ТЕМА 2. Базовые средства языка С
Операции вывода данных

Таблица 4 - Функции вывода данных

Имя функции	Описание	Файл-прототип
printf()	Производит форматированный вывод данных в stdout	stdio.h
puts()	Выводит строку символов в stdout	stdio.h
putchar()	Выводит символ в stdout	stdio.h
cprintf()	Осуществляет форматированный вывод на экран	conio.h
cputs()	Выводит строку на экран	conio.h
putch()	Выводит символ на экран	conio.h

printf(“строка форматов”, объект, объект, …);

Пример:

```
int y;  // объявление  целочисленной переменной
int x=5; //объявление и инициализация  переменной
 …
 y=x+20;  // операция присваивания
 printf(“получено число %d \n”, y);       //вывод числа
 printf(“получено число %d \n”, x+20); //вывод значения выражения
 printf(“получено число %d %d \n”, x,y); //вывод двух объектов
```

Таблица 5 Спецификации полей данных

Формат (Спецификатор)	Типы вводимой информации
%d	Десятичное целое число
%i	Десятичное целое число со знаком
%c	Символ
%s	Строка символов
%f	Число с плавающей точкой
%u	Десятичное целое число без знака
%ld	Длинное целое
%p	Целое указателя
%o	Восьмеричное целое без знака
%x	Шестнадцатеричное целое без знака

%-10s; %6d; %8.3f; %ld.

```
int x;
float y;
        . . .
printf (”Получено x = %f  y = %f \n”, x , y);
```

\n – переход на следующую строку;
\r – возврат каретки;
\t – табуляция;
\a – звонок;
\b – возврат на один шаг;
\f - перевод формата (страницы);
**\ ** - вывод знака \ ;
\ ’ – вывод знака ’;
\ ”- вывод знака ”.

Функции вывода puts( ) и cputs()

```
# include <stdio.h> / / Подключение функций ввода/вывода
# include <conio.h >
void main (void)      / / Главная функция
 {
   printf("Система Borland C\n") ; / / Печать строки
   puts("Система Borland C");       / / Печать строки
   cputs("Система Borland C\n\r"); / / Печать строки
  }
```

cout<< выражение.
cin>> переменная.

Пример:

```
#include <iostream.h>
main()
{
int i;
cout << "Целое число?: ";
cin >> i;
cout << " Квадрат числа: " << i*i << "\n";
return 0;
}
```

Ввод данных в языке С

Таблица 6 Функции ввода

Имя Функции	**Описание**	Файл, содержащий прототип
Scanf()	Выполняет форматный ввод из потока stdin	stdio.h
Gets()	Получает строку символов из потока stdin	stdio.h
Getchar()	Вводит символ из потока stdin	stdio.h
Cscanf()	Выполняет форматный ввод с клавиатуры	conio.h
Cgets()	Считывает строку символов с клавиатуры	conio.h
Getch()	Вводит символ с клавиатуры без эхо-печати	conio.h

(c)scanf (“строка форматов", адрес, адрес,…);

Пример:

/* ввод двух целых чисел в ячейки памяти "a" и "b"*/

(c)scanf (“%d %d”, &a, &b);

где &a, &b- адреса операндов "a" и "b".

Пример:

```
/* ввод строки символов, представленных массивом */
#include <conio.h>
void main (void)
{
  clrscr ( );
  char  im [10];
  cprintf (“Введите имя:” );
  cscanf (”%s”, im) ;
  cprintf (”\n\r Ваше имя : %s \n\r”, im);
}
```

Пример:

```
# include <stdio.h>
#include <conio.h>
void main (void)
  {
   clrscr ();
   char string [40];
   printf  (”Введите строку : ”);
   gets (string);
   printf (”Строка = %s \n”, string);
  getch ( );
}
```

Пример:

```
# include <stdio.h>
int main (void)
  {
   char c;
   while
((c=getchar())!=’\n’)
        printf (”%c”, c);
    return 0;
}
```

Переменные

int x

тип *список имён переменных;*

Таблица 7 -Типы простых данных

Имя типа	***Спецификация***		Объём памяти, байт
Целые	signed char	Знаковый символьный	1
	signed int	Знаковый целый	2
	signed short int	Знаковый короткий целый	2
	signed long int	Знаковый длинный целый	4
	unsigned char	Беззнаковый символьный	1
	unsigned int	Беззнаковый целый	2
	unsigned short int	Безнаковый короткий целый	2
	unsigned long int	Безнаковый длинный целый	4
Плаваю-щие	Float	Плавающий	4
	Double	Плавающий 2-й точности	8
	long float	Длинный плавающий	8
	long double	Длинный плавающий 2-й точности	10
Прочие	Void	Пустой	
	Enum	Перечислимый	

$$N = m \cdot E \pm P$$

Таблица 8 - Числовые значения типов

Тип переменной	Количество бит	Диапазон чисел
sortint (знаковый)	8 бит (левый бит отведён под знак)	$-128 \le a \le 127$
int	16 знаковый)	$-32768 \le a \le 32767$
longint	32 (знаковый)	$-2147483648 \le a \le +2147483647$

Пример записи данных в программе.

```
int a,b,c;
float x,y;
char ch;
double e;
        unsigned u;
```

Операции над данными

операция *присваивания* ``=``

Например,

x =362;

k = k + 2;

m = c = 1;

Базовая форма

<имя>=<выражение>

Разновидность операции присваивания

<имя>=<имя> <знак операции> <выражение>

Примеры:

A = a + b $\Rightarrow$ a + = b,

A = a*b $\Rightarrow$ a * = b,

A = a*(3*b +10) $\Rightarrow$ a*=3*b+10 ,

i++ $\Rightarrow$ i++

Бинарными операциями являются:

+ - * / %

Унарные операции:

-

++ положительного (увеличения на единицу - инкремент),

-- отрицательного (уменьшения на единицу – декремент).

апостериорное приращение,

c=a+b++ , $\Rightarrow$ c1=a+b; c2=a+(b+1); c3=a+(b+2); и т. д.,

априорное приращение,

c=a+++b, $\Rightarrow$ c1=a+(b+1); c2=a+(b+2)…

```
int s, k=2, p=3;
s=p*++k;
s=p*k++;
```

Старшинство арифметических операций следующее:

++, --

- (унарный минус)

*, /, %

+, –

x+++b или z---b $\Longrightarrow$ (x++)+b или (z--)-b

```
double d;
int m;
d= 5.0/2.0; /*результат 2.5*/
m=5/2;      /* результат 2*/
```

Поразрядные (побитовые) операции

& AND (и),

| OR (или),

^ XOR,

~ NOT (не),

<< сдвиг влево,

>> сдвиг вправо.

ch = ch & 127;

Пусть ch = ’A’

‘A’ в двоичном коде: 11000001;

127 в двоичном коде: 01111111

A’ & 127: 01000001.

ch = ch | 128

‘A’ 11000001

127 10000000

‘A’ | 128 11000001.

Форма представления:

value >> число позиций;

value<< число позиций.

Пример. Двоичное представление числа x = 9: 00001001.

Тогда x = 9<<3: 01001000,

x = 9>>3 : 00000001.

Пример. Пусть unsigned char x = 255 в двоичном виде 11111111.

X = x<<3: 11111000

X = x>>3: 00011111

X = x>>5: 00000111.

Примеры операций отношения:

a>b; a>=b; a<=b; a= =b; a != b,

где = = знак «равно»;

!= знак «не равно».

< меньше,

> больше,

>= больше или равно,

<= меньше или равно.

Примеры логических операций:

a&&b - операция логическое “И”,

a | | b - операция логическое “ИЛИ”,

! a - операция логическое “НЕ”.

Пример .Найти значение выражения

$$\frac{(a+b)^2-(a^2+2ab)}{b^2}$$ при a=95.0; b=0.02;

```
#include "stdio.h"
#include "conio.h"
void main()
{
   float a=95.0,b=0.02,c,t1,t2,t3;
   t1=(a+b)*(a+b);
   t2=-2.0*a*b-a*a;
   t3=b*b;
   c=(t1+t2)/t3;
    printf("\n значение выражения %f", c);
    getch();
}
```

Стандартные математические функции

double cos(*double* x); - косинус;
double sin(*double* x); - синус;
double tan(x) - тангенс;
double log(*double* x); - логарифм натуральный;
double sqrt(*double* x); - корень квадратный;
double floor(*double* x); - ближайшее меньшее целое;
double ceil(*double* x); - ближайшее большее целое;
int abs(*int* i); - модуль целого числа;
double acos(*double* x); - арккосинус;
double fabs(*double* x); - модуль числа с плавающей точкой;
double asin(*double* x); - арксинус;
double atan(*double* x); - арктангенс;
srand (seed) *int* seed; - инициализация генератора случайных чисел (ГСЧ)
rand( ) и *int* rand( ); - ГСЧ;
longint time(p), *longint* p - время в секундах, отсчитываемое от 1.01.1970 г. (0.00 по Гринвичу).
delay(t); - задержка во времени на t микросекунд;
double pow (*double* x, *double* y) и
long double pow (*long double* x, *long double* y) - возвращает значение, равное xy.
double exp(*double* x) и
long double exp (*long double* x) – возвращает значение exp(x).

Символьный тип

```
char c;                 /* байтовое число со знаком */
unsigned char u; /* байтовое число без знака */
```

```
c = 45;
```

'S' '&' '8' 'ф'

```
char c = 68;
char c = 'D';
```

Строки

"Образец строки"

"\n Текст \n разместится \n в 3-х строках дисплея"

(1) В отдельной переменной. Ее следует передавать во все функции обработки данной строки (причем она может изменяться).

```
char str[32];          /* массив для строки */
int slen;              /* брать первые slen букв в этом массиве */
...
func(str, &slen);      /* ДВА аргумента для передачи ОДНОЙ строки */
...
```

(2) Хранить текущую длину в элементе str[0], а буквы - в str[1] ... и т.д.

(3) Не хранить длину НИГДЕ, а ввести символ-признак конца строки.

```
func(str);      /* ОДИН аргумент - сам массив */
```

"Шалтай-Болтай \
сидел на стене."

Другие операции

Оператор условия

?:

выражение_1 ? выражение_2 : выражение_3

Примеры использования:

1. Нахождения max из двух чисел:

```
max = (a>b)? a: b;
```

2. Нахождение абсолютной величины числа x:

```
abs = (x >0)? x: -x;
```

sizeof – операция вычисления (в байтах) для объектов

Приведение типов.

(тип) выражение – преобразует выражение к данному типу.

```
int n=5, k=2;
double d;
int m;
d=(double) n/ (double) k;
m=n/k;/*d=2.5, m=2*/
```

Пример 1 Что напечатает следующая программа ?

```
main()
{  int x1,x2,x3,x4;
x1 = - 3 + 4 * 5 - 6;  printf("%d\n",x1);     (Операции 1.1)
x2 = 3 + 4 % 5 - 6; printf("%d\n",x2);        (Операции 1.2)
x3 = - 3 * 4 % - 6 / 5; printf("%d\n",x3);    (Операции 1.3)
x4 = ( 7 + 6 ) % 5 / 2; printf("%d\n",x4);    (Операции 1.4)
```

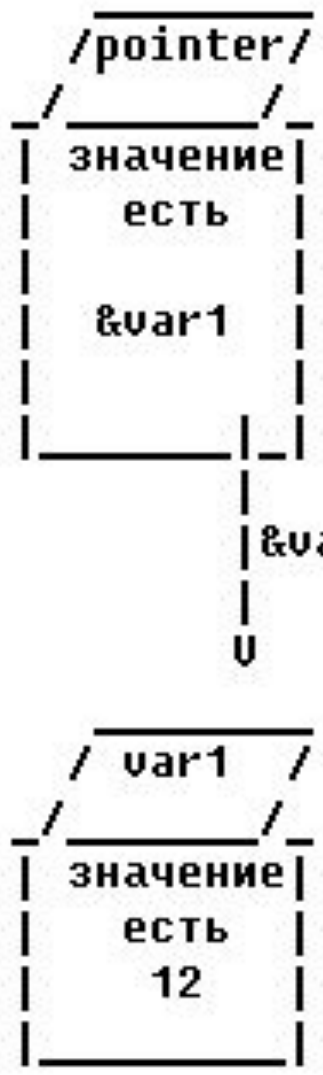
```
#include <stdio.h>
#define PR(x)
printf("x = %.8g\t",(double)(x))
#define NL putchar('\n')
#define PRINT1(x1) PR(x1);NL
#define PRINT2(x1,x2) PR(x1);
PRINT1(x2)
main()
{
    double d=3.2, x;
    int    i=2, y;
    x = (y=d/i)*2; PRINT2(x,y);
    y = (x=d/i)*2; PRINT2(x,y);
    y = d * (x=2.5/d); PRINT1(y);
    x = d * (y = ((int)2.9+1.1)/d); PRINT2(x,y);
}
```

Операции над адресами

&- определение адреса операнда;

***** - обращение по адресу.

```
int var1, var2, z;          /* целочисленные переменные */
int *pointer;          /* указатель на целочисленную переменную */
var1 = 12;
var2 = 43;
pointer = &var1;
```



Переменная, хранящая указатель
(адрес другой переменной)

- сам указатель на var1 -
"стрелка, указывающая на переменную var1".
Она обозначается &var1

```
int x;
int arr[5];
```

Законно	&x	стрелка на ящик "x"
	& arr[3]	стрелка на ящик "arr[3]"
Незаконно	&(2+2)	тут нет именованного "ящика", на который указывает стрелка, да и вообще ящика нет.

если bip – переменная типа int, то &bip – адрес переменной, по которому она расположена в памяти;

Если char *prt –указатель на данное типа char, то *prt – это само данное типа char (символ).

Пример операции над адресами:

```
# include <conio.h>
void main (void)
{
    int bip;
    char *prt;
    bip = 2+3;
    prt  = ‘’Язык Turbo C\n”
    cprintf (“bip = %d  &bip = %p\n\r”, bip, &bip);
    cprintf (“*prt = %c  prt = %p\n\r”,  *prt, prt);
}
```