2	Решение задач аналитической геометрии на языке Паскаль	13	15
13	2	Работа с многочленами и матрицами на языке Паскаль	15	17

Приложение Б

Основные именованные константы графической библиотеки TURBO-C

Таблица Б.1 - Константы для обозначения цветов в функции *setcolor*

Константа	Значение	Письмо
BLACK	0	Черный
BLUE	1	Синий
GREEN	2	Зеленый
CYAN	3	Бирюзовый
RED	4	Красный
MAGENTA	5	Малиновый
BROWN	6	Коричневый
LIGHGRAY	7	Светло-серый
DARKGRAY	8	Темно-серый
LIGHTBLUE	9	Голубой
LIGHTGREEN	10	Светло-зеленый
LIGHTCYAN	11	Светло-бирюзовый
LIGHTRED	12	Светло-красный
LIGHTMAGENTA	13	Светло-малиновый
YELLOW	14	Желтый
WHITE	15	Белый

Таблица Б.2 - Константы для установки стиля (вида) и ширины (толщины) линии в функциях *setlinestyle* и *getlinesetting*

Переменная	Константа	Значение	Семантика
line_style	SOLID_LINE	0	Сплошная линия
	DOTTED_LINE	1	Точечная линия
	CENTER_LINE	2	Штрих-пунктирная линия
	DASHED_LINE	3	Пунктирная линия
	USERBIT_LINE	4	Тип линии, задаваемый пользователем
line_width	NORM_WIDTH	1	Нормальная толщина линии
	THICK_WIDTH	3	Жирная линия

Таблица Б.3 - Константы для задания способа выравнивания текста по горизонтали и вертикали в функции *settextjustify*

Константа	Значение	Семантика
horiz	LEFT_TEXT	0 По левому краю
	CENTER_TEXT	1 По центру (центрирование)
	RIGHT_TEXT	2 По правому краю
vert	BOTTOM_TEXT	0 По нижней линии
	CENTER_TEXT	1 По центру (центрирование)
	TOP_TEXT	2 По верхней линии

Таблица Б.4 - Константы для задания типа шрифта текста в функциях *settextstyle* и *gettextsetting*

Константа	Значение	Семантика
font	DEFAULT_FONT	0 Латинский (кириллица): по умолчанию
	TRIPLEX_FONT	1 Тройной
	SMALL_FONT	2 Малый
	SANS_SERIF_FONT	3 Санскрит
	GOTHIC_FONT	4 Готический
direction	HORIZ_DIR	0 Слева направо (по горизонтали)
	VERT_DIR	1 Сверху вниз (по вертикали)
charsize	целое	* 0 Относительный размер символа
	0	Выбор размера по умолчанию или размер, указанный в предыдущем вызове <i>setusercharsize</i>

Таблица Б.5 - Логические операции, выполняемые над изображением в функциях *putimage* и *setwritemode* (значение переменной *op*)

Константа	Семантика
COPY_PUT	Операция переноса (MOVE)
XOR_PUT	Операция «Исключающее или» (XOR)
OR_PUT	Операция «Или» (OR)
AND_PUT	Операция «И» (AND)
NOT_PUT	Операция «Не» (NOT)

СОДЕРЖАНИЕ

Лабораторная работа N 7. Рекурсия в языке Си	3
Лабораторная работа N 8. Работа с файлами последовательного доступа на языке Си	25
Лабораторная работа N 9. Графические средства Turbo-C.....	55
Лабораторная работа N 10. Работа с динамическими структу- рами данных на языке Си.....	87
Лабораторная работа N 11. Работа с файлами прямого доступа на языке Си	101
Список рекомендуемой литературы	132
Приложение А. График выполнения лабораторных работ	133
Приложение Б. Основные именованные константы графической библиотеки Turbo-C.....	134