**Практическая работа 2**

Тема: Форматированный вывод

**Теоретические сведения**

Вывод символов на консоль осуществляется в языке Cи с помощью функции *printf(...).*

printf(“%d”, 12);

На экран выводится число 12



Эта функция может иметь переменное число аргументов, но первым всегда является управляющая форматом вывода строка.

Управление форматированием необходимо для представления данных в необходимом пользователю виде.

**Формат данных определяет их внешнее представление, т.е. оформление их записи**

*Например, число 500 может быть записано различными способами:*

|  |  |  |
| --- | --- | --- |
| *500,0* | *«пятьсот»* | *5е2 (*5 ∙ 102*)* |

*в зависимости от заданного формата:*

|  |  |  |
| --- | --- | --- |
| *с фиксированной запятой* | *прописью* | *в экспоненциальной форме*  (научная нотация) |

Для обеспечения возможности вывода данных различными способами и применяется управляющая строка или строка формата.



Управляющая строка содержит:

1. обычные символы, которые непосредственно выводятся на консоль (буквы, цифры, символы);
2. управляющие символьные константы (табл. 1), которые определяют расположение выводимых элементов;

Таблица 1. – Управляющие символьные константы

|  |  |
| --- | --- |
| \n | Новая строка, перевод строки |
| \r | Возврат каретки |
| \t | Горизонтальная табуляция |
| \v | Вертикальная табуляция |
| \" | Двойная кавычка |
| \\ | Обратная косая черта |
| \? | Знак вопроса |

1. спецификаторы, начинающиеся с символа **%** и заканчивающиеся некоторым символом, задающим преобразования (табл. 2), которые вызывают вывод на консоль значения очередного аргумента из последующего списка переменных.

Таблица 2. – Спецификация преобразования переменных

|  |  |
| --- | --- |
| %с | символ |
| %d или %i | целое десятичное число со знаком |
| %e или %E | десятичное число в виде x.xx e+xx |
| %f или %F | десятичное число с плавающей запятой xx.xxxx |
| %g или %G | %f или %e, с выбором по принципу, что короче |
| %o | восьмеричное число |
| %s | строка символов |
| %u | беззнаковое десятичное число |
| %x или %X | шестнадцатеричное число |
| %lf или %lg | длинное вещественное число |

Общий вид спецификатора формата вывода следующий:

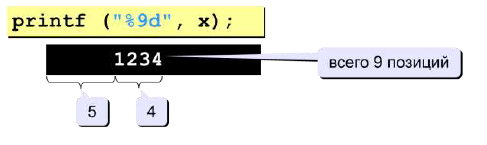
%[флаг][ширина][.точность][h|l|L]тип ,

где ***флаг*** – символ, указывающий на особенности заполнения полей вывода (табл. 3);

Таблица 3. – Спецификатор вывода флаг

|  |  |  |
| --- | --- | --- |
| **Знак** | **Значение** | **В отсутствие этого знака** |
| – | выводимое значение выравнивается по левому краю в пределах минимальной ширины поля | по правому |
| **+** | всегда указывать знак (плюс или минус) для выводимого десятичного числового значения | только для отрицательных чисел |
|  | помещать перед результатом пробел, если первый символ значения не знак | Вывод может начинаться с цифры |
| 0 | дополнять поле до ширины, указанной в поле *ширина* управляющей последовательности, символом 0 | дополнять пробелами |

***ширина*** (десятичное число или символ звёздочка) – указывает минимальную ширину поля (включая знак для чисел). Если представление величины больше, чем ширина поля, то запись выходит за пределы поля, если представление величины менее указанного числа, то оно будет дополнено (по умолчанию) пробелами слева, поведение может меняться предшествующими флагами.

****

**точность** – число, которое указывает:

* на минимальное количество символов, которое должно появиться при обработке типов d, i, o, u, x, X;
* на минимальное количество символов, которое должно появиться после десятичной запятой (точки) при обработке типов a, A, e, E, f, F;
* максимальное количество значащих символов для типов g и G;
* максимальное число символов, которые будут выведены для типа s.

**тип** – символьная константа, указывающая на типы выводимой переменной (табл. 2).

**Задания и указания по их выполнению**

**Задание 1. Знакомство со спецификаторами**

1. напишите и запустите программу:

#include <stdio.h>

int main()

{

printf("%20s\n", "Это текст");

}

1. добавьте в программу вывод еще одного повторения строки "Это текст" с выравниванием влево;
2. добавьте табуляцию перед выводом текста;
3. замените строку 4 на новую:

printf("%10.3f\n ", 12.234657);

1. измените ее так, чтобы число выводилось с точностью - 5 цифр после запятой;
2. замените строку 4 на новую:

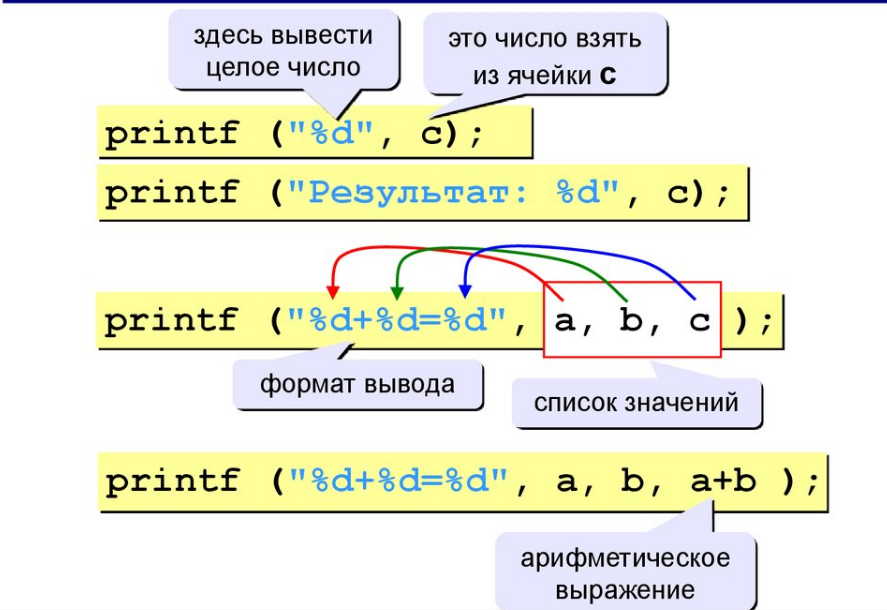
* printf ("Остаток от деления %d на %d равен %d\n ",5, 2, 5%2);

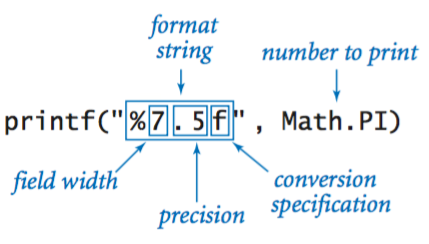
1. исправьте так, чтобы выполнить
   * деление 7 на 5
   * умножение 2000 на 4
2. замените строку 4 на новую:

* printf ("%g разделить %e равно %f\n ",5., 2., 5./2);

1. в этой строке установите все спецификаторы преобразования данных на d (что произойдет?), потом на f, потом на g и e (укажите в чем различие).







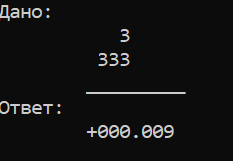
**Задание 2. Напишите программу вывода числовых значений "времени", для этого**

* 1. Инициализируйте значения переменных N и K значениями текущего времени (N - часов, K-минут)

int N, K;

* 1. Используя переменные N и К, выведите на экран сообщения:
* «*Сейчас \_ часов \_ минут 00 секунд*»
* «*Идет \_ минута суток*»
* «*До полуночи осталось \_ часов и \_ минут*»
* «*С 8.00 прошло \_ секунд*»
* «*Текущий час = N⁄24 суток и текущая минута = K⁄60 часа*»

**Задание 3. Напишите программу деления числа n на 333 с использованием одной функции printf и выведите результат в формате k цифр целой части и m – дробной. (по вариантам)**



Добавьте в программу:

1. Вывод в две строки - сначала данные, потом результат.
2. Данные отформатируйте по правому краю, результат по левому
3. Сделайте так чтобы в ответе выводился знак числа.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Вариант | n | k | m | Вариант | n | k | m |
| 1 | 2 | 3 | 1 | 16 | 2 | 3 | 4 |
| 2 | 3 | 4 | 2 | 17 | 4 | 3 | 3 |
| 3 | 4 | 5 | 3 | 18 | 4 | 5 | 4 |
| 4 | 2 | 3 | 3 | 19 | 4 | 3 | 4 |
| 5 | 4 | 3 | 2 | 20 | 4 | 5 | 5 |
| 6 | 4 | 5 | 1 | 21 | 2 | 2 | 3 |
| 7 | 2 | 2 | 4 | 22 | 3 | 3 | 4 |
| 8 | 3 | 3 | 5 | 23 | 4 | 2 | 5 |
| 9 | 4 | 2 | 6 | 24 | 2 | 3 | 6 |
| 10 | 2 | 3 | 7 | 25 | 3 | 2 | 3 |
| 11 | 3 | 2 | 2 | 26 | 4 | 3 | 3 |
| 12 | 4 | 3 | 1 | 27 | 3 | 4 | 3 |
| 13 | 2 | 3 | 2 | 28 | 4 | 2 | 4 |
| 14 | 3 | 4 | 1 | 29 | 4 | 2 | 2 |
| 15 | 4 | 5 | 2 | 30 | 3 | 3 | 4 |