1.1. Первая программа. Вывод текста и комментирование кода.

Исходный код

main

Wednesday, March 11, 2020 20:28

Hello world!

Как и в случае с микроконтроллерами, начнем с написания простейшей программы. Здесь наипростейшей программой считается вывод в консоль какой-нибудь фразы, часто "Hello world!". Запишем код и разберем его.

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5         printf("Hello world!\n");
6
7         return 0;
8 }
```

Итак, по порядку. Во-первых, в C важен регистр букв, поэтому надо за ним следить.
Например, `Int` и `int` - совершенно разные вещи с точки зрения языка. Основной файл может называться как угодно, но общепринятое название - main. Расширение обязательно должно быть .c

Теперь перейдем к разбору кода.
Первая строка подключает библиотеку `stdio.h`:

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

Она обычно будет нужна во всех программах на C, потому что эта подключаемая библиотека содержит рутины для ввода-вывода, на что и намекает ее название. Функция `printf` также является ее частью и не будет работать, если мы не подключим эту библиотеку.

Со строки

```
int main(void)
```

начинается выполнение программы. Это стандарт в C, отклоняться от которого недопустимо [2]. Основная функция должна называться `main` и возвращать значение типа `int`. Слово `void` означает, что функция не принимает на вход никакие значения. В некоторых случаях функция `main` может выглядеть следующим образом:

```
int main(int argc, char *argv[])
```

В нашем контексте эти функции идентичны и разницу между ними мы рассмотрим позже.

Строка

```
printf("Hello world!\n");
```

выводит текст в скобках в консоль. Параметр `\n` дает команду на перенос курсора на новую строку. Как println в паскале. Без него тоже будет работать, но так красивее выглядит, наверное.

И, наконец, последняя строка

```
return 0;
```

завершает функцию `main()` и возвращает значение типа `int`. Вместо 0 может быть любое значение типа `int`. Программисты договорились, что возвращение 0 означает, что программа завершилась без ошибок. Разными числами можно возвращать разные типы ошибок (вспомните, как разные программы часто выдают ошибки с номером).

Выведем несколько строк по порядку:

```
printf ("Testing...\n..1\n...2\n....3\n");
```

В результате на выходе получим:

```
Testing...
```

```
..1
...2
....3
```

Теперь попробуем вывести значения каких-нибудь переменных. Объявим три целочисленных (integer) переменных – `value1`, `value2` и `sum`, подставим их в простую формулу и выведем результаты.

```
#include <stdio.h>

int main (void)
{
          int  value1, value2, sum;

          value1 = 50;
          value2 = 25;
          sum = value1 + value2;
          printf ("The sum of %i and %i is %i.\n", value1, value2, sum);

          return 0;
}
```

На выходе получим:

```
The sum of 50 and 25 is 75.
```

Как видим, параметр `%i` подставляет значение переменной той же позиции в списке, что и параметр в строке.

Комментирование кода

Есть два способа комментировать код:

```
/* комментарий1
   комментарий 2
*/
```

и

```
// комментарий
```

Первый вариант обычно используется когда нужно вставлять большие блоки комментариев. Второй вариант работает в пределах одной строки. Следует обратить внимание, что если открыть комментарий `/*` и не закрыть его (`*/`), то будет закомментирован весь текст, который находится после открывающего оператора.

Давайте закомментируем наш код:

```
/* This program adds two integer values
   and displays the results             */

#include <stdio.h>

int main(void)
{
        // Declare variables
        int value1, value2, sum;

        // Assign values and calculate their sum
        value1 = 20;
        value2 = 50;
        sum = value1 + value2;

        // Display the result
        printf ("The sum of %i and %i is %i.\n", value1, value2, sum);

        return 0;
}
```

Теперь код гораздо удобнее читать. Главное – не увлекаться и не комментировать очевидное, как мы только что сделали.

Источники:

1. Stephen Kochan – Programming in C (4th Edition); chapter 2
2. https://stackoverflow.com/questions/449851/why-do-we-need-to-use-int-main-and-not-void-main-in-c