Министерство образования и науки Республики Казахстан

Павлодарский государственный университет им. С. Торайгырова

Кафедра Вычислительная техника и программирование

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к лабораторным работам

по дисциплине «Технология программирования»

для студентов специальности 050704 – Вычислительная техника и программное обеспечение

Павлодар

УТВЕРЖДАЮ
Декан факультета ФМиИТ
_____ С. К. Тлеукенов
«___» _____ 200 г.

Составитель: доцент _____Фандюшин В. И..

Кафедра Вычислительная техника и программирование

Методические указания

к лабораторным работам

по дисциплине Технология программирования

для студентов специальности 050704 Вычислительная техника и программное обеспечение

Рекомендовано на заседании кафедры

«__»_____ 200 г., протокол №

Заведующий кафедрой _____ Потапенко О.Г.

Одобрено МС факультета ФМиИТ

«__»_____ 200 г., протокол №

Председатель МС _____ А. Т. Кишубаева

Содержание

1	Лабораторная работа № 1 «Операторы ввода и вывода в языке программирования Си++»	5
2	Лабораторная работа № 2 «Условные и безусловные операторы в языке программирования Си++»	10
3	Лабораторная работа № 3 « Операторы цикла в языке программирования Си++»	17
4 Лабораторная работа № 4 « Массивы и указатели в	25	

языке программирования Си++»		
5	Лабораторная работа № 5 «Работа с функциями в языке программирования Си++»	29
6	Лабораторная работа № 6 «Работа со структурами в языке программирования Си++»	34
7	Лабораторная работа № 7 «Препроцессорные средства в языке программирования Си++»	39

Лабораторная работа №1

«Операторы ввода и вывода в языке программирования Си++»

Цель работы: Изучить операторы ввода и вывода, форматы, используемые в этих операторах.

1.ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1Ввод и вывод информации

Вначале рассмотрим функцию, определяющую форматный вывод:

printf("управляющая строка", аргумент1, аргумент2, ...);

Например, в результате работы программы получены переменная i, имеющая значение 100, и переменная j, имеющая значение 25. Обе переменные целого типа. Для вывода этих переменных на экран в виде

i=100 j=25

необходимо применить функцию

printf(“i=%d j=%d”,i,j);

Как было описано выше, в кавычках задается формат вывода. перед знаком % записываются символы, которые будут непосредственно выданы на экран. После знака % применена спецификация d, т.к. переменные i и j имеют целый тип. Сами i и j приведены через запятую в списке аргументов. Если результат должен быть представлен в виде

i=100; j=25

необходимо применить функцию

printf(“i=%d; j=%d, i, j);

Если после знака % стоит цифра, то она задает поле, в котором будет выполнен вывод числа. Приведем несколько функций printf, которые будут обеспечивать вывод одной и той же переменной S целого типа, имеющей значение 336.

Функция printf(“%2d”, S);

выдает на экран:

336

В этом примере ширина поля (она равна двум) меньше, чем число цифр в числе 336, поэтому поле автоматически расширяется до необходимого размера.

Функция printf(“%6d”, S);

выдаст на экран:

_ _ _336

(6 позиций)

То есть, в результате работы функции число сдвинуто к правому краю поля, а лишние позиции перед числом заполнены пробелами.

Функция printf(“%-6d”, S);

выдаст на экран:

336_ _ _

(6 позиций)

Знак «минус» перед спецификацией приводит к сдвигу числа к левому краю поля.

Рассмотрим вывод вещественных чисел.

Если перед спецификацией f ничего не указано, то выводится число с шестью знаками после запятой. при печати числа с плавающей точкой перед спецификацией f тоже могут находиться цифры.

1.2Ввод данных

Для форматного ввода данных используется функция

scanf(«управляющая строка», аргумент1, аргумент2,...);

Если в качестве аргумента используется переменная, то перед ее именем записывается символ &.

Управляющая строка содержит спецификации преобразования и используется для установления количества и типов аргументов. спецификации для определения типов аргументов такие же, как и для функции printf. Перед символами d,o,x,f может стоять буква l. В первых трех случаях соответствующие переменные должны иметь тип long, а в последнем double.

Рассмотрим пример. Требуется ввести значения для переменных i (целого типа) и a (вещественного типа). Эту задачу выполнит функция:

scanf(“%d%f”,&i,&a);

1.3Операторы и выражения

Выражения широко используются в программах на языке СИ и представляют собой формулы для вычисления переменных. Они состоят из операндов (переменные, константы и др.), соединенных знаками операций (сложение, вычитание, умножение и др.). Порядок выполнения при вычислении значения выражения определяется их приоритетами и может регулироваться с помощью круглых скобок. Наиболее часто арифметические выражения используются в операторе присваивания. Этот оператор заменяет значение переменной в левой части оператора на значение выражения, стоящего в правой части, и имеет следующую форму:

переменная = выражение;

В языке СИ может быть использован модификатор const, запрещающий какие бы то ни было переопределения константы: ее уменьшение, увеличение и т.п. Модификатор const, используемый отдельно, эквивалентен const int. Приведем примеры:

const float a=3.5;

const j=47;

В таблице 1 приведены арифметические операции, используемые в языке СИ.

Таблица 1- Арифметические операции

Знак операции	Выполнение действия
+	Сложение
–	Вычитание
*	Умножение
/	Деление
%	Деление по модулю

Результатом деления по модулю является остаток от деления. Например, если b=5, c=2, то при выполнении операции

a=b%c ,

переменная a получит значение 1.

В программах на языке СИ важная роль отводится комментариям, которые повышают наглядность и удобство чтения программ. Они могут быть записаны в любом месте программы и обрамляются символами /* и */.

Рассмотрим пример программы на языке СИ.

$y = \frac{a+b}{c}$, где $y = \frac{a+b}{c}$; $b = \sin x + tgx$ $c = \sqrt[5]{d+e^x}$

Для работы с математическими функциями необходимо перед функцией main поместить строку:

#include <math.h>

Программа на СИ имеет вид:

```
#include <stdio.h>
#include <math.h>
main()
{
float z,f,k; /*объявление вещественных переменных z,f,k*/
double y,a,b,c,d,x; /*объявление переменных y,a,b,c,d,x переменными двойной точности*/
scanf("%f %f %f %lf %lf", &z, &f, &k, &d, &x); /* ввод с клавиатуры переменных z,f,k,d,x*/
a=log(x)+(z+f)/k;
```

```
b=sin(x)+tan(x);
c=pow(d+exp(x),1./5);
y=(a+b)/c;
printf("%lf %lf %ef %lf", a, b, c, y); /*вывод на экран значений переменных a,b,c,y*/
}
```

Следует обратить внимание на то, что при вычислении переменной c, выражение, стоящее в правой части, представлено как $\sqrt[5]{(d+e^x)}$, поэтому применена функция pow. Еще одно замечание. Следует осторожно подходить к делению целых чисел. Если оба операнда целые, то результат тоже будет целым, а дробная часть отбрасывается. таким образом, при выполнении операции 1/5, результат будет равен нулю. Для того чтобы сохранить дробную часть, хотя бы один из операндов должен быть вещественным. Это условие выполнено при вычислении 1./5 .

Задание

Из таблицы 3 взять задание по варианту и написать программу для вычисления выражения на языке СИ++.

Содержание отчёта

Отчёт должен содержать:

1. задание к работе;
2. программу.
3. результаты работы программы.

Контрольные вопросы

1. Какова структура программы на С++?
2. Как задать имя переменной?
3. Как ввести данные в программу?
4. Как вывести результаты расчета?
5. Как осуществить форматный вывод результата?

Таблица 3- Задания по вариантам

№ варианта	Задание	Исходные данные
1 n = 10,2 2 x = 4,3 3 i = 5 4 m = 2	$r = \frac{d + b}{c}$, где $d = \sqrt[3]{x^2 + y^2}$; $b = \lvert i - j^3 \rvert$; $c = \sin^2(x + y) - \ln x$	x = 4,5 y = 8,5 i = 3 j = 6

$c = e^{2x} + \sqrt[3]{n^2 + 6}$

5 6	$q = \frac{c - b}{a}$, где $d = tg^2 \frac{m}{m+3} \times w$ $z = \frac{a\,d}{c - d}$, где $a = \sin^3(2m)$ $b = \lvert x^2 + y \rvert$ $c = \sqrt[4]{e^{3m} + \ln w}$ $d = \lvert i - m \rvert$	m = 8,1 w = 4,2 x = 10 y = 4
7	$s = \frac{x}{a + y} + \ln^2 z$, где $x = \cos^2(t + p) - e^{2t}$ $y = \sqrt[3]{\ln p + 10t}$ $a = \lvert k + m^2 \rvert$	t = 4,7 z = 0,8 p = 6,2 k = – 4 m = 6
8	$m = \frac{x^2 + y}{d}$, где $x = \sqrt[3]{e^{2t} + 10p}$ $y = tg^2 \frac{tw}{10}$ $d = \lvert j^3 - i^2 \rvert$	t = 4,1 p = 3,2 w = 8,7 j = 3 i = 7
9	$p = \frac{t - q}{f^2}$, где $t = \ln x^2 + \sin^2 y$ $q = \sqrt[4]{e^{2x} + 10y}$ $f = \lvert k^2 + m^3 \rvert$	x = 5,7 y = 1,9 k = 8 m = 2
10	$z = \ln^2 p - \frac{x - y^2}{t}$, где $x = \sin e^{2m}$ $y = \sqrt[5]{10n + m^3}$ $t = \left\lvert \frac{i^2 \cdot j^2}{j} \right\rvert$	m = 5,6 n = 9,4 i = 4 j = 9
11	$s = \frac{x^2 + y^2}{t}$, где $x = \sqrt[3]{e^{2q} + q^4}$ $y = tg^2 q - 4p$ $t = \left\lvert \frac{i^2 + 6}{j} \right\rvert$	q = 1,7 p = 2,3 i = 9 j = 4
12	$r = \frac{x \cdot a}{y} + t \cdot \ln^2 m$, где $x = \sqrt[4]{e^{2n} + 4m}$ $y = \sin^2(n + m)$ $a = \left\lvert \frac{k + p^2}{p} \right\rvert$	n = 2,6 m = 3,7 p = 6 k = 2
13	$s = \frac{x + y}{m + 5}$, где $x = \sqrt[3]{e^{2t} + t^2}$ $y = \sin(p^2 + t)$ $m = \left\lvert \frac{k^2 - i}{i^2} \right\rvert$	t = 1,1 p = 2,4 k = 4 i = 5
14	$z = \frac{t}{p} + f$, где $t = \log^2(x + y)$ $p = \sqrt[5]{e^{x+y} + 10y}$ $f = \left\lvert \frac{i + j}{j^2} \right\rvert$	x = 3,7 y = 2,1 i = 3 j = 4

15	$h = \sqrt[4]{t \times e^{2t}}$ $q = \frac{h}{c + d}$, где $c = tg^2(t+p) + \sin p^2$ $d = \left\lvert \frac{1}{m^2 + 1} \right\rvert$	t = 1,5 p = 4,8 i = 2 m = 6
16	$s = \frac{q + p}{p} + tg^2 z$, где $q = \sqrt[5]{e^{2m} + m^2}$ $p = \left\lvert \frac{n - j^2}{n^2 + j} \right\rvert$	z = 2,4 m = 5,8 n = 4 j = 5
17	$h = \sin(tg\, x^2)$ $r = \frac{h}{p - d}$, где $p = \sqrt[3]{e^{x^2} + y^3}$ $d = \left\lvert \frac{j}{i + j} \right\rvert$	x = 9,5 y = 3,6 j = 2 i = 5
18	$a = \cos^2(x+y)$ $d = \frac{a + b^2}{c}$, где $b = \sqrt[4]{x^2 + e^{x+y}}$ $c = \left\lvert \frac{k^2 + m}{m^2} \right\rvert$	x = 6,4 y = 1,7 k = 5 m = 7

Лабораторная работа №2

«Условные и безусловные операторы в языке программирования Си++»

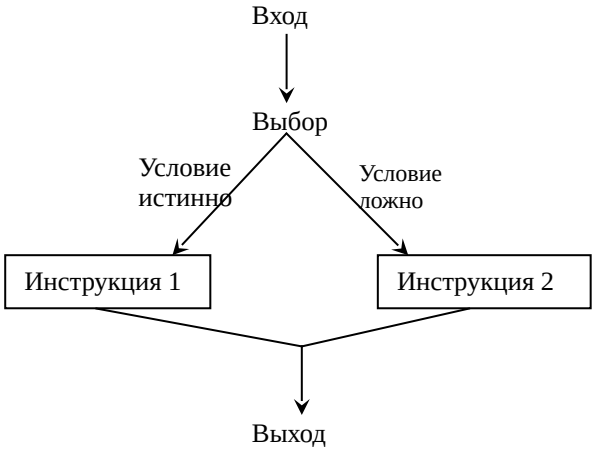
Цель работы: познакомиться с работой условного оператора и оператора перехода. Изучить оператор выбора варианта. Научиться применять их при составлении программ.

1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Условный оператор if имеет вид:

if (проверка условия) инструкция1; else инструкция2;

Если условие в скобках принимает истинное значение, выполняется



инструкция1, а если ложное – инструкция 2 (рисунок 1).

Рисунок 1 – Структурная схема условного логического оператора

Например:

```
if (a>b)
z=a;
else
z=b;
```

Необходимо обратить внимание на точку с запятой после z=a. Здесь она обязательна, поскольку за if должна следовать инструкция, которая всегда заканчивается точкой с запятой.

В операторе if слово else может отсутствовать. В этом случае, если условие в скобках принимает истинное значение, выполняется инструкция 1, а если ложное, то инструкция 1 пропускается и управление передается следующему оператору по тексту программы.

Например:

```
if (num>10) num=2*num;
printf(“%d\n”,num);
```

Оператор вывода будет выполняться всегда, а оператор присваивания только в том случае, если условие будет истинным.

Рассмотрим простейшую программу:

```
#include <stdio.h>
main( )
{
int a,b;
puts(“Введите значения a и b”);
scanf(“%d %d”, &a,&b);
if (a>b) puts(“a больше b”);
else puts(“”); /*демонстрация оператора if – else*/
if (a==b) puts(“”); /*демонстрация оператора if без слова else*/
}
```

Если для выполнения программы ввести числа 5 и 3, то на экране появится строка:

a больше b

При введении чисел 5 и 5, на экране появится две строки:

a меньше или равно b

a равно b

Оператор ***безусловного перехода*** можно представить в следующей форме:

goto метка;

Метка – это любой идентификатор.

Например:

goto a2;

Оператор goto указывает, что выполнение программы необходимо продолжить, начиная с инструкции, перед которой записана метка. В программе обязательно должна быть строка, где указана метка, поставлено двоеточие и записана инструкция, к которой должен выполняться переход.

Например:

a2: k=5;

Метки в программе описывать не нужно. Применение оператора безусловного перехода в языке СИ является нежелательным, так как он нарушает структурную наглядность программы.

Оператор выбора **switch** позволяет выбрать одну из нескольких альтернатив. Он записывается в следующем виде:

```
switch (выражение)
{case константа1, вариант 1; break;
  . . .
  case константа n, вариант n; break;
default: вариант n+1; break;}
```

В операторе switch вычисляется целое выражение в скобках (его называют селектором), и его значение сравнивается со всеми константами. При совпадении выполняется соответствующий вариант (одна или несколько инструкций). Все константы в записи оператора должны быть различными. Вариант с ключевым словом default (прочие) реализуется, если ни один другой не подошел (если слово default отсутствует, а все результаты сравнения отрицательны, то ни один вариант не выполняется. Для прекращения последующих проверок после успешного выбора некоторого варианта используется оператор break, обеспечивающий немедленный выход из оператора switch.

Например:

```
#include<stdio.h>
main( )
{
char y;
scanf(“%c”,&y);
switch(y)
{
case ‘1’:
```

```
printf(“Ветвь 1\n”);
break;
case ‘2’:
case ‘3’:
printf(“Ветвь 2 или 3\n”);
break;
default:
printf(“Ветви 1,2,3 не работают\n”);
}
}
```

Оператор scanf вводит переменную y. Ее значение в операторе switch сравнивается со всеми константами операторов case. Если ввести символ ‘1’, то на экране появится строка:

Ветвь 1

по оператору break произойдет выход из переключателя switch, и программа завершит свою работу. Если ввести символы ‘2’ или ‘3’, то на экран будет выведена строка:

Ветвь 2 или 3

При вводе любого другого символа управление перейдет к ключевому слову default и на экране появится строка:

Ветви 1,2,3 не работают.

Задание

1. Из таблицы 1 взять задание по варианту и написать программу, используя оператор условного перехода.
2. Из таблицы 2 взять задание по варианту и написать программу, используя оператор выбора. Для выбора четвертой ветви использовать вариант с ключевым словом default.

Содержание отчёта

Отчёт должен содержать:

1. задание к работе;
2. программу.
3. результаты работы программы.

Контрольные вопросы

1. Что такое ветвление?
2. Каким оператором реализуется ветвление?
3. Как записать сложное условие?
4. Как работает оператор выбора?

Таблица 1

№ варианта	Содержание		Исходные данные
1.	$Z = \begin{cases} a + bx - cx^2 & k = 1,2,3 \\ d + ex + fx^2 & k = 4,5 \\ ab + fx + cx^2 & k = 8 \end{cases}$		a=2 b=1,5 c=1 d=3 e=0,5
2.	$Z = \begin{cases} I + c\sqrt{d} & n = 0 \\ x - a & n = 1,6 \\ 2/3x^2 - 1/2d & n = 2,3,4 \end{cases}$		c=-2 a=1,5 d=2 x=3
3. $Z = \begin{cases} y^2 + 0,3a \\ a + e^{yb} \\ y^2 + y - b \end{cases}$ $x = 3$ $x = 5,2$ $x = 6,7,8$	a=5 b=1,2 y= 0,6		
4.	$Z = \begin{cases} 1 + d \sin d / a & j = 1 \\ (i - 1)i + a^2 & j = 2,4,8 \\ i + 2/3x & j = 5,6,7 \end{cases}$		d=2 a=3,5 x=3 i=2,3
5.	$Z = \begin{cases} \omega/3 + a^2 x & c = 3 \\ \omega - \ln b & c = 8,9,10 \\ b^2 + \omega x & c = 2,5,7 \end{cases}$		□=1 x=2,5 a=4 b=0,4
6.	$Z = \begin{cases} ab + arctga^2 & n = 0,1,6 \\ ah/2 & n = 7 \\ \pi R^2 & n = 2,3,4 \end{cases}$		a=0,5 b=2 h=4 R=1,4
7.	$Z = \begin{cases} p \cdot \ell & k = 3,4,5 \\ ph/2 + \cos P & k = 1,2,8, \\ \pi R\ell & k = 9 \end{cases}$		p=0,6 l=2 h=5 R=4
8.	$Z = \begin{cases} \sin x + cd & b = 1,2 \\ x/a + \sqrt{da2} & b = 3,4,5 \\ a + d \cos x & b = 8 \end{cases}$		x=0,63 c=1,5 d=2 a=0,37
9.	$Z = \begin{cases} y + (x - a)/(x + a) & c = 0,1,2 \\ y - x & c = 4,5 \\ y^2 + e^{ax} & c = 3 \end{cases}$		y=2,6 x=1,6 a=0,4
10.	$Z = \begin{cases} a + 2/b + 4\omega & \omega = 4,5,6 \\ (a + b)^2 & \omega = 2,3,7 \\ a - x\omega & \omega = 8 \end{cases}$		a=1,5 b=2 x=0,5

№ варианта	Содержание		Исходные данные
11.	$Z = \begin{cases} 1,5x + 9x^2 - 1,25 \\ d - a\sin x \\ a - x + y/d \end{cases}$	$n = 2,3,4$ $n = 5,6,8$ $n = 1$	x=0,5 a=4,3 y=2,6 d=0,3
12.	$Z = \begin{cases} 0,5a + \cos y/a \\ y^2 - i \\ x + x^2/2 - 1/3 \end{cases}$	$m = 0,1,2$ $m = 3,5,7$ $m = 4$	a=3 y=2,7 i=2 x=1
13.	$Z = \begin{cases} \sin x + e^x \\ (X + y)/(1 - xy) \\ x + t^2 \end{cases}$	$= 5,9$ $= 1,2,3$ $= 0$	x=0,73 y=0,4 t=2,6
14.	$Z = \begin{cases} e^{a\sin x} + e^x \\ \sqrt{\lvert a + ax\rvert} \\ \pi R^2 \end{cases}$	$d = 2$ $d = 3,4,5$ $d = 6,8$	a=2 x=0,54 c=2,3 b=1,8 R=3
15.	$Z = \begin{cases} 1 - \sin x \\ 1/2(1 + \cos a) \\ \sqrt{x + c} \end{cases}$	$t = 8$ $t = 0,1,2,3$ $t = 4,6,7$	x=0,4 a=0,88 c=3,6
16.	$Z = \begin{cases} (a \cdot \sin ix)^2 \\ \sqrt{\lvert a + bx\rvert} \\ ia\ell n\lvert c\rvert \end{cases}$	$i = 3,4,5,6$ $i = 7,9$ $i = 0$	a= -1 b=0,8 x=1 c= -0,7
17.	$Z = \begin{cases} -xy \\ ab/xy \\ (a + b)/a^2 \end{cases}$	$p = 3,4,5$ $p = 6$ $p = 8,10$	x=2 y=3,5 a=0,1 b=4
18.	$Z = \begin{cases} arctg(x + y)/(-xy) \\ e^x \\ a + bx + t \end{cases}$	$k = 5,5$ $k = 0,1,2$ $k = 3$	x=0,8 y=0,2 a=4 b=5 t=1
19.	$Z = \begin{cases} y^2 - 0,3 + a \\ 0 \\ (y + x^2)/2ab \end{cases}$	$c = 0,3$ $c = 5,6,7,8$ $c = 1$	y=2 a=0,5 x=1,8 b=0,6
20.	$Z = \begin{cases} ((x + y)/t)^2 \\ 1/a^2(\lambda/10) \\ \sin ab \end{cases}$	$n = 2,3,4,5$ $n = 6,8$ $n = 0,7$	x=1,8 y=2 t=4 a=1,5 R=4,6 b=0,3

Таблица 2

№ варианта	Содержание	Исходные данные
1	$v=\begin{cases} a+2/b+4w & w=4 \\ a-xw & w=8 \\ (a+b)^2 & w=2 \\ нет\_решения\_w=4,8,2 \end{cases}$	$a=1.5$ $b=2$ $x=0.5$
2	$v=\begin{cases} 1.5x+9x^2+1.25\_n=2 \\ d+a\sin x & n=5 \\ (a-x+y/d & n=1 \\ нет\_решения & n=1,5,2 \end{cases}$	$x=0.5$ $a=4.3$ $y=2.6$ $d=0.3$
3	$v=\begin{cases} 0.5a+\cos a/y\_m=1 \\ y^2-i & m=5 \\ x+x^2/2-1/3 & m=7 \\ нет\_решения & m=1,5,7 \end{cases}$	$a=3$ $y=2.7$ $i=2$ $x=1$
4	$v=\begin{cases} \sin x+e & k=3 \\ (x+y)/(1-xy) & k=1 \\ x+t^2 & k=0 \\ нет\_решения\_k=0,1,3 \end{cases}$ $x=0.73$ $y=0.4$ $t=2.6$	
5	$v=\begin{cases} a\sin x+c & d=2 \\ R^2 & d=6 \\ /a+bx/ & d=3 \\ нет\_решения\_d=2,3,6 \end{cases}$	$a=2$ $x=0.54$ $c=2.3$ $b=1.8$ $r=3$
6	$v=\begin{cases} 1-\cos x & t=1 \\ i/2(i+\sin a) & t=2 \\ \sqrt{x+c} & t=6 \\ нет\_решения\_t=1,2,6 \end{cases}$	$x=0.4$ $a=0.81$ $c=2.6$

№ варианта	Содержание	Исходные данные
7	$v=\begin{cases} a+bx+cx^2 & k=3 \\ d+x+fx^2 & k=4 \\ ab+fx+cx^2 & k=6 \\ нет\_решения\_ & d=3,4,6 \end{cases}$	$a=2$ $b=1.5$ $c=1.2$ $d=3$ $f=0.5$
8	$v=\begin{cases} 1+c\sqrt{d} & n=0 \\ x-a & n=6 \\ z/3x^2+1/2d & n=3 \\ нет\_решения\_ & n=0,3,6 \end{cases}$	$c=-2$ $d=1.5$ $x=3$ $d=2$
9	$v=\begin{cases} a+e & x=8 \\ y^2+y-b & x=9 \\ y^2+0.3d & x=11 \\ нет\_решения\_ & x=8,9,11 \end{cases}$	$a=3$ $y=0.9$ $b=1.2$
10	$v=\begin{cases} 1+d\sin d & j=1 \\ (i-1)i+a^2 & j=4 \\ i+2/3x & j=5 \\ нет\_решения\_ & j=1,4,5 \end{cases}$	$a=3.5$ $d=2$ $x=3$ $d=2.2$
11	$v=\begin{cases} w/3+a^2x & c=3 \\ w-1nb & c=4 \\ b^2+wx & c=8 \\ нет\_решения\_ & c=3,4,8 \end{cases}$	$w=1$ $x=1.5$ $a=3$ $b=0.8$
12	$v=\begin{cases} ab+arctga^2 & n=0 \\ ah/2 & n=3 \\ \Pi R^2 & n=4 \\ нет\_решения\_ & n=0,3,4 \end{cases}$	$b=2$ $a=0.6$ $b=4$ $R=1.8$
13	$v=\begin{cases} pl & K=3 \\ ph/2+\cos p & k=2 \\ \Pi Rl & k=5 \\ нет\_решения\_ & k=3,2,5 \end{cases}$	$p=0.7$ $l=2$ $R=4$ $h=4.8$
14	$v=\begin{cases} \sin x+cd & b=2 \\ \sqrt{d}a+x/a^2 & b=4 \\ a+d\cos x & b=6 \\ нет\_решения\_ & b=2,4,6 \end{cases}$	$x=0.66$ $c=1.5$ $a=0.37$ $d=2$

№ варианта	Содержание	Исходные данные
15	$v=\begin{cases} y+(x-a)/(x+a)\_ & c=0 \\ y-x & c=2 \\ y^2+e^{ax} & c=3 \\ нет\_решения\_ & c=0,2,3 \end{cases}$	$y=2.6$ $x=1.61$ $a=0.4$
16	$v=\begin{cases} (a-\sin xi)^2 & i=3 \\ /a+bx/ & i=7 \\ i*a*\ln/c/ & i=8 \\ нет\_решения\_ & i=3,7,8 \end{cases}$	$a=-1$ $b=0.8$ $x=1$ $c=-0.7$
17	$v=\begin{cases} -xy & p=3 \\ ab/x & p=4 \\ (a+b)/a^2 & p=8 \\ нет\_решения\_ & p=3,4,8 \end{cases}$	$x=2$ $y=3.5$ $a=0.1$ $b=4$
18	$v=\begin{cases} arctg(x+y)/xy & k=5 \\ e^x & k=0 \\ a+bx+t & k=3 \\ нет\_решения\_ & k=0,3,5 \end{cases}$	$x=0.3$ $y=0.1$ $a=4$ $b=5.2$ $t=1.2$
19	$v=\begin{cases} 0 & c=3 \\ y^2+0.3a & c=5 \\ (y+x^2)/2ab & c=2 \\ нет\_решения\_ & c=2,3,5 \end{cases}$	$y=2$ $a=0.6$ $x=1.8$ $b=0.66$
20	$v=\begin{cases} x+y/t^2 & n=2 \\ ia^2*(R/10) & n=6 \\ \sin ab & n=7 \\ нет\_решения\_ & n=2,6,7 \end{cases}$	$y=2$ $x=1.8$ $t=4$ $a=1.5$ $R=4.6$

Лабораторная работа №3

« Операторы цикла в языке программирования Си++»

Цель работы: ознакомиться с циклическими алгоритмами и операторами, реализующими эти алгоритмы. Освоить особенности применение каждого оператора. Составить программы с использованием всех операторов цикла.

1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1. Оператор цикла while

Описание:
while (выражение) оператор;

Оператор иногда называется телом цикла. В теле цикла должны выполняться действия, в результате которых меняется значение управляющего выражения. В противном случае можем получить бесконечный цикл.
Пример: Вычислить сумму дести натуральных чисел.

```
/*Демонстрация цикла while*/
#include <stdio.h>
main( )
{
int i=1, s=0;
while (i<=10)
{
s+=i; i++;
}
printf(“Сумма   s=%d”,s);
}
```

1.2. Оператор цикла do-while

Описание:
do оператор while (выражение);
Действие:

В операторе do-while тело цикла выполняется по крайней мере один раз. Тело цикла будет выполняться до тех пор, пока выражение в скобках не примет ложное значение. Если оно ложно при входе в цикл, то его тело выполняется ровно один раз.
Пример: Вычислить сумму дести натуральных чисел.

```
/*Демонстрация цикла do-while */
#include <stdio.h>
main()
{
int i=1,s=0;
do
s+=i;
i++;
while (i<=10);
printf(“Сумма s=%d-й”,s);
}
```

Программа, представленная выше, теперь написана с циклом do-while. Результат программы будет таким же.

1.3. Оператор цикла for

Описание:

for (выражение 1; выражение 2; выражение 3) оператор;

Действие:

В круглых скобках содержится три выражения. Первое из них служит для инициализации счетчика. Она осуществляется только один раз – когда цикл for начинает выполняться. Второе выражение необходимо для проверки условия, которая осуществляется перед каждым возможным выполнением тела цикла. Когда выражение становится ложным, цикл завершается. Третье выражение вычисляется в конце каждого выполнения тела цикла, происходит приращение числа на шаг.

```
/*Демонстрация цикла for*/
#include <stdio.h>
main( )
{
int i, s=0;
for(i=1;i<=10;i++)
s+=i;
printf("Сумма   s=%d",s);
}
```

1.4 Оператор break

Описание:

Break используется для прекращения выполнения цикла из-за обнаружения ошибки, для организации дополнения к условию в заголовке цикла, для прекращения бесконечного цикла.

Пример:

```
while (st>0 && st<25)
  {
    if st==4||st==8||st==12)
        break;
  }
```

Работа цикла полностью прекращается, как только условие в операторе if становится истинным.

1.5Оператор continue

Описание: continue

Действие:

Этот оператор может использоваться во всех трех типах циклов. Как и в случае оператора break, он приводит к изменению характера выполнения программы. Однако вместо завершения работы цикла наличие оператора continue вызывает пропуск "оставшейся" части итерации и переход к началу следующей.

Пример. Заменим в предыдущей программе оператор break на continue.

```
while (st>0 && st<25)
  {
     if (st==4||st==8||st==12)
         continue;
```

}

При истинном условии в операторе if оператор continue вызывает пропуск идущих за ним операторов тела цикла и осуществляется переход к началу следующей итерации.

Задание

1. Задание взять из таблицы 1 и таблицы 2 согласно варианту.
2. Разработать блок-схемы алгоритма.
3. Написать и отладить программы.

Содержание отчёта

Отчёт должен содержать:

1. задание к работе;
2. программу.
3. результаты работы программы.

Контрольные вопросы

1. Что такое циклический процесс?
2. Каким оператором реализуется цикл с предусловием?
3. Каким оператором реализуется цикл с постусловием?
4. Как работает оператор цикла for?
5. Как работают вложенные циклы?
6. Как применяются операторы break и continue?

Таблица 1

№ варианта	Задание
1.	$y = a^3\left[\sum_{i=1}^{3}(i+5) + b^2\prod_{j=1}^{4}(j+d)\right]$
2.	$y = n\left[\frac{a}{\prod_{j=1}^{3}(j+d)} + \frac{\sum_{i=1}^{3}(i+m)}{m}\right]$
3. $y = m\left[k^2\right]$	

№ варианта	Задание
4.	$y = a\left[\frac{1}{b\prod_{k=1}^{3}(k+2)} + \sum_{i=1}^{3}(i+4)\right]$
5.	$y = \frac{n^2\prod_{j=1}^{4}(j+4)}{d^3} + \sum_{i=1}^{8}(i+a)$
6.	$y = m\left[\frac{\sum_{i=1}^{4}(i+4)+d}{\prod_{j=1}^{3}(j+n)}\right]$
7.	$y = \frac{m\prod_{j=1}^{3}(j+b)+d}{a^3\sum_{i=1}^{3}(i+4)}$
8.	$y = a\prod_{j=1}^{3}(b+j) + \sum_{i=1}^{3}(i+b\bullet i)$
9.	$y = \frac{a^2}{\sum_{i=1}^{4}(i+b)} + c\prod_{j=1}^{3}(j+2)$
10.	$y = a^2\left[\prod_{j=1}^{4}(j+d) + t\bullet\sum_{m=1}^{3}(m+2)\right]$
11.	$y = \frac{1}{b\prod_{j=1}^{3}(j+5)} + \frac{\sum(i+a)}{a^2}$
12.	$y = \frac{c+\sum_{i=1}^{3}(i+d)}{\prod_{j=1}^{3}\left[\frac{1}{j+2}\right]}$
13.	$y = \frac{d\prod_{j=1}^{3}(i+a^3)}{t^2\sum_{i=1}^{3}(i+3)}$
14.	$y = a\left[\sum_{k=1}^{3}(k+b^2) + \prod_{j=1}^{3}(j\bullet c+2)\right]$
15.	$y = \frac{a\sum_{j=1}^{3}(j+5/j)}{b^2\prod_{i=1}^{3}(i+6i)}$

№ варианта	Задание
16.	$y=\frac{\prod_{k=1}^{4}(k+d)+m}{\sum_{i=1}^{3}(i+5)+a^2}$
17.	$y=a\left[b^2\sum_{k=1}^{3}(k+4)+\prod_{j=1}^{4}(j^2+1)\right]$
18.	$y=a\left[\frac{\prod_{i=1}^{3}(k+bk)}{\sum_{j=1}^{3}(j+\frac{b}{j})}\right]$
19.	$y=\frac{\sum_{i=1}^{3}(i+2)+a\prod_{j=1}^{4}(j+b^2)}{a+b^2}$
20. $y=\frac{a\sum_{i=1}^{3}(i}{b^2}$	
21.	$y=a\left[\frac{b^2}{\prod_{j=1}^{3}(j+3)}+\sum_{i=1}^{3}(i+3)\right]$
22.	$y=\frac{m\sum_{k=1}^{3}(k^2+b^2)}{\prod_{j=1}^{3}(j+tj^2)}$
23	$y=m\left[\frac{b\sum_{i=1}^{3}(i+a)}{\prod_{j=1}^{3}(j+1/b)}\right]$
24.	$y=\frac{1}{\prod_{j=1}^{3}(j+4)}+f\sum_{i=1}^{3}(5+i)$
25.	$y=\frac{a^2+b\sum_{i=1}^{3}(i+5)}{4+\prod_{j=1}^{3}(j+2)}$

Таблица 2

№ варианта	Задание
1.	$y=\prod_{j=1}^{3}(j\sum_{k=1}^{3}(j+k))$
2. $y=\sum_{i=1}^{3}\left[\frac{1}{i}\right]$	
3.	$y=\prod_{j=1}^{3}\frac{\sum_{k=1}^{3}(k+j)}{j^2}$
4.	$y=\prod_{i=1}^{3}\left[\frac{i+\sum_{j=1}^{3}(i+j)}{i^2}\right]$
5.	$y=\sum_{i=1}^{3}\left[i\bullet\prod_{i=1}^{3}(i+3j)\right]$
6.	$y=\sum_{i=1}^{3}\left[i-\frac{i}{\prod_{k=1}^{3}(k+2)}\right]$
7.	$y=\prod_{j=1}^{3}\left[2+a\sum_{i=1}^{3}(i+2j)\right]$
8.	$y=\prod_{i=1}^{3}\left[5+\frac{i}{4\sum_{j=1}^{3}(i+j)}\right]$
9.	$y=\sum_{j=1}^{4}\left[\frac{1+i}{\prod_{i=1}^{3}(i^2+2j)}\right]$
10.	$y=\prod_{j=1}^{3}\frac{a}{\sum_{i=1}^{3}(i^2+j^2)}$
11.	$y=\sum_{i=1}^{3}\left[i^2+\frac{c}{\prod_{j=1}^{3}(i+ja)}\right]$
12.	$y=\sum_{m=1}^{3}\left[\frac{1+m}{\prod_{j=1}^{3}(j+m)}\right]$

№ варианта	Задание
13.	$y = \frac{1}{a}\sum_{i=1}^{3}\frac{i}{\prod_{j=1}^{3}(j+\frac{i}{b})}$
14.	$y = \sum_{i=1}^{3}(i+\frac{1}{\prod_{j=1}^{3}(i+j2)})$
15.	$y = \prod_{i=1}^{3}\left[\frac{a^2+i^2}{\sum_{j=1}^{3}(j+i)}\right]$
16.	$y = \prod_{i=1}^{3}\left[\frac{a+\sum_{j=1}^{3}(i+j)}{a+b}\right]$
17.	$y = \sum_{i=1}^{3}\left[\frac{1}{\prod_{j=1}^{3}(i^2+j)}+i^2\right]$
18.	$y = \frac{1}{a}\prod_{j=1}^{3}\left[\frac{1}{b}+c\sum_{i=1}^{4}(i+j)\right]$
19. $y = \frac{1}{b}\sum_{j=1}^{4}\left[$	
20.	$y = \frac{1+c}{\prod_{j=1}^{4}\left[j+2\sum_{k=1}^{3}(k+j)\right]}$
21.	$y = \prod_{j=1}^{3}\left[\frac{j+1}{\sum_{i=1}^{3}(i^2+j^2)}\right]$
22.	$y = \prod_{i=1}^{3}\left[i^2+\frac{1+i}{\sum_{j=1}^{3}(i+j)}\right]$
23	$y = a\prod_{j=1}^{3}\left[\frac{a^2+\sum_{i-1}^{3}(i+j^2)}{b+a}\right]$
24.	$y = \sum_{i=1}^{3}\left[\frac{1}{i}+\frac{\prod_{j=1}^{3}(a+i+j)}{b^2}\right]$

№ варианта	Задание
25.	$y=\sum_{j=1}^{3}\frac{i+\prod_{j=1}^{3}(j^2+i)}{5i}$

Лабораторная работа №4

« Массивы и указатели в языке программирования Си++.»

Цель работы: ознакомиться с основными принципами работы с одномерными и двумернми массивами. Освоить работу с указателями и операциями над указателями.

1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1. Массивы

Массивы объявляются так же, как и переменные. Например:

```
int a[100];
float c[10][20];
```

В первой строке объявляем массив а из 100 элементов целого типа: a[0],a[1], … ,a[99] (индексация всегда начинается с нуля). Во второй строке объявлен двумерный массив вещественного типа. Двумерный массив представляется как одномерный, элементы которого являются тоже массивами. В первых квадратных скобках указывается количество строк в массиве, во вторых – количество столбцов.

Пример 1. Переставить местами элементы главной и побочной диагоналей массива D(6,6). Полученную матрицу вывести на экран дисплея.

```
#include <stdio.h>
main()
{int i,j,a,d[6][6];
for (i=0;i<6;i++)
for (j=0;j<6;j++)
scanf(“%d”, &d[i][j]); /*ввод матрицы*/
/*перестановка местами элементов главной и побочной диагоналей*/
for (i=0; i<5; i++)
{a=d[i][i];
d[i][i]=d[i][6-i];
d[i][6-i]=a;
}
for (i=0; i<6; i++)
for (j=0; j<6; j++)
printf(“%d%c”, d[i][j], (j==5)?”\n”:” “);
/*вывод по строкам элементов матрицы*/
```

}

Массив можно инициализировать, т.е. присвоить его элементам начальные значения. Это делается при объявлении типа массива, например:

int a[5]= { 0, 0, 0, 0, 0};

Это значит, что все элементы массива получают нулевое значение.

Двумерный массив можно инициализировать следующим образом:

```
int a[3][3] = {{10,20,30},
               {40,50,60},
               {70,80,90}};
```

При инициализации число элементов можно не указывать, т.к. в этом случае оно может быть вычислено по количеству присваиваемых значений (в фигурных скобках), например:

int a[] = {10,20,30,40,50};

1.2. Указатели

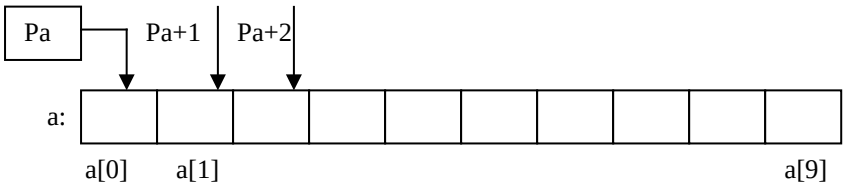
Указатель – это переменная, которая содержит адрес переменной. Так как указатель – это адрес некоторого объекта, то через него можно обращаться к данному объекту.

В СИ существует тесная связь между указателями и массивами. Любой доступ к элементу массива, осуществляемый операцией индексирования, может быть выполнен при помощи указателя.

Декларация

int a[10];

определяет массив a размера 10, т.е. блок из десяти последовательных объектов, представленных на рисунке, с именами a[0], a[1], … ,a[9].



Запись a[i] отсылает нас к i-му элементу массива. Если pa есть указатель, т.е. определен как

int *pa; , то в результате присваивания

pa = &a[0];

pa будет указывать на нулевой элемент массива a; иначе говоря, pa будет содержать адрес элемента a[0] (см. рис.). Теперь присваивание

x=*pa;

будет копировать содержимое a[0] в x.

Пример 2. Вывести значения одномерного массива обычным способом и с использованием указателей.

```
#include <stdio.h>
int a[6]={10,20,30,40,50,60};
/*объявление и инициализация массива*/
main ()
{int i, *p;
```

```
for (i=0; i<6; i++)
printf(“%d”,a[i]); /*вывод массива обычным способом*/
for (p=&a[0];p<=&a[5];p++)
printf(“%d”,*p); /*вывод массива с использованием указателя*/
for (p=&a[0],i=0; i<6; i++)
printf(“%d”,p[i]); /*еще один вариант с использованием указателя*/
}
```

Пример 3. Задана матрица a(3,3). Вывести на экран элементы главной диагонали, первой строки и значений первых элементов каждой строки матрицы, применив для этого указатели.

```
#include <stdio.h>
int a[3][3]={{10,20,30},
        {40,50,60},
        {70,80,90}};
        /*объявление и инициализация двумерного массива*/
int *pa[3]={a[0],a[1],a[2]};
```

/*объявление и инициализация указателя ра на строки массива а и присвоение начальных значений : ра[0]=a[0]; ра[1]=a[1]; ра[2]=a[2]*/

```
int p=a[0]; /*объявление указателя на нулевой элемент нулевой строки массива а*/
main ()
{int i;
for (i=0;i<9;i+=4)
printf(“%d”,*(p+i)); /*вывод на экран элементов главной диагонали*/
for (i=0; i<3; i++)
printf(“%d”,*p[i]); /*вывод на экран элементов первой строки*/
for (i=0; i<3; i++)
printf(“%d”,pa[i]); /*вывод на экран первых элементов каждой строки матрицы*/
}
```

Сделаем некоторые пояснения для первого оператора цикла. Представим матрицу в виде одномерного массива, записанного по строкам:

a[0][0],a[0][1],a[0][2],a[1][0],a[1][1],a[1][2],a[2][0],a[2][1],a[2][2]

Тогда элементы, стоящие на главной диагонали, занимают нулевое, четвертое и восьмое места, т.е. интервал между интересующими нас элементами равен четырем, поэтому переменная i изменяется с шагом 4. Соответственно с таким же шагом меняются адреса ячеек, содержимое которых выводится на экран.

Задание

Задание взять из таблицы 1 согласно заданному варианту. Написать два варианта программы: без применения указателей и с указателями.

Содержание отчёта

Отчёт должен содержать:

1. задание к работе;
2. программу.
3. результаты работы программы.

Контрольные вопросы

1. Как описывается массив?
2. Как описывается указатель на массив?
3. Как осуществляется поиск максимального и минимального элементов массива?
4. Как вывести матрицу на экран?
5. Как найти сумму массива, используя указатель?

Таблица 1

№ варианта	Задание
19	Определить, является ли заданная квадратная матрица A(5,5) симметричной относительно главной диагонали.
20	Задана матрица B(4,4). Определить, отсортированы ли все элементы первого столбца в возрастающем порядке.
21	Задана матрица C(5,5). Поменять местами максимальный элемент каждой строки с первым элементом соответствующей строки.
22	Переписать первые элементы каждой строки матрицы D(3,3), которые больше 10, в массив B.
23	Задана матрица Q(5,5). Заменить последний нуль в каждой строке на 5.
24	Задана матрица D(4,4). Определить максимальный среди положительных, минимальный среди отрицательных и поменять их местами.
25	Задана матрица A(4,4). Заменить первый нуль в каждом столбце на количество нулей в этом столбце.
26	Задана матрица F(9,3). определить, равны ли все элементы первого столбца соответствующим элементам главной диагонали. Если нет, то поменять их местами.
27	Задана матрица C(5,5). Получить вектор B, каждый элемент которого равен количеству нулей, стоящих в столбце матрицы.
28	Задана матрица B(4,4). Если в строке есть хотя бы одна единица, то заменить эту строку нулями.
29	Задана матрица Q(3,3). Если на главной диагонали стоит нуль, то соответствующую строку заменить единицами.
30	Задана матрица D(4,4). Если максимальный элемент матри-

	цы стоит на главной диагонали, то все элементы главной диагонали сделать равными максимальному.
31	Задана матрица C(5,5). Если минимальный элемент стоит в первой строке, то все элементы, стоящие в строке за ним, заменить нулями.
32	Задана матрица A(4,4). Если максимальный элемент матрицы равен сумме элементов первой строки, то поменять местами первую строку с той строкой, где находится максимальный элемент.
33	Задана матрица A(4,4). Если максимальный элемент матрицы равен сумме элементов первой строки, то поменять местами первую строку с той строкой, где находится максимальный элемент.

Лабораторная работа №5

«Работа с функциями в языке программирования Си++.»

Цель работы: ознакомиться с особенностями применения функций в языке Си++, с понятием прототипа и областью его применения, с понятием автоматических внешних, статических и регистровых переменных и их применением при составлении программ с использованием функций.

1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1. Функции

Вызов функции осуществляется следующим образом:

<тип функции >(параметр 1, параметр 2 , …);

Если функция имеет переменное число параметров, то вместо последнего из них указывается многоточие.

Передача одного значения из вызванной функции в вызвавшую происходит с помощью оператора возврата, который записывается в следующем виде:

return (выражение);

В этом случае значение выражения (в частном случае может быть просто переменная) передается в основную программу и подставляется вместо обращения к функции.

Пусть вызывающая программа обращается к функции следующим образом:

a=fun(b,c);

Здесь b и c – аргументы, значения которых передаются в вызываемую подпрограмму.

Если описание функции начинается так:

fun(i,j) , то переменные i и j получат значения a и b соответственно.

Пример 1. Оформить в виде функции вычисление f=√x + y/z.

В первом примере функция хранилась в виде отдельного файла и включалась процедурой #include. Функция может быть включена в один файл с вызывающей программой. В этом случае процедура #include не требуется, а сама функция должна быть объявлена в основной программе, если она имеет не целый тип. Приведем программу для примера 2, оформленную таким способом.

Программа имеет вид:

```
#include <stdio.h>
main()
{ double f,x=5.5,y=10.1,z=20.5, vv()  /*объявлены переменные и функция vv*/
f=vv(x,y,z); /*обращение к функции vv*/
printf(“lf”,f); /*вывод результата */
}
/*функция */
double vv(x,y,z)
double x,y,z; /*объявление переменных функции */
{double f;
f=sqrt(x)+y/z; /*вычисление значения функции */
return(f); /*возврат вычисленного значения функции */
}
```

В языке СИ аргументы функции передаются по значению, т.е. вызванная функция получает временную копию каждого аргумента, а не его адрес. Это означает, что функция не может изменять оригинальный аргумент в вызвавшей ее программе. Однако это легко сделать, если передавать в функцию не переменные, а их адреса.

Пример 2. Увеличить все элементы массива a(5,5) в два раза. Оформить этот алгоритм в виде подпрограммы.

```
#include <stdio.h>
main()
{int a[5][5]; /*описание массива a*/
int i,j; /*объявление переменных i,j*/
for (i=0;i<5;i++)
for (j=0; j<5; j++)
scanf(“%d”,a[i][j]); /*ввод массива*/
mas(a); /*обращение к функции mas*/
for (i=0; i<5; i++)
for (j=0; j<5; j++)
printf(“%d”, a[i][j]); /*вывод полученного результата*/
}
/*функция*/
mas(a)
int a[][5]; /*описание массива a*/
{int i,j; /*описание переменных i,j*/
for (i=0; i<5; i++)
```

```
for (j=0; j<5; j++)
a[i][j] = 2*a[i][j]; /*увеличение элементов массива в 2 раза*/
}
```

1.2. Классы памяти

В языке СИ различают четыре основных памяти: внешнюю (глобальную), автоматическую (локальную), статическую и регистровую.

В файле с вызываемой функцией внешние переменные будут доступны после их описания с помощью ключевого слова extern.

Пример 5. Оформить в виде функции вычисление выражения:

$f=a\cdot x^2+b\cdot x+c;$

В приведенной ниже программе заданные переменные объявлены как внешние, причем основная программа и функция находятся в одном файле.

```
#include <stdio.h>
int a=5, b=7, c=10,x; /* Объявление внешних переменных a,b,c,x целого типа*/
main ()
{ int f;
scanf (“%d”, &x); /*Ввод значения переменной x*/
f=kv(); /*обращение к функции*/
printf (“%d”,f); /*вывод на экран значения переменной f*/
}
/*функция*/
kv()
{int f;
f=a*x*x+b*x+c; /*вычисление значения f*/
return (f); /*возвращает значение f вызывающей программе*/
}
```

Если сравнить эту программу с программой, приведенной в примере 2, то можно обнаружить два различия:

1) после имени функции в скобках отсутствуют аргументы;
2) в функции не объявлены переменные, с которыми работает функция.

Это стало возможным потому, что переменные объявлены внешними, а значит они известны всему файлу, в том числе и функции.

Внешние переменные должны быть описаны до функции main(). Только в этом случае они становятся внешними.

Приведем программу для этого же примера, рассмотрев случай, когда основная программа и функция расположены в разных файлах.

```
#include <stdio.h>
int a=5, b=7, c=10,x,f; /* Объявление внешних переменных a,b,c,x,f целого типа*/
main ()
{
scanf (“%d”, &x); /*Ввод значения переменной x*/
```

```
f=kv(); /*обращение к функции*/
printf (“%d”,f); /*вывод на экран значения переменной f*/
}
#include “kv.c” /*включение файла kv.c функцией kv*/
/*функция*/
kv()
{extern int a,b,c,x,f;
f=a*x*x+b*x+c; /*вычисление значения f*/
return (f); /*возвращает значение f вызывающей программе*/
}
```

Как было сказано выше, если основная программа и функция расположены в разных файлах, то переменные в функции должны быть вписаны при помощи ключевого слова extern.

Рассмотрим теперь статические переменные. Статические переменные имеют такую же область действия, как автоматические, но они не исчезают, когда содержащая их функция закончит свою работу. Компилятор хранит их значения от одного вызова функции до другого. Статические переменные объявляются с помощью ключевого слова static. Можно статические переменные описать вне любой функции. Это создает внешнюю статическую переменную. Разница между внешней переменной и внешней статической переменной заключается в области их действия. Обычная внешняя переменная может использоваться функциями в любом файле (с помощью ключевого слова extern), в то время как внешняя статическая переменная может использоваться только функциями того же самого файла.

Регистровые переменные относятся к последнему классу. Ключевое слово register указывает, что переменная, о которой идет речь, будет интенсивно использоваться. Если это возможно, значения таких переменных помещаются во внутренние регистры процессора, благодаря чему программа будет более быстрой.

Задание

1. Взять задание из таблицы 1 и написать программу, используя функцию.
2. Взять задание из лабораторной работы №4 и оформить ее решение в виде функции следующими способами:

- функция расположена после ее вызова;
- функция расположена после до ее вызова;
- функция расположена после в другом файле.

Содержание отчёта

Отчёт должен содержать:

1. задание к работе;
2. программу.
3. результаты работы программы.

Контрольные вопросы

1. Как описывается функция?
2. Какова область действия внешних переменных?
3. Какова область действия внешних переменных с учетом описания extern?
4. Как осуществляется обращение к функции?
5. Как происходит возврат результата в основную программу?
6. Какие соотношения должны быть между формальными и фактическими аргументами?

Таблица 1

N вар.	Содержание задания
1.	$f = \frac{A \cdot x^2 + B \cdot x - C}{\sin x + \cos x}$ при a=4,5; b=0,7; c=6,2; A x принимает значения 0,2; 0,56; 0,83
2.	$d = \cos(a - b) + \frac{\sin(a+b)}{\sqrt{a \cdot b \cdot c}}$, при a=0,8; b=0,16; c=0,4; a=0,6; b=0,4; c=1,2; a=0,47; b=0,1; c=0,5.
3	$n = \frac{\sqrt{a^2 + b^2}}{\ln(a+b)}$, при a=0,15; b=1,5; a=1,7; b=0,1.
4.	$k = \frac{e^{d \cdot x + y} - e^{d \cdot x - y}}{x^2 + y^2}$, при x=1,4; y=0,8; x=0,9; y=0,6; x=2,9; y=0,4; a d=5,3, при всех значениях x,y.
5.	$z = arctg(x + a) - \frac{\cos(y - b)}{a \cdot x + b \cdot y}$, при x=0,4; y=1,2; x=0,25; y=1,3; a=0,54; b=1 при всех значениях x, y
6.	$i = a \cdot \sin(x + y) - \frac{b \cdot \cos(x - y)}{\sqrt{4 \cdot x \cdot y}}$, при a=10,7; b=6,3; y=0,35; a x принимает значения 0,6; 0,51; 0,42.
7.	$c = \frac{a \cdot e^{x^2} + b \cdot e^{y^2}}{a \cdot x + b \cdot y}$, при x=3; y=4; x=1,6; y=5,8; x=4,5; y=2,7; a=7,1; b=2,4 при всех значениях x, y
8.	$t = \frac{tg(x \cdot a) - b^3}{\cos^2(y \cdot b) + a}$, при x=0,1; y=0,7; x=0,4; y=0,6 x=0,5; y=0,2; a=2; b=0,1 при всех значениях x, y
9.	$m = \frac{e^{x + a \cdot b} + e^{a \cdot x}}{\sqrt[3]{a \cdot b}}$, при a=9,7; b=2,7, a x принимает значения 4,8; 9,6; 0,44.

10.	$w = \frac{\sin^3(x - y) + \cos^3(x + y)}{e^{(a+b)\cdot x}}$, при x=0,35; y=0,1; x=0,82; y=0,12; x=0,67; y=0,3, a=0,24; b=4,9 при всех значениях x, y.
11.	$y = \frac{e^{(a^2 - b^2)\cdot x} + \ln(a \cdot x)}{\sqrt{x + a \cdot b}}$, при a=4; b=2,7, а x принимает значения 0,1; 0,25; 0,14; 0,21.
12	$f = \ln(a + 8 \cdot x) - \ln^2(b + 6 \cdot y) / \sqrt{a \cdot b + x \cdot y}$ при x =0,1; y =2,7; x =0,4; y =6,2; x =1,6; y =1,8; a =0,2;
13	$d = (e^{x-a} + e^{y+b}) / \sqrt{a \cdot x^2 + b \cdot y^2}$ при x =4,8; y =2,1; x =5,2; y =4,7; x =9,6; y =3,2; a =2,5; b =4,1; при всех значениях x, y.
14.	$n = (0,7e^{x-5} + \cos(y)) / \sqrt{a \cdot x \cdot y}$ при x =2,4; y =1,5; x =0,8; y =1,2; a =2,8 при всех x =0,5; y =2,9; значениях x, y.
15.	$g = (\sin^2(a \cdot x) + \cos^2(b \cdot y)) / \sqrt{a \cdot x + b \cdot y}$ при x =0,2; y =0,4; x =0,5; y =0,6; x =0,14; y =0,18; a =1,17; b =1,6 при всех значениях x, y

Лабораторная работа №6

«Работа со структурами в языке программирования Си++.»

Цель работы: познакомиться с понятием структуры и структурной переменной. Научиться создавать массивы структур и работать с вложенными структурами.

1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Структура – это объединение одного либо более объектов(переменных, массивов, указателей, других структур).

Объявление структуры осуществляется с помощью ключевого слова struct, за которым следует ее тип, список элементов, заключенных в фигурные скобки. Ее можно представить в следующем общем виде:

struct тип {тип элемента 1 имя элемента 1;
тип элемента n имя элемента n; };

Именем элемента может быть любой идентификатор. В одной строке можно записывать через запятую несколько идентификаторов одного типа.

Следом за фигурной скобкой, заканчивающей список элементов, могут записываться переменные данного типа, например:

struct date {…} a, b, c;

При этом выделяется соответствующая память.

Выведенное имя типа можно использовать для объявления записи, например: struct date day;. Теперь переменная day имеет тип date.

Разрешается вкладывать структуры одна на другую. Для лучшего восприятия структуры используем русские буквы в идентификаторах, в языке СИ этого делать нельзя.

Рассмотрим пример программы:

```
/* Демонстрация записи */
#include < stdio.h >
   struct computer  {   int mem;
                        int sp;
                       char model [20]; };
/* Объявление записи типа computer, состоящей из трех элементов: mem, sp, model */
struct computer pibm =
                       {512, 1, “ПЭВМ ЕС 1840.05”}
/* Объявление и инициализация переменной  pibm типа computer */
main ( )
{ printf (“ персональная ЭВМ % s\n\n “, pibm.model);
          printf ( “объем оперативной памяти - %  d  K байт \n”, pibm.mem);
          printf ( “производительность - % d млн. операций в секунду  \n”, pibm.sp);
/* вывод на экран значений элементов структуры */
}
```

В данной программе объявляется запись computer, которая состоит из трех элементов: mem (память ЭВМ), sp (быстродействие), model [20] (модель ПЭВМ). Переменная pibm имеет тип computer и является глобальной. Строки pibm.model, pibm.mem, pibm. sp в операторе printf вызывают обращение к соответствующим элементам записи pibm типа computer , которым ранее были присвоены определенные значения.

Результат работы программы имеет вид:

персональная ЭВМ ПЭВМ ЕС 1840.05
объем оперативной памяти – 512 К байт
производительность – 1 млн. операций в секунду

Рассмотрим использование в программе вложенных структур:

```
/* Демонстрация вложенных структур*/
```

```
# include < stdio.h >
struct date { int day;
     int month;
     int year;   };
/* Объявление записи типа date*/
struct person { char fam [20];
     char im [20];
     char ot [20];
     struct date f1;};
/* Объявление структуры типа person;одним из элементов записи person является запись f1
 типа date   */
main ( )
{ struct person ind1;
/* обьявление переменной ind1 типа person */
printf ( “Укажите фамилию, имя, отчество, день, \n месяц”
“ и год рождения гражданина ind1\n”);
scanf (“ % S  % S  % S  %d  %d”, &ind1.fam, &ind1.im, &ind1.ot,
& ind1.f1.day, &ind1.f1.month, &ind1.f1.year );
/* Ввод сведений о гражданине ind1 */
printf (“ Фамилия, имя, отчество: % S % S % S  \n”, ind1.fam, ind1.im, ind1.ot);
printf (“ Год рождения - % d  \n”, ind1.f1.year);
printf (“ Месяц  рождения - % d -й  \n”, ind1.f1.month);
printf (“ День  рождения - % d -й  \n”, ind1.f1.day);
/* Вывод сведений о гражданине ind1 */
}
```

Структура типа date (дата) содержит три элемента: day (день), month (месяц), year (год). Структура типа person (человек) содержит четыре элемента: fam[20] (фамилия), im[20] (имя) , ot[20] (отчество), f1 (дата рождения). Последний из них (f1) – это вложенная запись типа date.

Результаты работы программы:

Укажите фамилию, имя, отчество, день, месяц и год рождения гражданина ind1
Алексеев
Сергей
Петрович
3
5
1978

Подчеркнутая информация вводится пользователем.

Сведения о гражданине ind1
Фамилия, имя, отчество: Алексеев Сергей Петрович
Год рождения – 1978
Месяц рождения – 5-й
День рождения – 3-й

Задание

Из таблицы 1 взять задание по варианту и написать программу.

Содержание отчета

Отчет должен содержать:
1. задание к работе;
2. программу;
3. результаты расчетов.

Контрольные вопросы

1. Что такое структура?
2. Как происходит обращение к элементу структуры?
3. Как работать с вложенными структурами?
4. Как создать массив записей?

Таблица 1

№ вар.	Задание
1	Опишите запись СТУДЕНТ и поместите в нее следующую информацию: Ф.И.О., оценки (математика, физика, черчение, химия, сопромат). Определите, сколько студентов имеют неудовлетворительную оценку по математике.
2	Воспользовавшись записью СТУДЕНТ из варианта №1, определите, сколько студентов имеют неудовлетворительную оценку хотя бы по одному предмету.
3	Воспользовавшись записью СТУДЕНТ из варианта №1, определите, сколько студентов сдали все экзамены на 5.
4	Воспользовавшись записью СТУДЕНТ из варианта №1, определите средний балл группы по физике.
5	Воспользовавшись записью СТУДЕНТ из варианта №1, определите количество отличных оценок, полученных группой по всем предметам.
6	Воспользовавшись записью СТУДЕНТ из варианта №1, определите, сколько студентов имеют средний балл от 4 до 5.
7	Воспользовавшись записью СТУДЕНТ из варианта №1, определите, какое количество неудовлетворительных оценок полу-

	чено по всем предметам.
8	Воспользовавшись записью СТУДЕНТ из варианта №1, определите, какой из предметов был сдан группой лучше всего.
9	Воспользовавшись записью СТУДЕНТ из варианта №1, определите, сколько студентов не имеют задолженностей.
10	Опишите запись АНКЕТА и поместите в нее следующую информацию: Ф.И.О.(фамилия, имя, отчество), адрес (улица, номер дома, номер квартиры), пол, возраст. Определите, сколько лиц женского и сколько мужского пола проживают в одном доме.
11	Воспользовавшись записью АНКЕТА из варианта №10, определите, сколько лиц мужского пола в возрасте старше 18 лет и младше 60 проживают на одной улице.
12	Воспользовавшись записью АНКЕТА из варианта №10, определите, сколько лиц женского пола в возрасте старше 30 лет проживают в одном доме.
13	Воспользовавшись записью АНКЕТА из варианта №10, определите, сколько детей до 7 лет проживают на одной улице.
14	Воспользовавшись записью АНКЕТА из варианта №10, определите, сколько лиц мужского пола и женского в возрасте до 50 лет проживают на одной улице.
15	Воспользовавшись записью АНКЕТА из варианта №10, определите, сколько детей от 1 года до 5 проживают в одном доме.
16	Опишите запись ТРАНСПОРТ и поместите в нее следующую информацию: Ф.И.О. (фамилия, имя, отчество пассажира), багаж (количество вещей, вес в кг). Определить число пассажиров, вес багажа которых превышает 30 кг.
17	Воспользовавшись записью ТРАНСПОРТ из варианта №16, определите, имеется ли пассажир, багаж которого состоит из одной вещи весом в 20 кг.
18	Воспользовавшись записью ТРАНСПОРТ из варианта №16, определите средний вес багажа.
19	Воспользовавшись записью ТРАНСПОРТ из варианта №16, определите количество пассажиров, вес багажа которых превосходит средний.
20	Воспользовавшись записью ТРАНСПОРТ из варианта №16, определите количество пассажиров, имеющих более трех вещей.

Лабораторная работа №7

«Препроцессорные средства в языке программирования Си++.»

Цель работы: Ознакомиться с назначением и особенностями использования директив препроцессора. Научиться применять препроцессор для замены в тек-

сте, для работы с макроподстановками, а также для условной компиляции программ.

1.ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Транслятор СИ имеет встроенное средство, расширяющее возможности языка и называемое препроцессором. Он рассматривает программу до компиляции (отсюда и термин препроцессор) и заменяет символические аббревиатуры в программе на соответствующие директивы, подключает указанные файлы. Для препроцессора строки программы, начинающиеся с символа #.

Его команда:

 # include “имя файла”

уже использовалась в работе, связанной с процедурами, для подключения файлов.

Рассмотрим другую директиву препроцессора

 # define <идентификатор> <подстановка>

Пример 1.

```
# include<stdio.h>
# define TWO 2
main ( )
{    int x=TWO;
      PX;
      printf(FNT,X);
}
```

После выполнения программы на экран будет выведена следующая информация:

X равен 2

X равен 4

Рассмотрим подробно, что произошло при выполнении программы.

Оператор

int x=TWO; превращается в int x=2;

Т.е. слово TWO заменилось цифрой 2

Оператор

PX; превращается в printf(“x равно %d\n”,x);

Оператор

x=FOOR; превращается в x=TWO*TWO.

а далее в x=2*2;

Оператор

printf(FNT,X);превращается в printf(“x равен %d\n”,x);

1.1.2. Директивы #if, #ifdef, #ifndef, #else, #endif.

Директивы. рассматриваемые в этом разделе позволяют выполнить условную компиляцию. Условная компиляция- это выборная компиляция только тех частей программы, которые удовлетворяют некоторым условиям. Одна из целей условной компиляции - сделать программу более мобильной. Изменяя несколько ключевых определений в начале файла, вы можете устанавливать различные

значения и включать различные файлы для разных систем; таким образом, ненужный код не хранится в памяти во время выполнения . Решение о включении той или иной части программы применяется на этапе компиляции, а не во время выполнения.

Директива #if похожа на обычный оператор if:

#if константное выражение.

Она проверяет, будет ли отличаться от нуля выражение, составленное и константное. Выражение истинно, если оно не равно нулю. Например:

```
#if  sys==”IBM”
#include “ibm.h”
#endif
```

Как видно из примера, оно заканчивается директивой #endif.

Директива

#ifdef идентификатор

Устанавливает, определён ли в данный момент идентификатор, т.е. входил ли он в команду вида #define.

Директива

#ifndef идентификатор

проверяет, не определен ли в данный момент, указанный идентификатор.

Например:

```
#ifndef  SIZE
#define SIZE 128
#endif
```

В результате переменная SIZE получает значение 128, если ранее она не была определена программистом. За любой из перечисленных выше директив может следовать произвольное количество строк.

Указанные директивы могут быть применены вместе с инструкцией #else/ Назначение этой инструкции такое же, как и в условном операторе. Рассмотрим пример.

```
#ifdef MAVIS
#include “horse.h”
#define STABLE 5
#else
#include “cow.h”
#define STABLE 15
#endif
```

Директива #ifdef сообщает, что если идентификатор MAVIS определен препроцессором, то выполняются все последующие директивы вплоть до #else. В противном случае будут выполнены директивы, стоящие после #else до директивы #endif.

Задание

Взять задание из лабораторной работы №4. Написать программу используя операторы Паскаля. Применив директивы препроцессора языка СИ, обеспечит замену операторов Паскаля на операторы языка СИ; выполнить программу.

Содержание отчёта

Отчёт должен содержать:

1. задание к работе;
2. программу;
3. результаты работы программы.

Контрольные вопросы

1. Что такое препроцессорные средства?
2. Как осуществить замены в тексте, используя препроцессорные средства?
3. Что такое директивы препроцессора?

Литература

1. Подбельский В.В. Язык Cu ++: Учебное пособие. - М.: Финансы и статистика,1995, - 560 с.
2. Страуструп Б. Язык программирования Сг ++. - М.: Радио и связь, 1991. - 352 стр.
3. Собоцинский В.В. Практический курс Turbo Cu ++. Основы объктно- ориентированного программирования. - М.: Свет, 1993. - 236 с.
4. Романов В.Ю. Программирование на языке Cu ++. Практический подход. - М.: Компьтер, 1993. - 160 с.
5. Уинер Р. Язык турбо Cu . - М.: Мир, 1991. - 384 с.
6. Юлин В.А., Булатова И.Р. Приглашение к Cu. - Мн.: Высш. Шк., 1990,- 224 с.
7. Котлинская Г.П., Галиновский О.И. Программирование на языке Cu. -Мн.: Высш. Шк., 1991. - 156 с.