СТРУКТУР
НА СИ/СИ

Функции – это самостоятельные подпрограммы, выполняющие определенную подзадачу. Функции могут выполнять некоторые действия – печать и обработку данных, возвращать значения (как например, sin, cos)

Общая структура п

```
директивы_препроцессора
определение_функции_1
определение_функции_2
        … … … … … … …
определение_функции_N
```

Среди функций обязательно присутствует главная функция с именем **main – точка входа в программу**

Простейшая программа содержит только главную функцию и имеет следующую структуру:

```
директивы_препроцессора
int main()
{  определения_объектов
   исполняемые_операторы;
}
```

void main()

Пока будем составлять простейшие программы.

Пример печати на экране приветствия

Директива препроцессора

```
#include <stdio.h>
int main( )
{ //Печать приветствия
  printf (“Hello World");
}
```

Заголовок главной функции

Строка комментария компилятором не обрабатывается

В наших примерах программ будет использоваться ввод исходных данных либо с клавиатуры, либо из файла. Язык Си предоставляет также возможность указывать аргументы программы в командной строке.

Аргументы командной строки являются параметрами функции main, с которой начинается выполнение Си-программы. Мы будем применять вариант функции main без параметров, однако, при необходимости доступа к аргументам командной строки можно использовать следующий заголовок функции main:

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

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

Здесь целая переменная argc равна числу аргументов, т.е. отдельных слов командной строки, а массив argv содержит указатели на строки, каждая из которых равна очередному слову командной строки. Нулевой элемент *argv[0]* равен имени программы. Таким образом, число аргументов argc всегда не меньше единицы.

Например, при запуске программы testprog с помощью командной строки

```
testprog -x abcd.txt efgh.txt
```

значение переменной argc будет равно 4, а массив argv будет содержать 4 строки "testprog", "-x", "abcd.txt" и "efgh.txt".

Простейшая программа нахождения суммы двух чисел a и b.

```
#include <stdio.h>
int main ()
{   float s, a = 5.3, b = 9.733;
    s=a+b;
    printf ("\nСумма %f + %f =%f", a, b, s);
}
```

a и b - исходные данные

s - результирующая переменная

```
#include <stdio.h>
int main ( )
{   float s, a = 5.3, b = 9.733;
    s=a+b;
    printf ("\nСумма %f + %f =%f", a, b, s);
}
```

объявление используемых переменных

исполняемые операторы

главной функции

Программа состоит из одной главной функции со стандартным именем main.

Слово int означает, что функция main возвращает ОС целое значение, пустые скобки ( ) – отсутствие у функции аргументов.

В случае использования void перед функцией означает отсутствие какого-либо возвращаемого значения функцией.

Пример печати на экране приветствия

```
#include <stdio.h>
int main( )
{  //Печать приветствия
   printf ("Привет, МИР!!!");
}
```

Пример печати на экране приветствия

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

int main( )
{  setlocale(LC_ALL, “rus"
   printf ("Привет МИР!!!");
}
```

Пример печати на экране приветствия

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

int main( )
{  setlocale(LC_ALL, “rus"
   printf ("Привет, МИР!!!");
}
```

Вывести каждое слово в отдельной строке

Вывести слово "МИР" в кавычках

Кроме стандартного текста мы можем передавать в строку специальные группы символов, которые называются управляющими последовательностями. Наиболее распространенные из них:

\n: перевод на новую строку

\t: табуляция

\r: возврат каретки (курсора) в начало строки

\\: обратный слеш

\': одинарная кавычка

\": двойная кавычка

```
#include <stdio.h>
int main ()
{   float s, a = 5.3, b = 9.733;
    s=a+b;
    printf ("\nСумма %f + %f =%f", a, b, s);
}
```

1. **Поменяйте тип переменных на целый**
2. **Вычислите** s=a/b;
3. **Поменяйте тип переменных на double**

print.cpp | Безымянный1.cpp

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
 float m=84.3,p=32.15;
 int k=-12;
 printf("\nm=%6.3f\tk=%8d\tp=%8.2e\tp=%11.4e\n", m, k, p, p);
 system ("Pause");
}
```

D:\ковчег\Алутина\Программирование\К занятиям\Задания к занятиям\print.exe

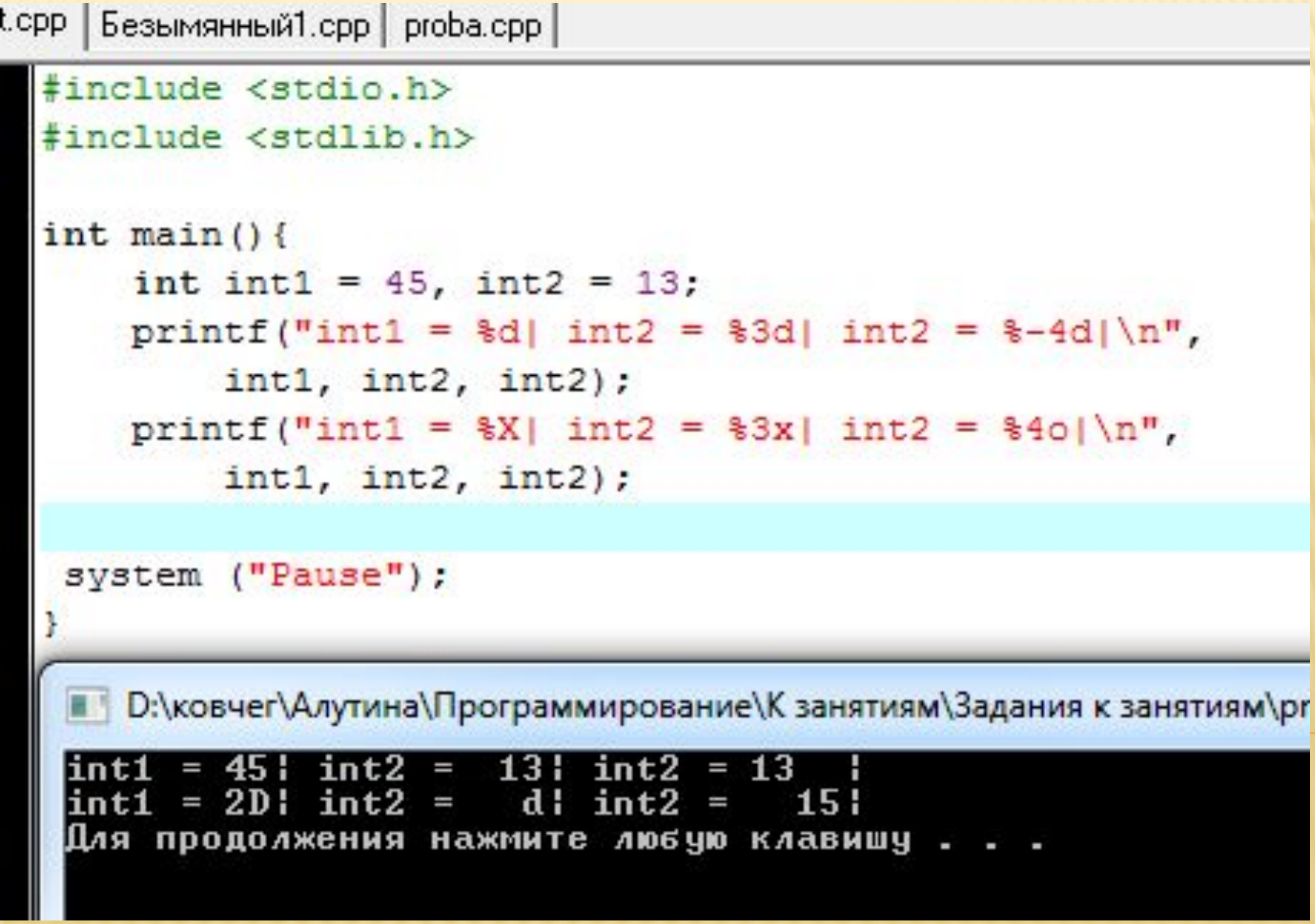
```
m=84.300        k=      -12      p=3.22e+001      p=3.2150e+001
Для продолжения нажмите любую клавишу . . .
```

print.cpp | Безымянный1.cpp

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
 float m=84.3,p=32.15;
 int k=-12;
 printf("\nm=%6.3f\tk=%8d\tp=%8.2e\tp=%11.4e\n", m, k, p, p);
 system ("Pause");
}
```

D:\ковчег\Алутина\Программирование\К занятиям\Задания к занятиям\print.exe

```
m=84.300        k=      -12      p=3.22e+001      p=3.2150e+001
Для продолжения нажмите любую клавишу . . .
```


t.cpp | Безымянный1.cpp | proba.cpp |
#include <stdio.h>
#include <stdlib.h>
int main(){
int int1 = 45, int2 = 13;
printf("int1 = %d| int2 = %3d| int2 = %-4d|\n",
int1, int2, int2);
printf("int1 = %X| int2 = %3x| int2 = %4o|\n",
int1, int2, int2);
system ("Pause");
}
D:\ковчег\Алутина\Программирование\К занятиям\Задания к занятиям\pr
int1 = 45| int2 = 13| int2 = 13 |
int1 = 2D| int2 = d| int2 = 15|
Для продолжения нажмите любую клавишу . . .

cpp | Безымянный1.cpp | proba.cpp

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

int main(){
float f = 3.621;
double dbl = 2.23;
printf("f = %f| f = %4.2f| f = %6.1f|\n", f, f, f);
printf("f = %g| f = %e| f = %+E|\n", f, f, f);
printf("dbl = %5.2lf| dbl = %e| dbl = %4.1G|\n",
        dbl, dbl, dbl);
 system ("Pause");
}
```

D:\ковчег\Алутина\Программирование\К занятиям\Задания к занятиям

```
f = 3.621000| f = 3.62| f =    3.6|
f = 3.621| f = 3.621000e+000| f = +3.621000E+000|
dbl =  2.23| dbl = 2.230000e+000| dbl =    2|
Для продолжения нажмите любую клавишу . . .
```

```
#include <stdio.h>
int main()
{
char ch = 'z', *str = "ramambahari";
printf("ch = %c| ch = %3c|\n", ch, ch);
printf("str = %14s|\nstr = %-14s|\nstr =
%s|\n", str, str, str);
}
```

```
ch = z| ch =   z|
str =    ramambahari|
str = ramambahari   |
str = ramambahari|
```