**Кто подготовил материал:** Благодарный Артём

**Источники:**

1. [**https://radioprog.ru/post/1196**](https://radioprog.ru/post/1196)
2. **https://metanit.com/cpp/tutorial/4.8.php**
3. Лекции по С++ Казанцевой О.Г.:

конспект на лекции

**Вопрос 27.** Указатель на функцию; std::function.

**Указатель на функцию** (*function pointer*) хранит адрес функции, а именно содержит адрес первого байта в памяти, по которому располагается выполняемый код функции.

 Самым распространенным указателем на функцию является ее имя. С помощью имени функции можно вызывать ее и получать результат ее работы. Но также указатель на функцию мы можем определять в виде отдельной переменной с помощью следующего синтаксиса:

Здесь “тип’’ представляет тип возвращаемого функцией значения.

“имя\_указателя” представляет произвольно выбранный идентификатор в соответствии с правилами о наименовании переменных.

И “параметры” определяют тип и название параметров через запятую при их наличии.

Например, определим указатель на функцию:

В данном случае определен указатель, который имеет имя **message**. Он может указывать на функции без параметров, которые возвращают тип **void** (то есть ничего не возвращают).

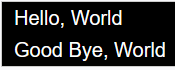
Используем указатель на функцию:

Указателю на функцию можно присвоить функцию, которая соответствует указателю по возвращаемому типу и спецификации параметров:

В качестве альтернативы мы можем обращаться к указателю на функцию следующим образом:



В итоге результатом данной программы будет следующий вывод:



**ВАЖНО!!!** При определении указателя стоит обратить внимание на скобки вокруг имени. Так, использованное выше определение:



НЕ будет аналогично следующему определению:

**Массивы указателей на функции**

Кроме одиночных указателей на функции мы можем определять их массивы. Для этого используется следующий формальный синтаксис:

Например:

Здесь **actions** представляет массив указателей на функции, каждая из которых обязательно должна принимать два параметра типа **int** и возвращать значение типа **double**.

Посмотрим применение массива указателей на функции на примере:

#include <iostream>

void add ( int, int);

void subtract ( int, int);

void multiply ( int, int);

int main(){

    int a = 10;

    int b = 5;

    void (\*operations[3] )( int, int) = {add, subtract, multiply};

    // получаем длину массива

    int length = sizeof( operations ) / sizeof( operations[0] );

    for ( int i = 0; i < length; i++){

        operations[i](a, b);    // вызов функции по указателю

    }

    return 0;

}

void add (int x, int y){

    std::cout << "x + y = " << x + y << std::endl;

}

void subtract (int x, int y){

    int result = x - y;

    std::cout << "x - y = " << x - y << std::endl;

}

void multiply(int x, int y){

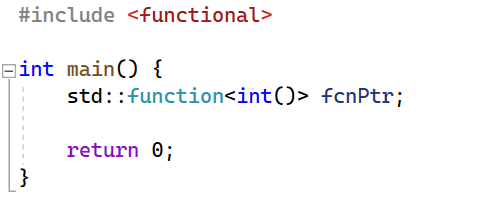
    std::cout << "x \* y = " << x \* y << std::endl;

}

Здесь массив **operations** содержит три функции **add**, **subtract** и **multiply**, которые последовательно вызываются в цикле через перебор массива в функции **main**.

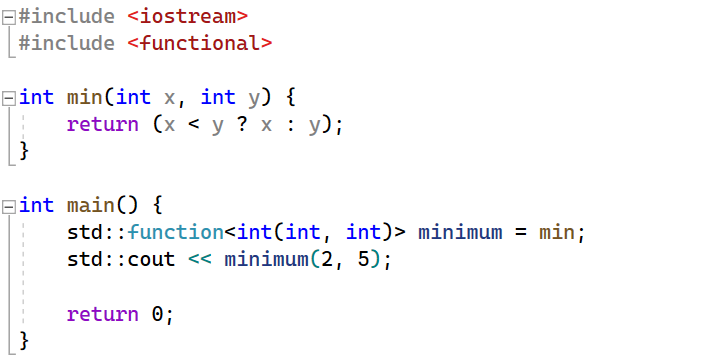
Консольный вывод программы:

В C++11 был введён альтернативный способ определения и хранения указателей на функции, который выполняется с использованием **std::function**. std::function является частью заголовочного файла *functional* стандартной библиотеки C++. Для определения указателя на функцию с помощью этого способа нужно объявить объект std::function следующим образом:



Как видно, тип возврата и параметры находятся в угловых скобках, а параметры ещё и внутри круглых скобок. Если параметров нет, то внутренние скобки можно оставить пустыми.

Обновим код нашей программы:



Таким образом, указатели на функции полезны, прежде всего, когда необходимо хранить функции в массиве (или в структуре) или когда требуется передать одну функцию в качестве аргумента другой функции. Поскольку синтаксис объявления указателей на функции несколько подвержен ошибкам, то рекомендуется использовать std::function.

**Итоги:**

1. **Термины**: указатель на функцию; std::function.
2. **Code snippets: Вызов функции через указатель на функцию**

int boo(int a){

return a;

}

int main(){

int (\*fcnPtr)(int) = boo; // присваиваем функцию boo() указателю fcnPtr

(\*fcnPtr)(7); // вызываем функцию boo(7), используя fcnPtr

return 0;

}

1. **Контрольные вопросы:**
2. Каким синтаксисом вызывается функция int boo(int a)?
3. Какую библиотеку нужно подключить перед использованием std::function?
4. **Тестовые вопросы:**

* Выберите правильные варианты синтаксиса указателя на функцию:

1. boo(arg); !
2. (\*fcnPtr)(int) = boo; !

(\*fcnPtr)(7);

3) boo (&arg);

4) boo (\*arg);

* В какой версии С++ был введён способ определения и хранения указателей на функции с помощью std::function?

1. С++23;
2. С++17;
3. С++98;
4. С++11; !