Лекция 3. Функции и перегрузка функций

Казахский национальный университет имени аль-Фараби

Кафедра компьютерных наук

Объектно-ориентированное программирование

Лектор: старший преподаватель, PhD, Кенжебек Ержан Ғалымжанұлы

Цель лекции

Цель лекции

Изучить назначение и устройство функций в C++, способы передачи аргументов, рекурсию и перегрузку функций.

Ожидаемые результаты

1. Описывать функцию: прототип, вызов, заголовок и тело
2. Выбирать способ передачи аргументов: по значению или по ссылке
3. Записывать простые рекурсивные функции
4. Применять перегрузку функций

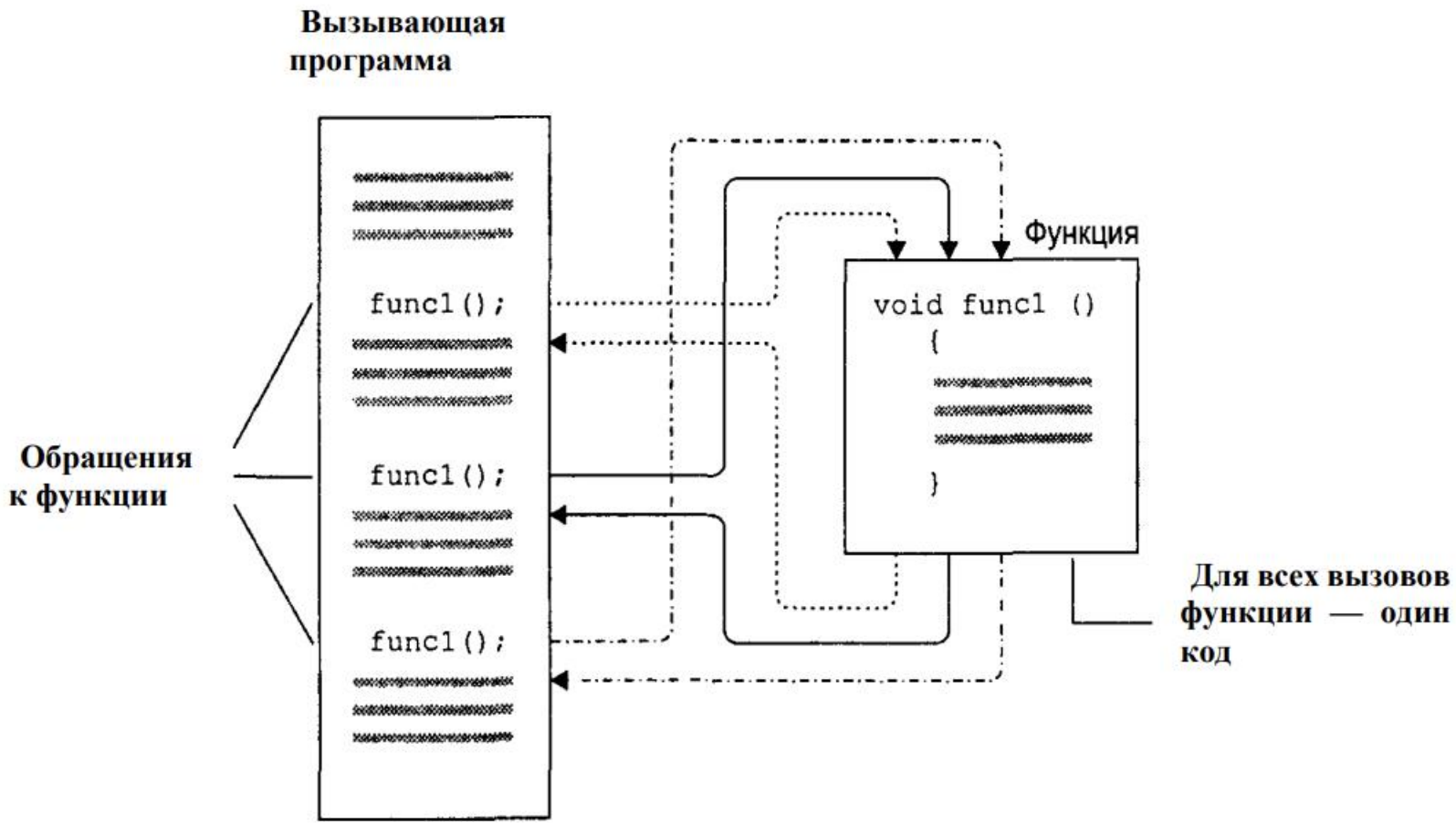
План лекции

- Функция
- Функциональные компоненты
- Возвращаемое значение функции
- Передача аргументов по значению
- Ссылки на аргументы
- Перегруженные функции

Зачем нам нужны функции?

- Функции помогают нам уменьшить избыточность кода. Если функциональность выполняется в нескольких местах программы, то вместо того, чтобы снова и снова писать один и тот же код, мы создаем функцию и вызываем ее везде. Это также помогает в обслуживании, так как нам нужно измениться в одном месте, если мы будем вносить изменения в функциональность в будущем.
- Функции делают код модульным. Рассмотрим большой файл с большим количеством строк кода. Читать и использовать код становится действительно легко, когда он разбит на функции.
- Функции обеспечивают абстракцию. Например, мы можем использовать библиотечные функции, не беспокоясь об их внутренней работе.

Функции


Вызывающая программа
func1();
func1();
func1();
Обращения к функции
Функция
void func1 ()
{
}
Для всех вызовов функции — один код

Компоненты функции

- Объявления функций (прототип)
- Вызов функции
- Заголовок и определение функции

```
#include<iostream.h>
int add(int,int);                    //function prototype

void main()
{
   int x,y,z;
   cout<<"Enter a number: ";
   cin>>x;
   cout<<"Enter another number: ";
   cin>>y;
   z=add(x,y);                       //function call
   cout<<z<<endl;
}
int add(int a,int b)                 //function definition
{
   return (a+b);
}
```

Возвращаемое значение функции

- Тип возвращаемого значения недействителен,и оператор возврата других типов

Передача аргументов по значению

```
int sum(int a, int  b){
return a+b;
}
```

Передача по ссылке

- Мы уже видели несколько примеров передачи аргументов функциям по значению. При передаче по значению вызываемая функция создает новые переменные того же типа, что и переданные аргументы, и копирует значения аргументов в эти переменные. Как мы видели, функция не имеет доступа к переменным-аргументам, но работает с копиями сделанных ей значений. Конечно, такой механизм полезен в тех случаях, когда функции не нужно менять значения аргументов, и мы защищаем аргументы от несанкционированного доступа.
- Передача аргументов по ссылке происходит по другому механизму. Вместо того, чтобы передать в функцию значение переменной, в функцию передается ссылка на эту переменную (на самом деле в функцию передается адрес переменной-аргумента в памяти, но эта деталь для нас не так важна, поэтому далеко).
- Важной особенностью передачи аргументов по ссылке является то, что функция имеет прямой доступ к значениям аргументов. К достоинствам ссылочного механизма можно отнести также возможность функции возвращать в программу не одно, а множество значений.ПРИМЕР

Передача по значению и по ссылке

По значению

- Функция работает с копией аргумента
- Изменения внутри функции не влияют на исходную переменную
- Копирование больших объектов требует времени и памяти

```
void inc(int a) { a++; }
// inc(x): x не изменится
```

По ссылке

- Параметр — псевдоним переменной
- Функция изменяет исходное значение
- Копирование не выполняется
- const int& — чтение без копирования

```
void inc(int& a) { a++; }
// inc(x): x увеличится на 1
```

Рекурсия

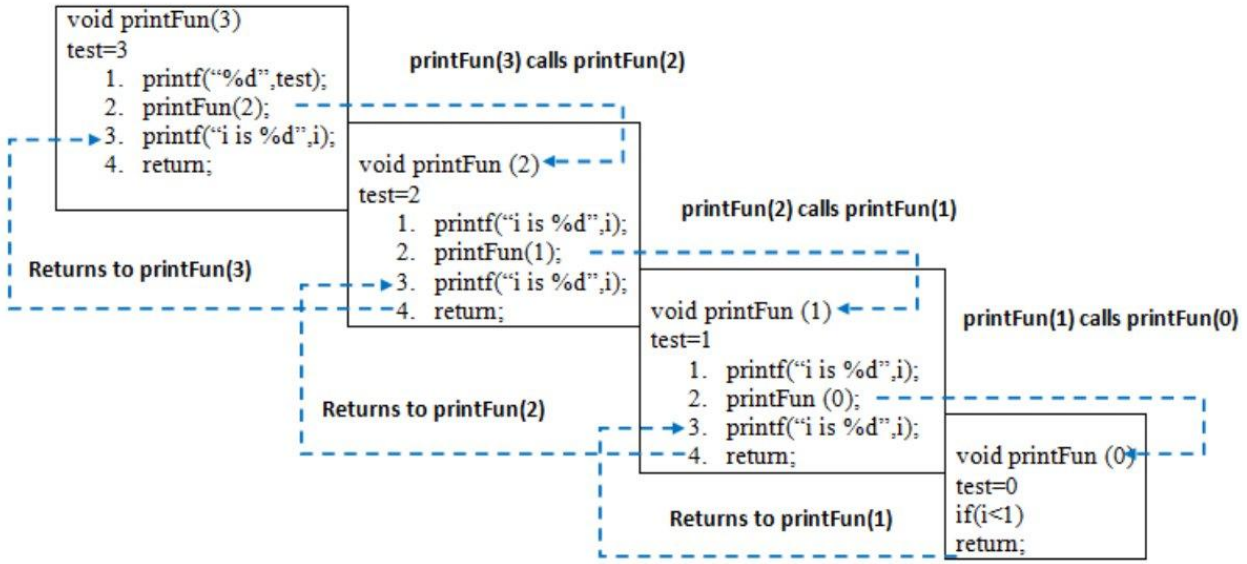
- Процесс, в котором функция прямо или косвенно вызывает саму себя, называется рекурсией, а соответствующая функция называется рекурсивной функцией.
- Рекурсия — удивительная техника, с помощью которой мы можем уменьшить длину нашего кода и упростить его чтение и запись. Он имеет определенные преимущества перед итерационным методом, о котором будет сказано ниже. Задача, которую можно определить с помощью аналогичной подзадачи, рекурсия — одно из лучших решений для нее. Например; Факториал числа.

Пример рекурсии

```
// A C++ program to demonstrate working of
// recursion
#include <bits/stdc++.h>
using namespace std;

void printFun(int test)
{
    if (test < 1)
        return;
    else {
        cout << test << " ";
        printFun(test - 1); // statement 2
        cout << test << " ";
        return;
    }
}

// Driver Code
int main()
{
    int test = 3;
    printFun(test);
}
```



Output :

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

Перегрузка функции

- Перегрузка функций — это возможность определить несколько функций с одним и тем же именем, но с разными параметрами. Например:

```
int subtract(int a, int b)
{
    return a - b;
}
```

```
int subtractInteger(int a, int b)
{
    return a - b;
}

double subtractDouble(double a, double b)
{
    return a - b;
}
```

Пример перегрузки функции

```
#include<iostream.h>
int add(int,int);                    //first prototype
int add(int,int,int);                //second prototype

void main()
{
   int x,y;
   x=add(10,20);                     //matches first prototype
   y=add(30,40,50);                  //matches second prototype
   cout<<x<<endl<<y<<endl;
}

int add(int a,int b)
{
   return(a+b);
}

int add(int a,int b,int c)
{
   return(a+b+c);
}
```

Выводы

1. Функции уменьшают избыточность кода, делают программу модульной и обеспечивают абстракцию.
2. Функция состоит из объявления (прототипа), вызова, заголовка и тела; тип void означает отсутствие возвращаемого значения.
3. При передаче по значению функция работает с копией, при передаче по ссылке — напрямую с исходной переменной.
4. Рекурсия — вызов функцией самой себя; обязательны базовый случай и шаг, приближающий к нему.
5. Перегрузка позволяет определить несколько функций с одним именем, но разными параметрами.

Контрольные вопросы

1. Зачем нужны функции? Назовите их преимущества.
2. Из каких компонентов состоит функция?
3. Чем передача аргументов по значению отличается от передачи по ссылке?
4. Что такое рекурсия? Для чего нужен базовый случай?
5. Что такое перегрузка функций? По каким признакам компилятор выбирает нужную функцию?
6. Что выведет рекурсивная функция printFun(3) из примера лекции?

Список литературы

Основная литература

1. Лафоре Р. Объектно-ориентированное программирование в C++. 4-е изд. — СПб.: Питер, 2004.
2. Страуструп Б. Язык программирования C++. Специальное издание. — М.: Бином, 2011.
3. Липпман С., Лажойе Ж., Му Б. Язык программирования C++. Базовый курс. 5-е изд. — М.: Вильямс, 2014.

Дополнительная литература

4. Шилдт Г. Полный справочник по C++. 4-е изд. — М.: Вильямс, 2006.
5. Страуструп Б. Программирование: принципы и практика с использованием C++. 2-е изд. — М.: Вильямс, 2016.
6. Майерс С. Эффективный и современный C++. — М.: Вильямс, 2016.

Электронные ресурсы

7. Учебник по C++ (learncpp): www.learncpp.com
8. Справочник по C++ и стандартной библиотеке: en.cppreference.com

- Спасибо за внимание!