Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 РЕФЕРАТ

По дисциплине «Основы алгоритмизации и программирования»

На тему «Отличие перегрузки функции от переопределения»

Выполнила:

Студентка1 курса 7 группы

Гриценко Анна Александровна

2023, Минск

Оглавление

[Перегрузка функции(overload) 3](#_Toc158306373)

[Переопределение функции(override) 3](#_Toc158306374)

[Вывод: отличие перегрузки функции от переопределения 4](#_Toc158306375)

[Литература 6](#_Toc158306376)

# Перегрузка функции(overload)

Перегрузка функций позволяет определить несколько функций с одним и тем же именем, но различающимися параметрами (типами параметров или их количеством).

Перегрузка функции— это механизм полиморфизма, который позволяет задавать несколько функций с одинаковым именем, но с разными параметрами в одном классе. Компилятор выбирает наиболее подходящую функцию для вызова на основе типов аргументов, передаваемых при вызове функции.

**Пример кода:**

#include <iostream>

using namespace std;

void print(int a) {//создаем функцию для данных типа int

cout << "Parent class" << endl;

};

void print(double w) {//создаем функцию для данных типа double

cout << "Derived class" << endl;

};

int main() {

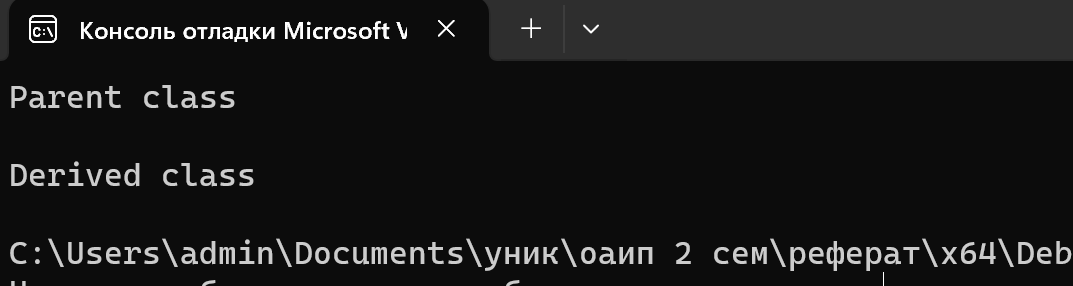
print(4);

cout << "\n";

print(0.6);

return 0;

}



# Переопределение функции(override)

Переопределение функции происходит, когда в производном классе предоставляется новая реализация уже существующей функции базового класса. В контексте функций в классах это также называется виртуальной функцией.

Переопределение функции — это механизм полиморфизма, который позволяет создать классы-наследники для родительских классов для того, чтобы создать несколько реализаций функции и использовать их в программе. Для переопределения функции в классе-наследнике, необходимо использовать такое же самое имя функции и типы ее аргументов и указать новую реализацию. При вызове функции на объекте класса-наследника, будет вызвана его переопределенная реализация, а не реализация из родительского класса.

**Пример кода:**

#include <iostream>

using namespace std;

class Base {//объявляем базовый класс и в его теле пишем виртуальную функцию

public:

virtual void print() {

cout << "Parent class" << endl;

}

};

class Derived : public Base {//создаем класс-наследник и переопределяем в теле этого класса реализацию функции print()

public:

void print() override {

cout << "Derived class" << endl;

}

};

int main() {

Base elementOne;

Derived elementTwo;

elementOne.print();// вызывается функция print() из базового класса

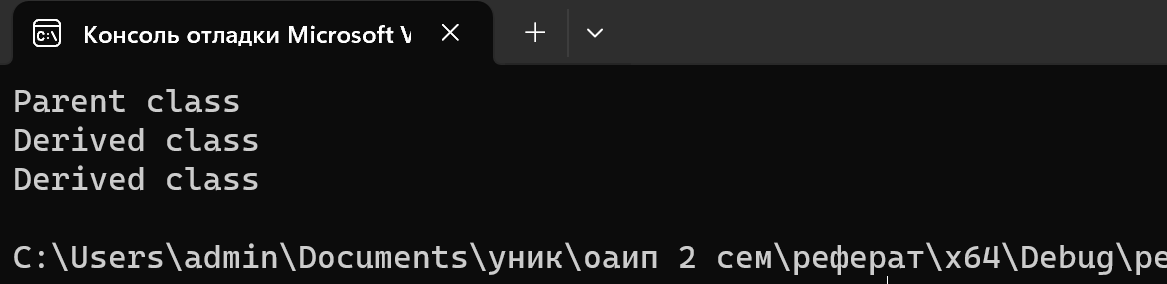
elementTwo.print();// вызывается переопределенная функция print() из наследника базового класса

Base \*ptr = &elementTwo;//Объявляем указатель на базовый класс

(\*ptr).print(); // вызывается переопределенная функция print() из наследника базового класса

return 0;

}



# Вывод: отличие перегрузки функции от переопределения

1. Воспользоваться перегрузкой функции мы можем внутри одного класса, а для переопределения нужно создать два и более классов.
2. При использовании перегрузки подбор соответствующих параметров происходит в этапе компиляции программы.( полиморфизм времени компиляции). Переопределение выполняется во время выполнения ( полиморфизм времени выполнения).
3. Возможно изменение возвращаемого значения функции при использовании перегрузки функции, но при использовании переопределения это невозможно.
4. Перегрузка функции упрощает программу. А вот переопределением функции пользуются, если не устраивает значения родительского класса.

# Литература

* [**https://uchet-jkh.ru/i/cem-peregruzka-metodov-otlicaetsya-ot-pereopredeleniya-metodov/**](https://uchet-jkh.ru/i/cem-peregruzka-metodov-otlicaetsya-ot-pereopredeleniya-metodov/)
* [**https://learn.microsoft.com/ru-ru/cpp/cpp/function-overloading?view=msvc-170**](https://learn.microsoft.com/ru-ru/cpp/cpp/function-overloading?view=msvc-170)
* [**https://askanydifference.com/ru/difference-between-overloading-and-overriding/**](https://askanydifference.com/ru/difference-between-overloading-and-overriding/)