

## Простейшие типы данных. Функции ввода и вывода

### Задание 1.1

Выполнено в сети Интернет.

### Задание 1.2

1. Ввести программу, запустить на выполнение, объяснить результат. Исправить ошибку.

```
#include<stdio.h>
void main()
{
    int z1=12, z2=-88, z3=32789;
    printf("z1=%d\n", z1);
    printf("z2=%d\n", z2);
    printf("z3=%d\n", z3);
}
```

Результат выполнения программы:

```
z1=12
z2=-88
z3=32789

Process returned 9 (0x9)   execution time : 0.059 s
Press any key to continue.
```

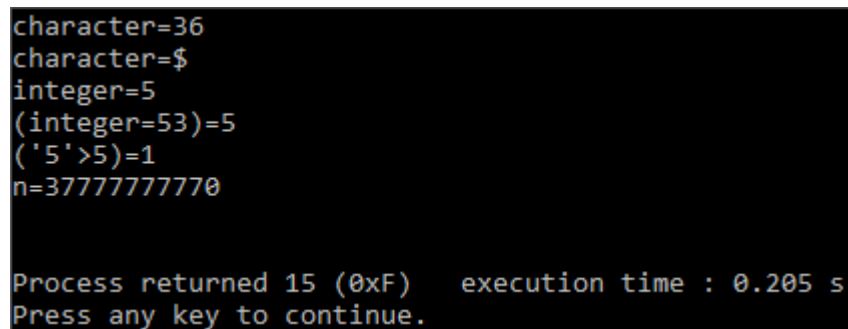
Программа вывела значения трех переменных, значения которых мы определили при вводе данных.

Так как в современном компиляторе CodeBlocks все работает без ошибок, то могу предположить, что ошибка для «старых» компиляторов могла быть в том, что тип `int` довольно маленький и переменная `z3` могла просто не вывестись на экран, поэтому в «старых» компиляторах использовался бы тип `long int` для переменной `z3`.

2. Ввести программу, запустить на выполнение, объяснить каждый результат.

```
#include<stdio.h>
void main()
{
int integer=5, n=-8;
char character="5";
printf("character=%d\n", character);
printf("character=%c\n", character);
printf("integer=%d\n", integer);
printf("(integer=53)=%c\n", integer=53);
printf("(„5“>5)=%d\n", „5“>5);
printf("n=%o\n\n", n);
}
```

Результат выполненной программы:



```
character=36
character=$
integer=5
(integer=53)=5
('5'>5)=1
n=3777777770

Process returned 15 (0xF)   execution time : 0.205 s
Press any key to continue.
```

Первый результат необъясним так же, как и второй.

Третий результат выводит правильно значение переменной integer.

Четвертый результат вывел строку “(integer=53)” и приравнял ее к начальному значению переменной integer, хоть мы и пытались заменить его на значение 53.

Пятый результат вывел строку “(‘5’>5)” и приравнял ее к единице, не понятно по какой причине.

Шестой результат вывел значение переменной n в восьмеричной системе счисления без знака и сделал 2 переноса строки.

3. Ввести программу, запустить на выполнение, объяснить каждый результат.

```
#include<stdio.h>
void main()
{
float z1=2,5, z2=5,67;
double u1=2,5, u2=5,67;
printf("результат для типа float : %f\n", z1+z2);
printf("результат для типа double : %e\n", u1+u2);

printf("самый короткий результат : %g\n", u1+u2);
}
```

Результат выполненной программы:

```
float resalt: 8.170000
double resalt: 8.170000e+000
the shortest resalt: 8.17

Process returned 26 (0x1A)   execution time : 0.175 s
Press any key to continue.
```

Первый результат выводит сумму двух значений переменных типа float. После запятой стоит столько знаков, сколько определяется типом float.

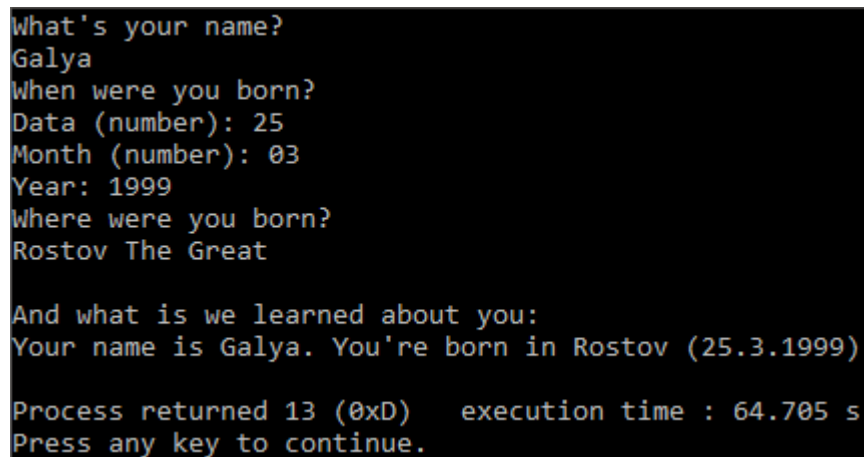
Второй результат также выводит сумму двух значений переменных, но уже типа double. После запятой стоит столько знаков, сколько определяется типом double.

Третий результат вывел сумму двух значений переменных типа double. Но после запятой стоит только столько знаков, сколько есть в значении результата, что продиктовано типом %g. Он использует типы float и double, но тот, что короче.

4. Ввести программу «ввод данных с клавиатуры», запустить на выполнение, объяснить каждый результат.

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int data, month, year;
    char name[15], town[15];
    printf("как вас зовут?"); scanf("%s", name);
    printf("укажите дату, месяц и год Вашего рождения.\n");
    printf("Дата (число): "); scanf("%d", &data);
    printf("Месяц (номер): "); scanf("%d", &month);
    printf("Год : "); scanf("%d", &year);
    printf("В каком городе родились?"); scanf("%s", town);
    printf("\nВот мы о Вас и узнали кое-что. А именно...\n ");
    printf("Вас зовут %s. Вы родились в городе %s (%d.%d.%d)\n", name,
    town, data, month, year);
    getch();
}
```

Результат выполненной программы:



```
What's your name?
Galya
When were you born?
Data (number): 25
Month (number): 03
Year: 1999
Where were you born?
Rostov The Great

And what is we learned about you:
Your name is Galya. You're born in Rostov (25.3.1999)

Process returned 13 (0xD)   execution time : 64.705 s
Press any key to continue.
```

Первые строчки нацелены на вывод «помогающей» фразы и считывания вводимой информации.

После считывания нужной информации с помощью строки 14 выводится результат в нужном порядке.

5. Вычислите значение арифметического выражения. Объясните каждый результат (приоритеты выполнений).

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int x;
    x=-3+4*5-6; print("1. x=%d\n", x);
    x=3+4%5-6; print("2. x=%d\n", x);
    x=-3*4%-6/5; print("1. x=%d\n", x);
    x=(7+6)%5/2; print("1. x=%d\n", x);
    getch();
}
```

Результат выполненной программы:

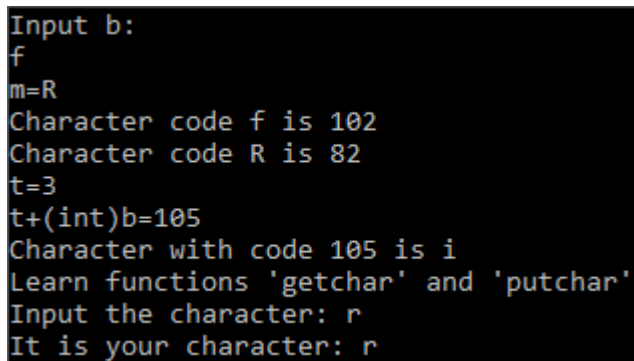
```
1. x=11
2. x=1
3. x=0
4. x=1
```

1. Первым выполняется умножение, а затем по порядку сложение.
2. Первым считается остаток от деления, а затем по порядку сложение.
3. Действия выполняются по порядку, т.к. одного приоритета, т.е. сначала умножение, далее деление с остатком, а затем деление нацело.
4. Первым выполняется действие в скобочках, а затем по порядку деление с остатком и деление нацело.

6. Вычислите значение арифметического выражения. Объясните каждый результат (операции над типом char).

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int t=3;
    char b,
    m="R";
    printf("\nВведите значение b:"); scanf("%c", &b);
    printf("m=%c\n", m);
    printf("Код символа %c равен %d\n", b, (int)b);
    printf("Код символа %c равен %d\n", m, m);
    printf("t=%d\n", t);
    printf("t+(int)b=%d\n", t+=(int)b);
    printf("Символ с кодом %d – это %c\n", t, (char)t);
    printf("Познакомимся с функциями getchar и putchar\n");
    fflush(stdin); /* очистка буфера клавиатуры */
    printf("Введите символ: "); b=getchar();
    printf("Вот Ваш символ: "); putchar(b);
    printf("\n");
    getch();
}
```

Результат выполненной программы:



```
Input b:
f
m=R
Character code f is 102
Character code R is 82
t=3
t+(int)b=105
Character with code 105 is i
Learn functions 'getchar' and 'putchar'
Input the character: r
It is your character: r
```

Программа:

1. считывает вводимый нами символ
2. выводит значение переменной m
3. выводит код символа, который мы ввели
4. выводит код символа, который хранится в переменной m
5. выводит значение переменной t
6. присваивает переменной t новое значение равное сумме предыдущего значения и арифметического значения кода введенного нами символа
7. выводит символ, код которого равен новому значению переменной t

8. с помощью вспомогательной текстовой строки считывает новый символ, введенный нами с клавиатуры, и записывает его в переменную `b`

9. выводит переменную `b`