



Алгоритмизация и программирование

Программирование на С/С++
(ч.7 – функции)



#include <stdio.h>
int main(void)
{
 printf("Hello, World!\n");
}
C/C++

Беркунский Е.Ю., кафедра ИУСТ, НУК
eugeny.berkunsky@gmail.com
<http://www.berkut.mk.ua>

ФУНКЦІИ В С/С++

- Опытные программисты знают, что для написания больших программ необходимо пользоваться функциями.
- Программа будет состоять из отдельных фрагментов кода, под отдельным фрагментом кода понимается функция.
- Отдельным, потому, что работа отдельной функции практически не зависит от работы какой-нибудь другой.
- Алгоритм в каждой функции должен быть функционально достаточен и максимально независим от других алгоритмов программы.

Функції в C/C++

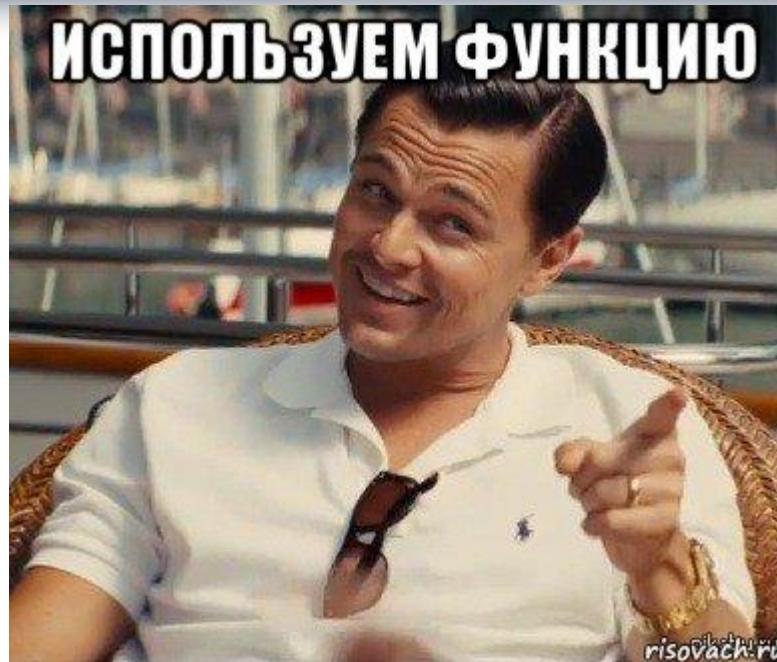
- Функция (в программировании) — это фрагмент кода или алгоритм, реализованный на каком-то языке программирования, с целью выполнения определённой последовательности операций.
- Однажды написав функцию, её можно будет с лёгкостью переносить в другие программы.

Функції в C/C++

- Функції позволяють сделать программу модульной, то есть разделить программу на несколько маленьких подпрограмм (функций), которые в совокупности выполняют поставленную задачу.
- Еще одно преимущество функций в том, что их можно многократно использовать.
- Данная возможность позволяет многократно использовать один раз написанный код, что в свою очередь, намного сокращает объем кода программы!



ФУНКЦИИ В С/С++



- Чтобы воспользоваться функцией, определённой в заголовочном файле, нужно его подключить.
- Например, чтобы воспользоваться функцией, которая возводит некоторое число в степень, нужно подключить заголовочный файл `<cmath>` и использовать функцию `pow()` в программе.

Пример программы

```
//действие 1 - подключаем заголовочный файл <cmath>
// который содержит прототипы основных математических функций
#include <cmath>

int main()
{
    double power = pow(3.1415926, 0.5);
    //действие 2 - запуск функции возведения числа в степень
    cout << power << endl;
    return 0;
}
```

ФУНКЦІИ В С/С++

- Всегда после имени функции ставятся круглые скобки, внутри которых, функции передаются аргументы, и если аргументов несколько, то они отделяются друг от друга запятыми.
- Аргументы нужны для передачи информации в функцию.
- Например, чтобы возвести число 3.1415926 в степень 0.5 используя функцию pow(), нужно как-то этой функции сообщить, какое число, и в какую степень его возводить.
- Для этого и придуманы аргументы функций, но бывают функции, в которых аргументы не передаются, такие функции вызываются с пустыми круглыми скобками.

Функції в C/C++

Для того, чтобы воспользоваться функцией из стандартного заголовочного файла C++ необходимо выполнить два действия:

- Подключить необходимый заголовочный файл;
- Запустить нужную функцию.

Кроме вызова функций из стандартных заголовочных файлов, в языке C++ предусмотрена возможность создания своих функций.



ФУНКЦИИ В С/С++



ФУНКЦІИ В С/С++

В языке программирования С++ есть два типа функций:

- Функции, которые не возвращают значений
- Функции, возвращающие значение

ФУНКЦІИ В С/С++

Функции, не возвращающие значения, завершив свою работу, никакого ответа программе не дают.

Рассмотрим структуру объявления таких функций.

```
// структура объявления функций не возвращающих значений
void /*имя функции*/ /*параметры функции*/ // заголовок
{
// тело функции
}
```

- **void** говорит о том, что данная функция не возвращает никаких значений.
- После зарезервированного слова **void** пишется имя функции.
- Затем ставится пара круглых скобок.
- Если нужно функции передавать какие-то данные, то внутри круглых скобок объявляются параметры функции, они отделяются друг от друга запятыми.
- После заголовка функции пишутся две фигурные скобки, внутри которых находится код, называемый телом функции.

Пример

```
#include <iostream>
using namespace std;

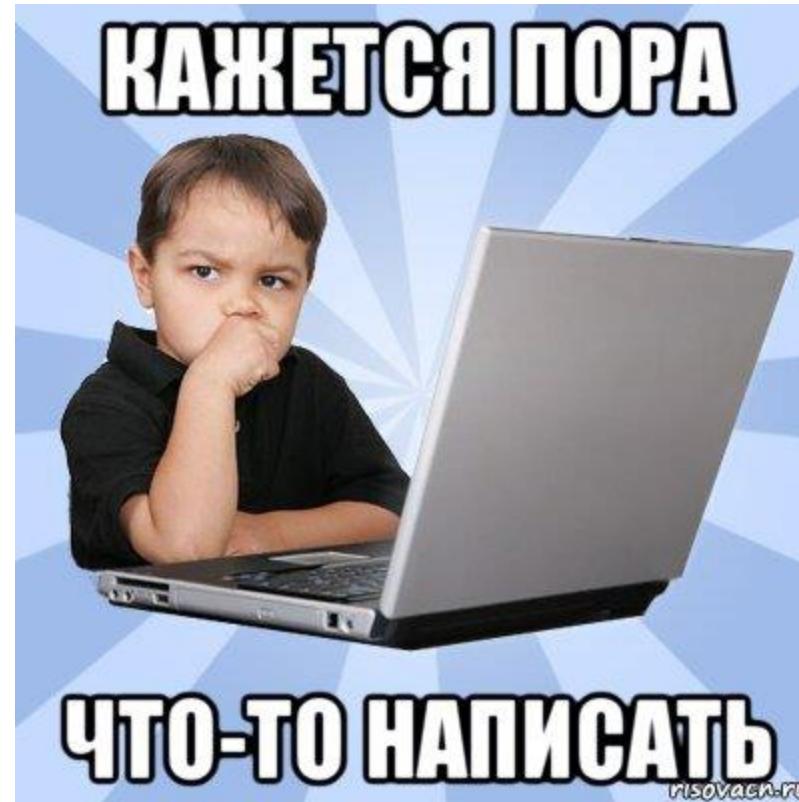
// объявление функции нахождения n!
void faktorial(int numb) // заголовок функции
{
    int result = 1;
    for (int i = 1; i <= numb; i++)
    {
        result *= i;
    }
    cout << numb << " ! = " << result << endl;
}

int main()
{
    int digit; // переменная для хранения значения n
    cout << "Enter number: ";
    cin >> digit;
    faktorial(digit); // запуск функции нахождения факториала
    return 0;
}
```

Чем плоха такая функция?



Демонстрация



ФУНКЦІИ В С/С++

- Функции, возвращающие значение, по завершению своей работы возвращают определённый результат.
- Такие функции могут возвращать значение любого типа данных.
- Структура функций, возвращающих значение будет немного отличаться от структуры функций рассмотренных ранее.

```
// структура объявления функций возвращающих значения
/*возвращаемый тип данных*/ /*имя функции*/ /*параметры*/
// заголовок функции
{
// тело функции
return /*возвращаемое значение*/;
}
```

ФУНКЦІИ В С/С++

- В заголовке функции сначала нужно определять возвращаемый тип данных, это может быть тип данных `int`, если необходимо возвратить целое число или тип данных `float` - для чисел с плавающей точкой, и т.п.
- Так как функция должна вернуть значение, то для этого должен быть предусмотрен специальный оператор **`return`**. С его помощью можно вернуть значение, по завершении работы функции.
- Для этого нужно указать переменную, содержащую нужное значение, или некоторое значение, после оператора **`return`**.
- Тип данных возвращаемого значения должен совпадать с типом данных в заголовке.

Пример

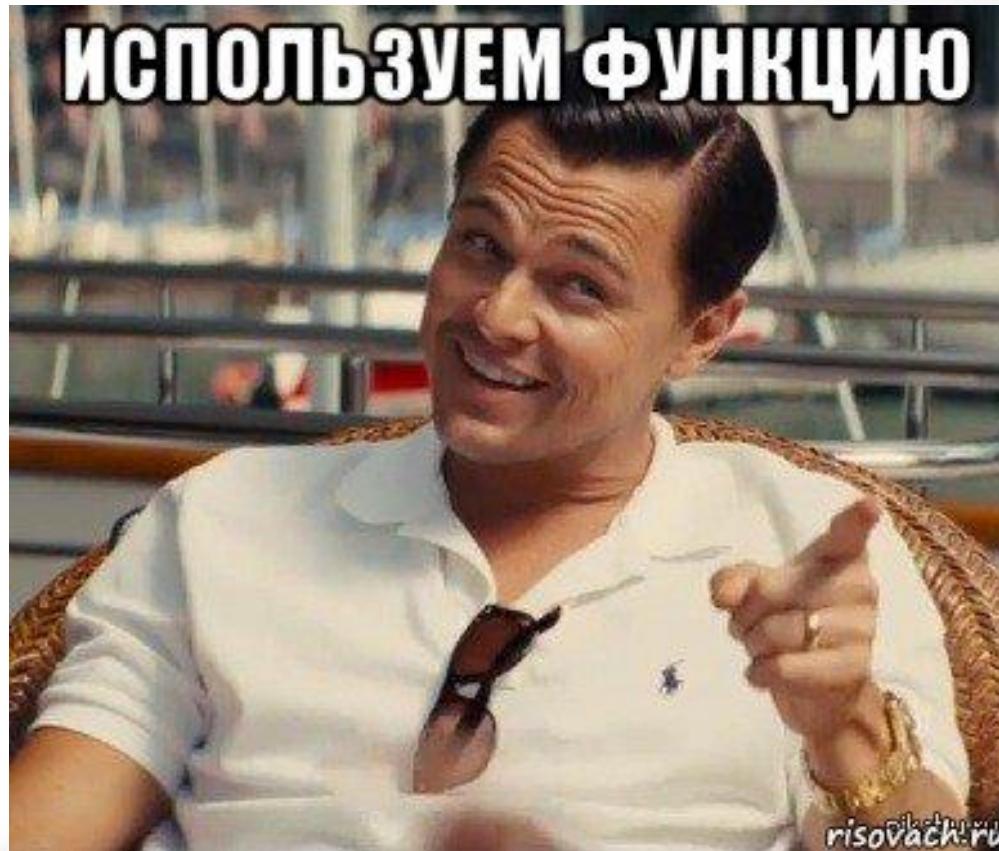
```
#include <iostream>
using namespace std;

// объявление функции нахождения n!
int faktorial(int numb) // заголовок функции
{
    int result = 1;
    for (int i = 1; i <= numb; i++)
        result *= i;
    return result;
}

int main()
{
    int digit; // переменная для хранения значения n
    cout << "Enter number: ";
    cin >> digit;
    int result = faktorial(digit);
    cout << digit << "!" = " << result << endl;
    return 0;
}
```

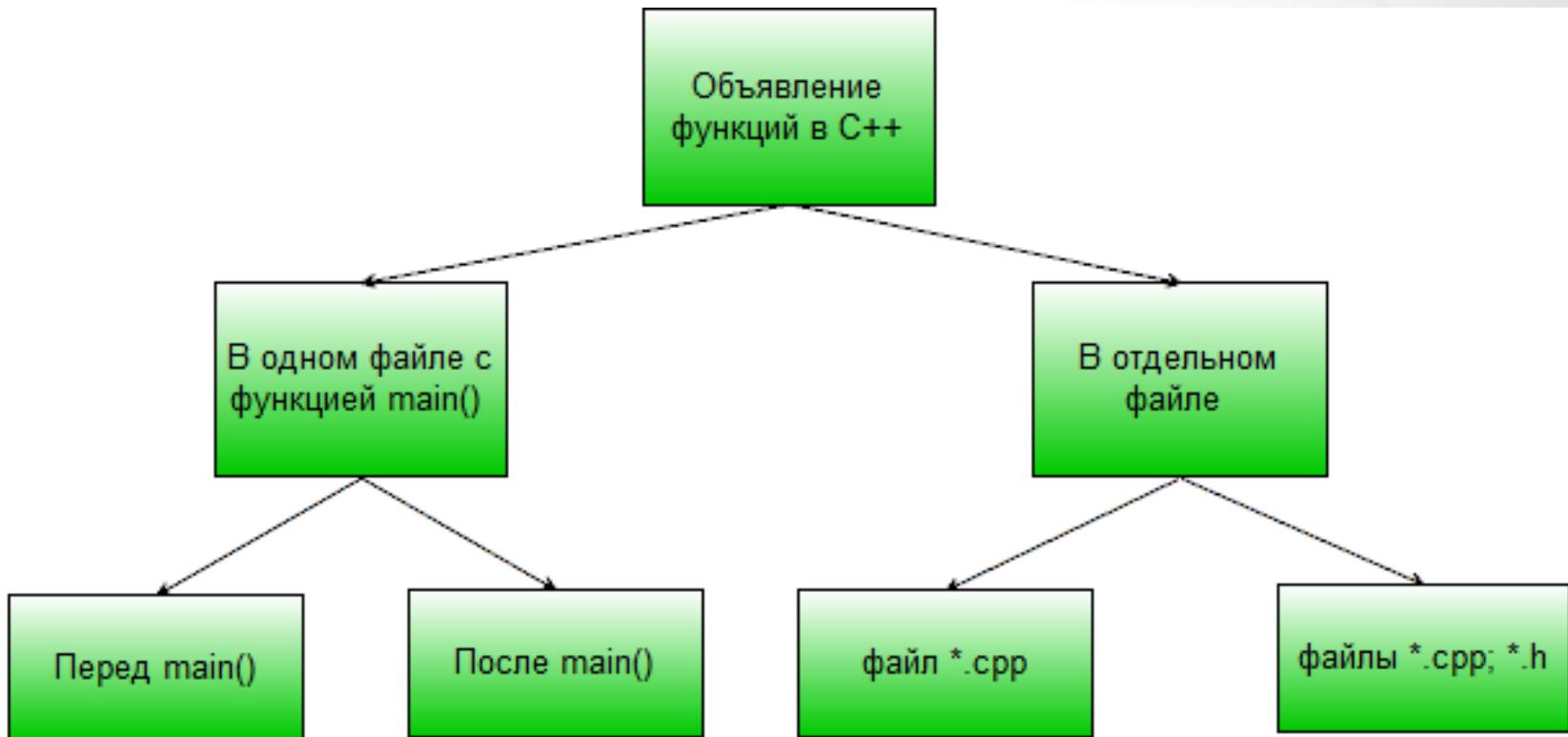


Демонстрация





Объявление функций в C/C++



ФУНКЦІИ В С/С++

- Функции можно объявлять в двух областях, до начала функции `main()`, после функции `main()`.
- Ранее мы объявляли функции в одном файле, перед функцией `main()` - это самый простой из способов.

```
/* область 1 - объявление функций до начала main()
   место для объявления функций
   функциям объявленным в этой области
   не нужны прототипы
*/
int main()
{
    return 0;
}
```



ФУНКЦІИ В С/С++

- Хорошему стилю программирования соответствует способ объявления функций после main().
- Рассмотрим структуру такого объявления функций.

```
// место для объявления прототипов функций

int main()
{
    return 0;
}

/* область 2 - объявление функций после main()
   место для объявления функций
*/
```

ФУНКЦІИ В С/С++

- Область объявления функций переместилась в самый конец программы, после `main()`.
- При этом способ объявления функций не поменялся.
- Но так как функции объявлены после `main()`, использовать их не получится, ведь порядок объявлений изменился и функция `main()` просто не будет видеть функции объявленные после.
- Чтобы эти функции можно было увидеть в `main()` существует понятие прототипа.
- Прототип функции — это заголовок функции, который объявляется перед функцией `main()`.
- И если объявить прототип функции, тогда функцию можно будет увидеть в `main()`.

ФУНКЦІИ В С/С++

```
// синтаксис объявления прототипа
/*тип возвращаемого значения*/ /*имя функции*/ /*параметры*/ ;
```

- Структура объявления прототипа очень схожа со структурой объявления функции.
- Разработаем программу, которая определяет, является ли введённое пятизначное число палиндромом.
- Палиндром — это число или текст, который читается одинаково как слева, так и справа: 93939; 49094; 11311.

Пример

```
#include <iostream>
using namespace std;

bool palindrome5(int) ; // прототип функции

int main()
{
    cout << "Enter 5digit number: ";
    int number;
    cin >> number;

    if (palindrome5(number))
        cout << "Number " << number
            << " - palindrome" << endl;
    else
        cout<<"This is not palindrome"<<endl;
    return 0;
}
```



Пример

```
.....
bool palindrome5(int number)
{
    int digit1, digit2, digit4, digit5;
    digit1 = number % 10;
    number = number / 10;

    digit2 = number % 10;
    number = number / 100;

    digit4 = number % 10;
    number = number / 10;

    digit5 = number % 10;
    if ((digit1 == digit5) && (digit2 == digit4))
        return true;
    else
        return false;
}
```

Описание функций в отдельных файлах

В C++ существует возможность поместить объявления функций в отдельный файл, тогда необходимо будет подключать файл с функциями, как в случае с подключением стандартных заголовочных файлов.

Есть два способа:

- создание файла типа *.cpp, в котором объявляются функции;
- создание файлов типа *.cpp и *.h.

Хорошим стилем программирования считается второй способ. Переделаем программу нахождения палиндрома так, чтобы объявление функции `palindrom5()` находилось в отдельном файле *.cpp.

Файл *.h нужен для того, чтобы скрыть реализацию функций.

Способи передачи параметров функций

В языке C++ есть 3 способа передачи параметров в функции:

- По значению
- По указателю
- По ссылке (только в C++)

Рассмотрим такой пример.

Необходимо разработать программу, которая вводит два числа, затем вызывает функцию, меняющую введенные числа местами, а затем выводит их.

Пример

Передача по значению (обычная передача параметров)

```
#include<iostream>
using namespace std;

void swap(int a, int b)
{
    int t;
    t=b;
    b=a;
    a=t;
}

int main()
{
    int p=3, q=5;
    swap(p, q);
    cout << p << " " << q << endl;
    return 0;
}
```

Результат: 3 5

Демонстрация



Способи передачи параметров функций

- При вызове функции swap создаются новые переменные a и b, им присваиваются значения 3 и 5.
- Эти переменные никак не связаны с переменными р и q и их изменение не изменяет значения р и q.
- Такой способ передачи параметров называется передачей параметров по значению.
- Чтобы функция могла изменять значения переменных, объявленных в других функциях, необходимо указать, что передаваемый параметр является не просто константной величиной, а переменной, необходимо передавать значения по ссылке.
- Для этого функцию swap следовало бы объявить следующим образом:

```
void swap(int & a, int & b)
```

Пример

Передача по ссылке – только C++

```
#include<iostream>
using namespace std;

void swap(int & a, int & b)
{
    int t;
    t=b;
    b=a;
    a=t;
}

int main()
{
    int p=3, q=5;
    swap(p, q);
    cout << p << " " << q << endl;
    return 0;
}
```

Результат: 5 3



Демонстрация

ПОСМОТРИМ

КОД



Способи передачи параметров функций

void swap(int & a, int & b)

- Амперсанды перед именем переменной означают, что эта переменная является не локальной переменной, а ссылкой на переменную, указанную в качестве параметра при вызове функции.
- Теперь при вызове **swap (p, q)** переменные a и b являются синонимами для переменных p и q, и изменение их значений влечет изменение значений p и q.
- А вот вызывать функцию в виде **swap(3,5)** уже нельзя, поскольку 3 и 5 — это константы, и сделать переменные синонимами констант нельзя.

Способи передачи параметров функций

- Однако в языке С (не C++) вообще не было такого понятия, как передача параметров по ссылке.
- Для того, чтобы реализовать функцию, аналогичную swap в рассмотренном примере, необходимо было передавать адреса переменных р и q, а сама функция при этом должна быть объявлена, как

```
void swap(int * a, int * b)
```

- Поскольку в этом случае функция swap знает физические адреса в оперативной памяти переменных р и q, то разыменовав эти адреса функция swap сможет изменить значения самих переменных р и q.

Пример

Передача по ссылке – только C++

```
#include<iostream>
using namespace std;

void swap(int * a, int * b)
{
    int t;
    t = *b;
    *b = *a;
    *a = t;
}

int main()
{
    int p=3, q=5;
    swap(&p, &q);
    cout << p << " " << q << endl;
    return 0;
}
```

Результат: 5 3

Демонстрация



Передача массива, как параметра функции

- Вспомним, что в C++ массивы и указатели — это почти одно и то же.
- Поскольку передача массива по значению, то есть создание копии всего массива является трудоемкой операцией (особенно для больших массивов), то вместо массива всегда передается указатель на его начало.
- То есть два объявления функции
 - `void f(int A[10])` и `void f(int * A)` абсолютно идентичны.
- В любом случае функция `f` получит указатель на начало массива, а, значит, будет способна изменять значения его элементов.



Передача массива, как параметра функции

Рассмотрим пример:

- Разработать функцию, нахождения номера минимального элемента в массиве.

Передача массива, как параметра функции

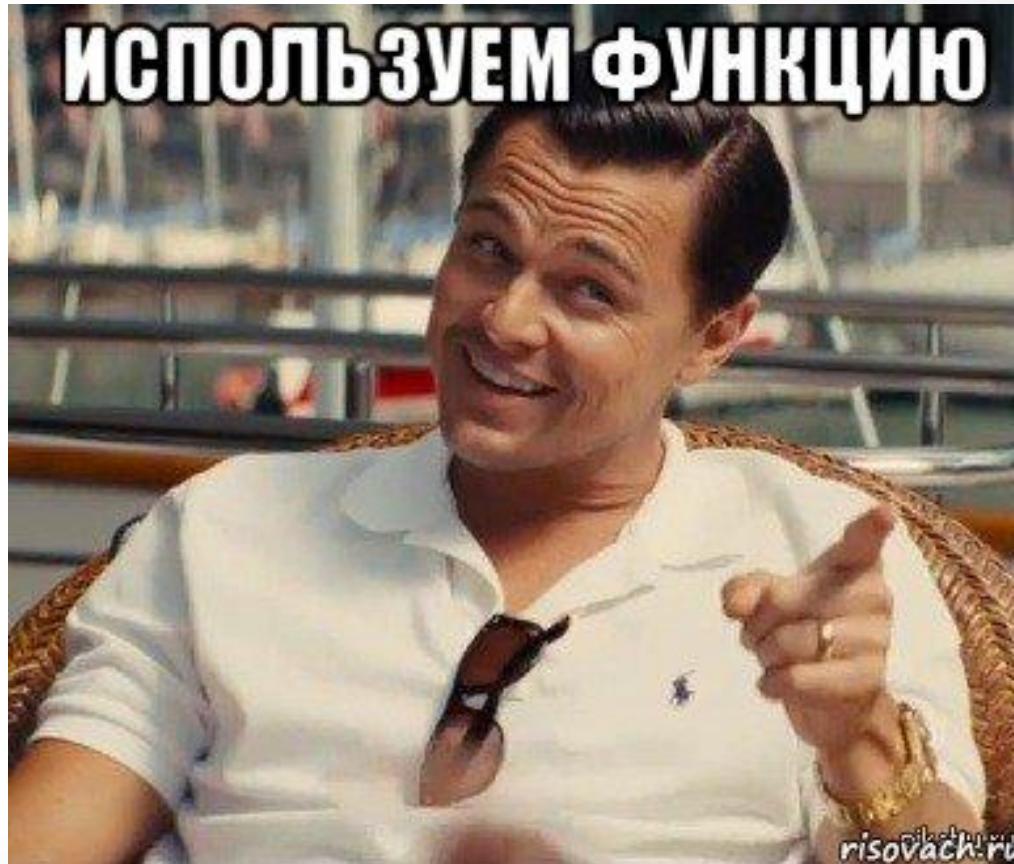
```
#include <iostream>
using namespace std;

int nmin(int *a, int size)
{
    int result = 0;
    for (int i=1; i<size; i++) {
        if (a[i]<a[result]) result = i;
    }
    return result;
}
int main()
{
    int arr[]={7, 5, -2, 4, 1, 0, -10, 8, 3};
    int num = nmin(arr, 10);
    int min = arr[num];
    cout << "min=" << arr[num]
        << " num=" << num << endl;
    return 0;
}
```



НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ
КОРАБЛЕБУДУВАННЯ
ІМЕНІ АДМІРАЛА МАКАРОВА

Демонстрация



Аргументы функций по умолчанию

- При обращении к функции, можно опускать некоторые её аргументы.
- Для этого необходимо при объявлении прототипа данной функции проинициализировать её параметры какими-то значениями, эти значения и будут использоваться в функции по умолчанию.
- Аргументы по умолчанию должны быть заданы в прототипе функции.
- Если в функции несколько параметров, то параметры, которые опускаются должны находиться правее остальных.
- Таким образом, если опускается какой-то параметр, то все параметры, расположенные перед ним могут не опускаться, но после него они должны быть опущены.

Аргументы функций по умолчанию

```
#include <iostream>
#include <cmath>
using namespace std;

double heron(double a = 5, double b = 6.5, double c = 10.7);

int main()
{
    cout << "S = " << heron()          << endl << endl;
    cout << "S = " << heron(10,5)    << endl << endl;
    cout << "S = " << heron(7)       << endl << endl;
    return 0;
}

double heron(double a, double b, double c)
{
    double p = (a + b + c) / 2;
    cout <<"a= "<< a <<"\nb= "<< b << "\nc = "<< c << endl;
    return (sqrt(p * (p - a) * (p - b) * (p - c)));
}
```

Демонстрация



Весь код по ссылке: <http://code.re/5Mz>

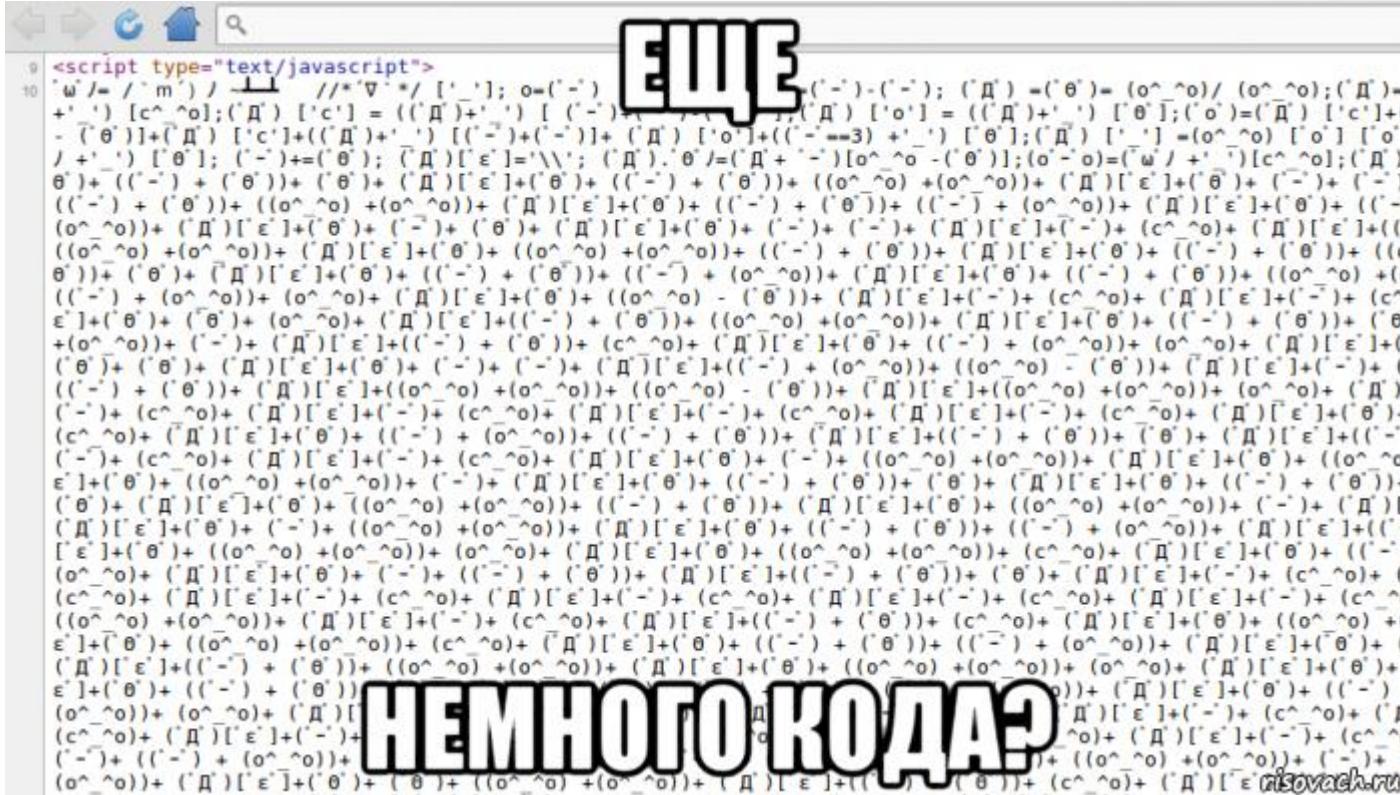
Пример задачи

Даны действительные числа s, t . Получить

$$f(t, -2s, 1.17) + f(2.2, t, s - t),$$

где $f(a, b, c) = \frac{2a - b - \sin c}{5 + |c|}$.

Демонстрация



Весь код по ссылке: <http://code.re/6MR>

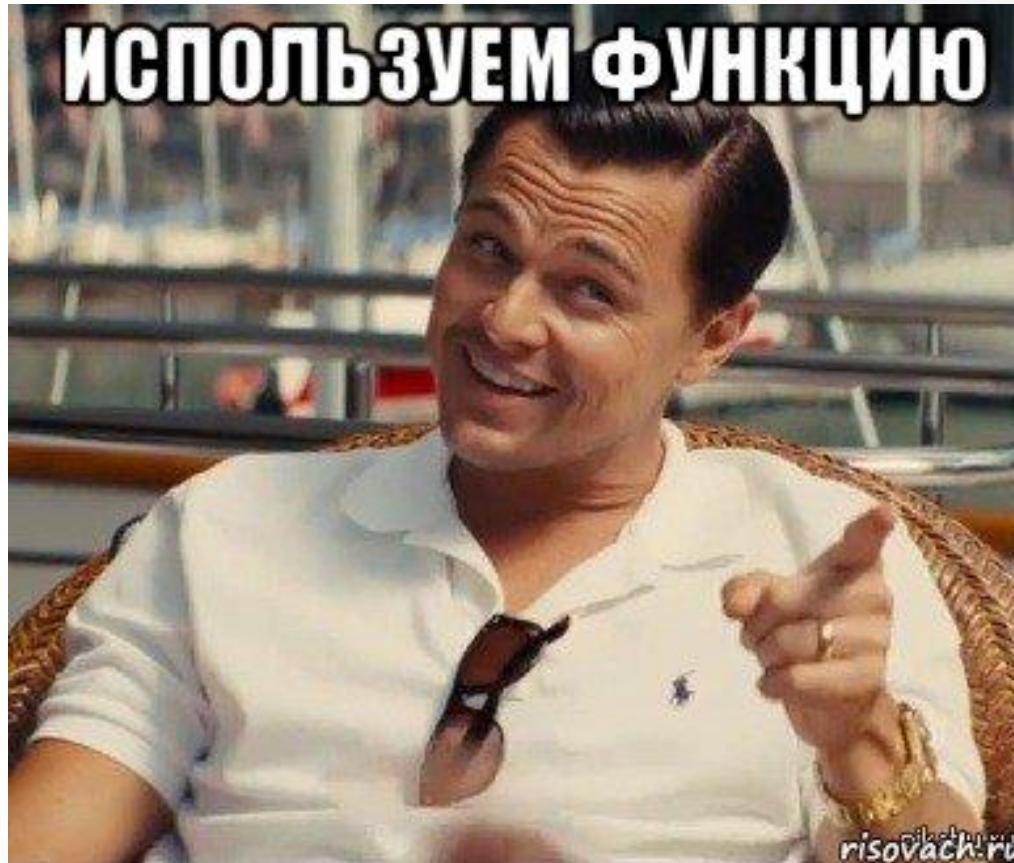
Пример задачи

Дано действительное число y . Получить

$$\frac{1.7t(0.25) + 2t(1+y)}{6 - t(y^2 - 1)}, \text{ где } t(x) = \frac{\sum_{k=0}^{10} \frac{x^{2k+1}}{(2k+1)!}}{\sum_{k=0}^{10} \frac{x^{2k}}{(2k)!}}$$



Демонстрация



Весь код по ссылке: <http://code.re/6MT>



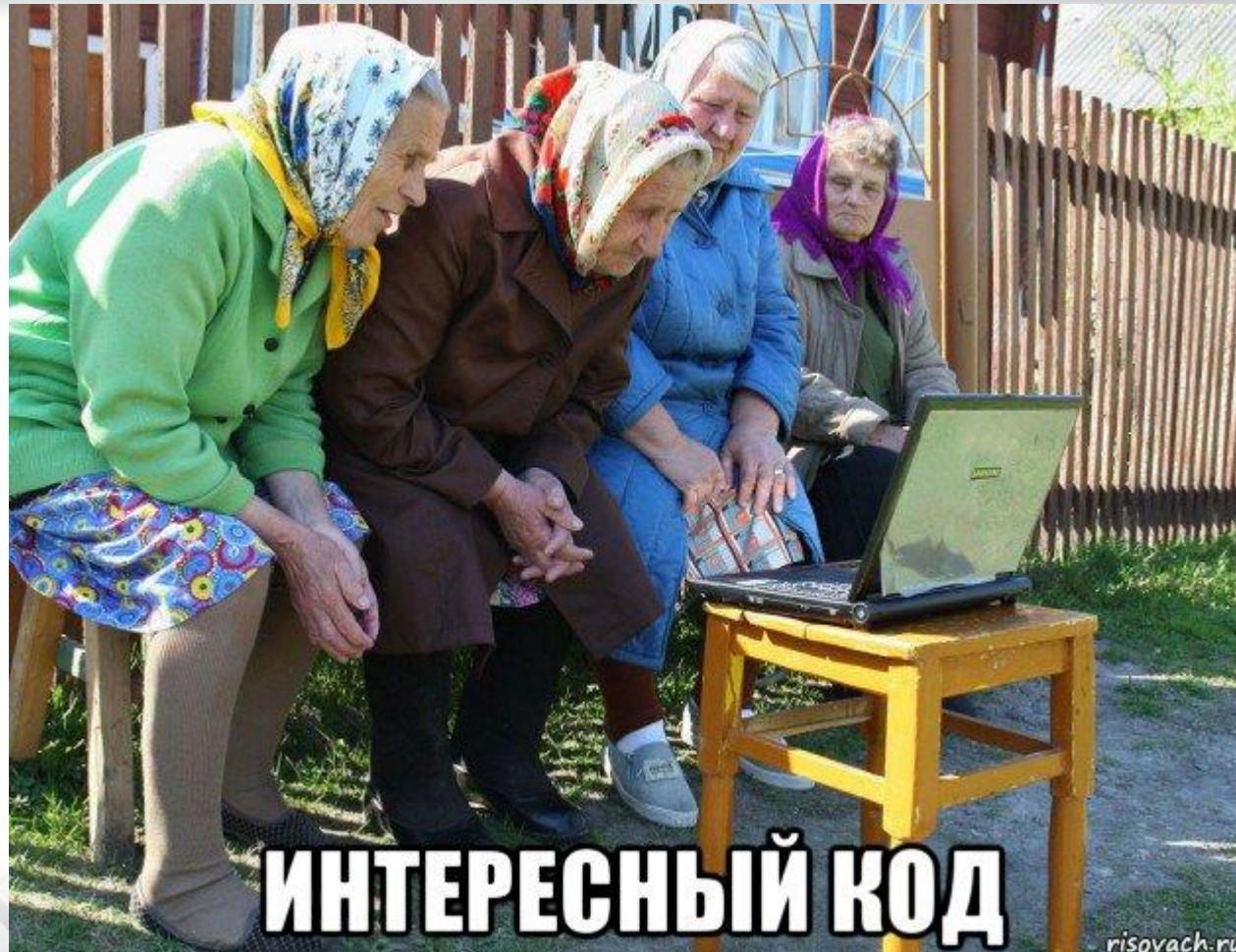
Пример задачи

Даны действительные числа a, b, c . Получить

$$\frac{\max(a, a+b) + \max(a, b+c)}{1 + \max(a+bc, 1, 15)}.$$



Демонстрация



Весь код по ссылке: <http://code.re/6MU>



НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ
КОРАБЛЕБУДУВАННЯ
ІМЕНІ АДМІРАЛА МАКАРОВА

QUESTIONS & ANSWERS

A close-up photograph of a white computer keyboard. The focus is on a few keys: the 'P' key, the 'R' key, and the '?' key. The '?' key is prominently displayed in the center. The background is blurred, showing other keys of the keyboard.

Алгоритмизация и программирование

Программирование на С/С++
(ч.7 – функции)



#include <stdio.h>
int main(void)
{
 printf("Hello, World!\n");
}
C/C++

Беркунский Е.Ю., кафедра ИУСТ, НУК
eugeny.berkunsky@gmail.com
<http://www.berkut.mk.ua>