**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**КЫРГЫЗСКОЙ РЕСПУБЛИКИ**

**КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. И.Раззакова**

**ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

Кафедра: **Программное обеспечение компьютерных систем**

Курс «Объектно-ориентированное программирование»

**ОТЧЕТ**

**Лабораторная работа №1**

Выполнил: студент группы ПИ-3-21

Урманбетов Султан

Проверил: Мусабаев Э.Б.

**Бишкек 2024**

# **Задание №1**

**Постановка задачи:**

Возведение числа **n** в степень **p** – это умножение числа **n** на себя **p** раз.

Напишите функцию (не библиотечную) с именем **power** (), которая в качестве аргументов принимает значение типа **double** для **n** и значение типа **int** для **p** и возвращает значение типа **double**.

Для аргумента, соответствующего степени числа, задать значение по умолчанию, равное **2**, чтобы при отсутствии показателя степени при вызове функции число **n** возводилось в квадрат.

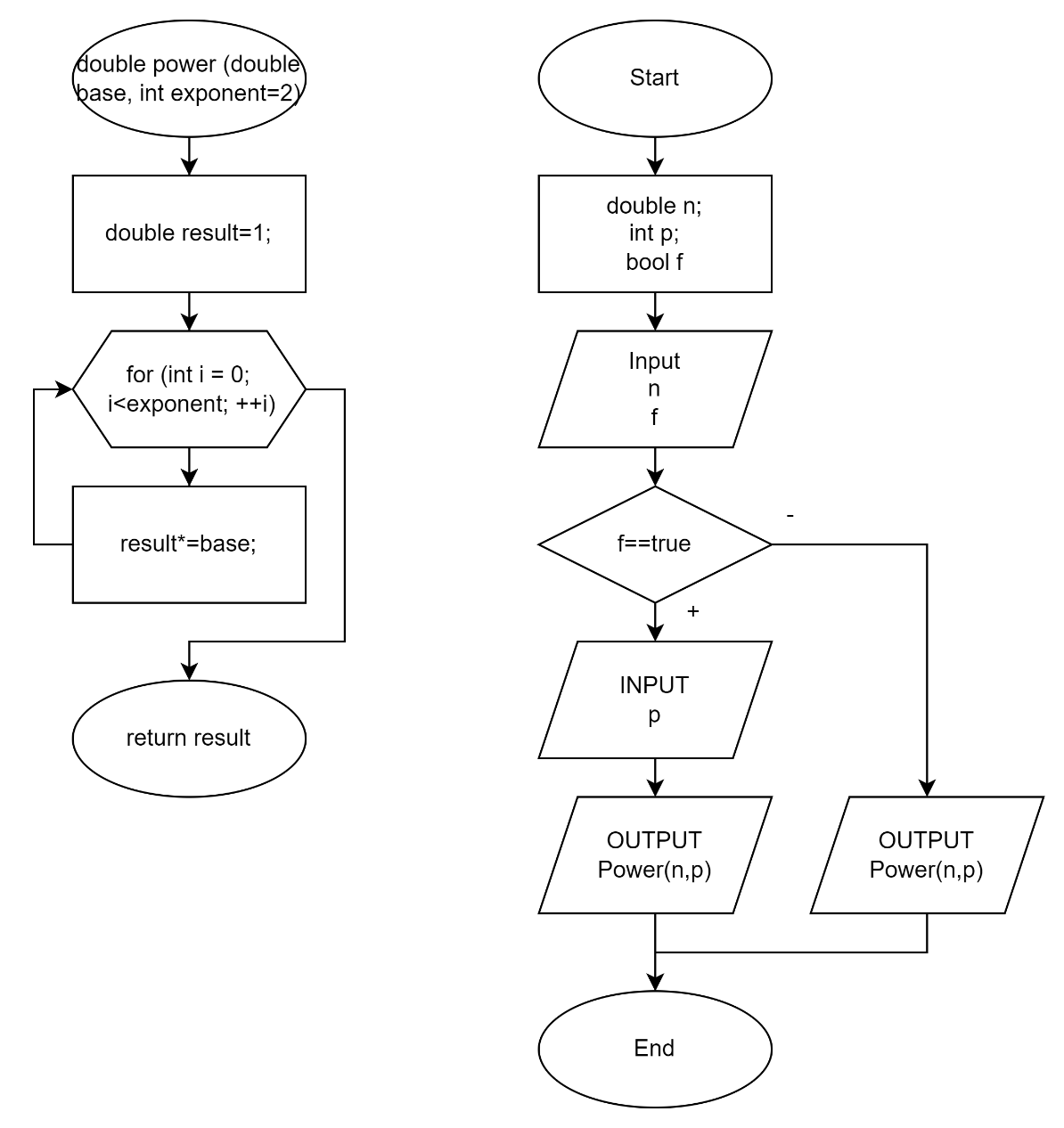
Сделайте проверку, будет ли пользователь вводить степень числа. Если будет, то эта степень – число, большее 2. Напишите функцию **main**(), которая запрашивает у пользователя ввод аргументов для функции **power** () и отображает на экране результаты её работы. Напишите код первого задания с *объявлением* (прототипом) функции power ().

**Исходные данные:**

n – base – основание

p – exponent – степень

**Блок-схемы:**

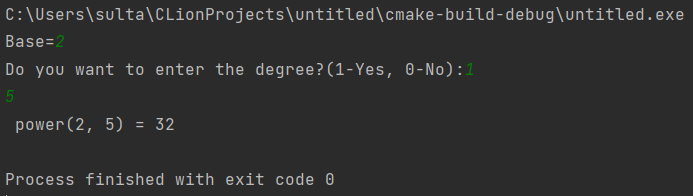


**Исходный код программы на C++:**

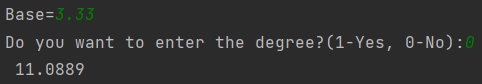
#include <bits/stdc++.h>  
using namespace std;  
  
double power(double base, int exponent=2) {  
 double result = 1.0;  
 for (int i = 0; i < exponent; ++i) {  
 result \*= base;  
 }  
 return result;  
}  
  
int main() {  
 double n;  
 int p;  
 cout<<"Base=";  
 cin >> n;  
 bool f;  
 cout<<"Do you want to enter the degree?(1-Yes, 0-No): ";  
 cin>>f;  
 if (f){  
 cin>>p;  
 cout << "power(" << n << ", " << p << ") = " << power(n, p) <<endl;  
 }  
 else  
 cout << power(n) << endl;  
  
  
 return 0;  
}

**Тесты:**

**Тест1**



**Тест2**



# **Задание №2**

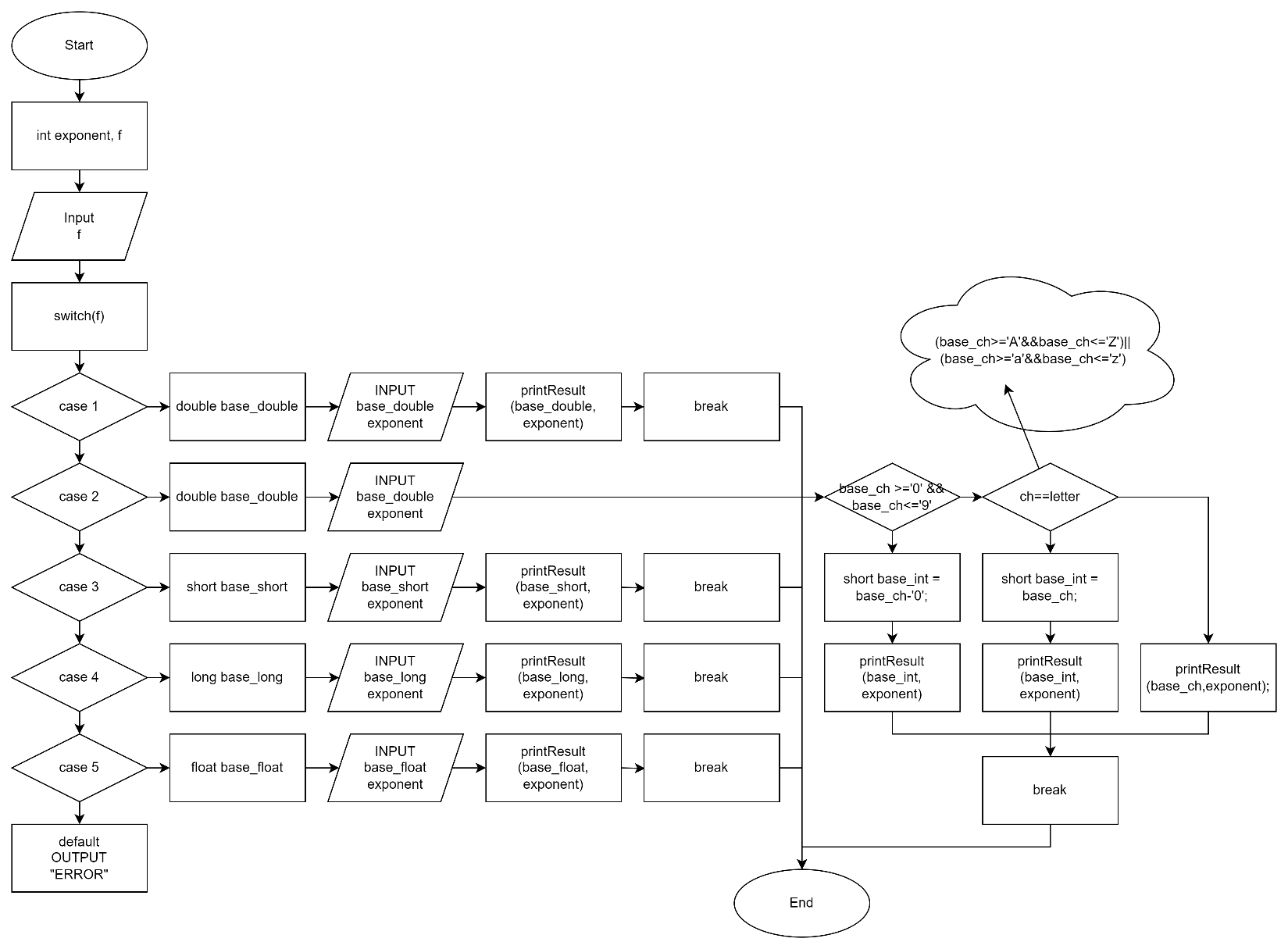
**Постановка задачи:**

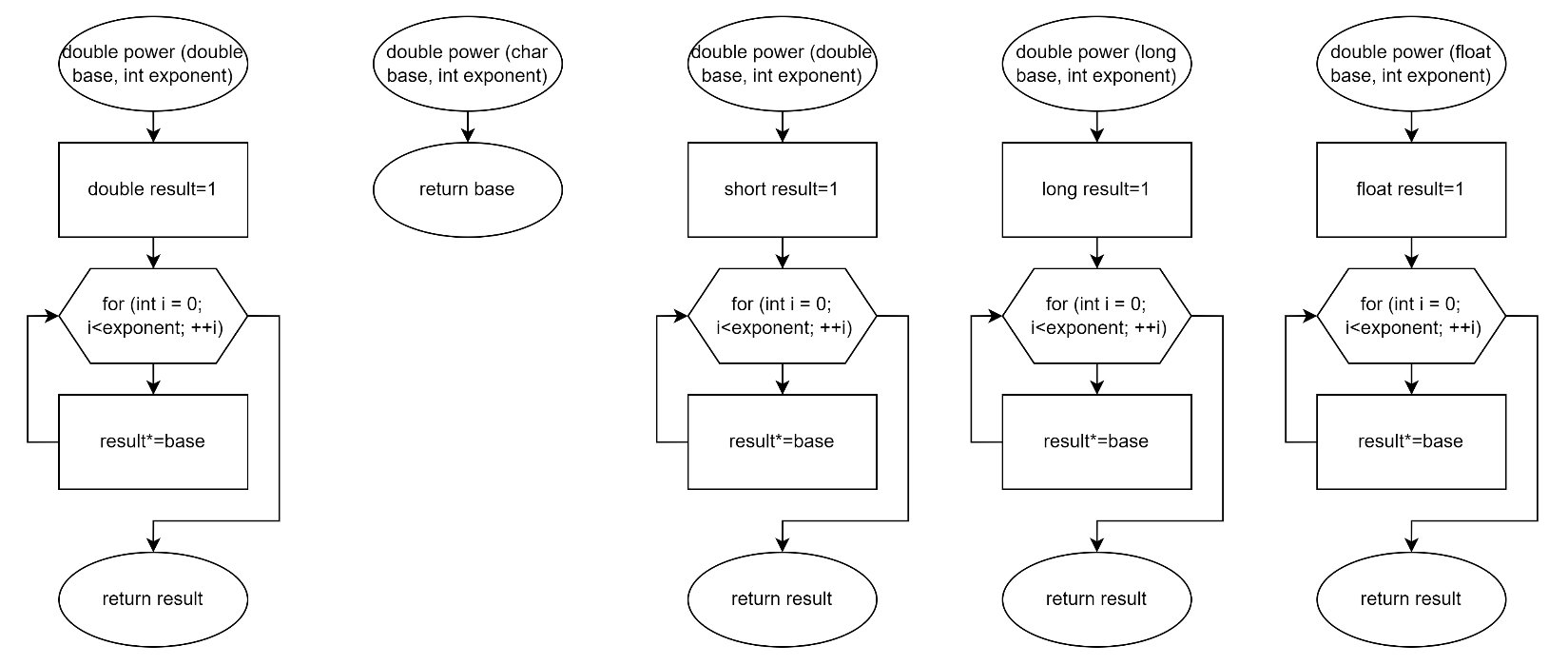
Взяв в качестве основы функцию **power** () из первого задания, работающую только со значением типа **double**. Создайте перегруженные функции с этим же именем, принимающими в качестве аргумента значения типа **char**, short **int**, **long int, float**. Напишите программу, вызывающую функцию **power** () со всеми возможными типами аргументов. Напишите код второго задания без *объявления* функции power ().

**Исходные данные:**

Double power (double base, int exponent) - функцию **power** () из первого задания, работающую только со значением типа **double.**

**Блок-схемы:**



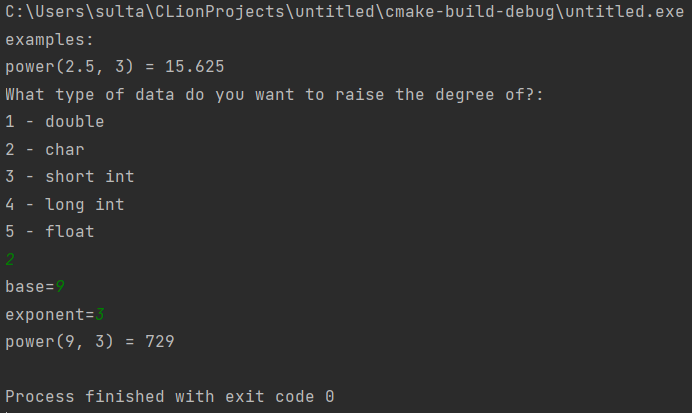


**Исходный код программы на C++:**

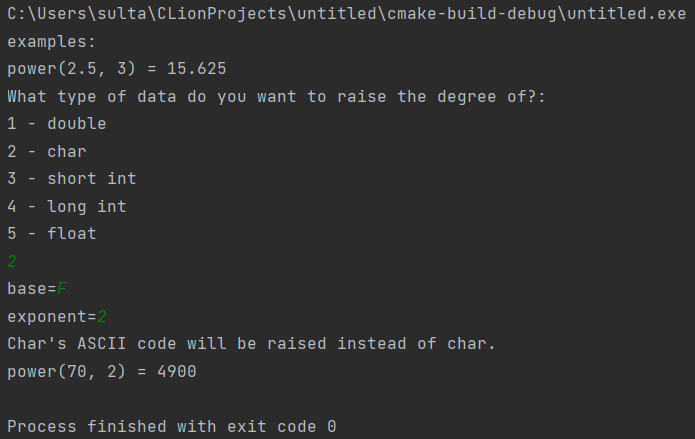
# #include <bits/stdc++.h>  
using namespace std;  
  
  
double power(double base, int exponent) {  
 double result = 1.0;  
 for (int i = 0; i < exponent; ++i) {  
 result \*= base;  
 }  
 return result;  
}  
  
char power(char base, int exponent) {  
 return base;  
}  
  
short power(short base, int exponent) {  
 short result = 1;  
 for (int i = 0; i < exponent; ++i) {  
 result \*= base;  
 }  
 return result;  
}  
  
long power(long base, int exponent) {  
 long result = 1;  
 for (int i = 0; i < exponent; ++i) {  
 result \*= base;  
 }  
 return result;  
}  
  
float power(float base, int exponent) {  
 float result = 1.0f;  
 for (int i = 0; i < exponent; ++i) {  
 result \*= base;  
 }  
 return result;  
}  
  
template<typename T>  
void printResult(T base, int exponent) {  
 cout << "power(" << base << ", " << exponent << ") = " << power(base, exponent) << std::endl;  
}  
  
int main() {  
 cout<<"examples:"<<endl;  
 printResult(2.5, 3);  
 int f, exponent;  
 cout<<"What type of data do you want to raise the degree of?:"<<endl;  
 cout<<"1 - double"<<endl;  
 cout<<"2 - char"<<endl;  
 cout<<"3 - short int "<<endl;  
 cout<<"4 - long int"<<endl;  
 cout<<"5 - float"<<endl;  
 cin>>f;  
 switch(f) {  
 case 1:  
 double base\_double;  
 cout<<"base=";cin>>base\_double;  
 cout<<"exponent=";cin>>exponent;  
 printResult(base\_double,exponent);  
 break;  
 case 2:  
 char base\_ch;  
 cout<<"base=";cin>>base\_ch;  
 cout<<"exponent=";cin>>exponent;  
 cout<<"Char's ASCII code will be raised instead of char."<<endl;  
 int result;  
 if (base\_ch >='0' && base\_ch<='9') {  
 short base\_int = base\_ch-'0';  
 printResult(base\_int,exponent);  
 }  
 else if ((base\_ch>='A'&&base\_ch<='Z')||(base\_ch>='a'&&base\_ch<='z')) {  
 short base\_int = base\_ch;  
 printResult(base\_int,exponent);  
 }  
 else  
 printResult(base\_ch,exponent);  
 break;  
 case 3:  
 short int base\_shortInt;  
 cout<<"base=";cin>>base\_shortInt;  
 cout<<"exponent=";cin>>exponent;  
 printResult(base\_shortInt,exponent);  
 break;  
 case 4:  
 long int base\_longInt;  
 cout<<"base=";cin>>base\_longInt;  
 cout<<"exponent=";cin>>exponent;  
 printResult(base\_longInt,exponent);  
 break;  
 case 5:  
 float base\_float;  
 cout<<"base=";cin>>base\_float;  
 cout<<"exponent=";cin>>exponent;  
 printResult(base\_float,exponent);  
 break;  
 default:  
 cout<<"error"<<endl;  
 }  
  
 return 0;  
}

**Тесты:**

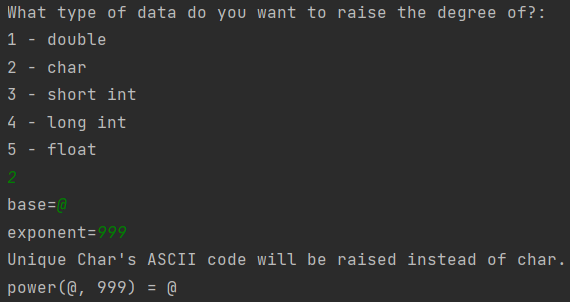
**Тест№1**



**Тест№2**



**Тест№3**



# **Задание №3**

**Постановка задачи:**

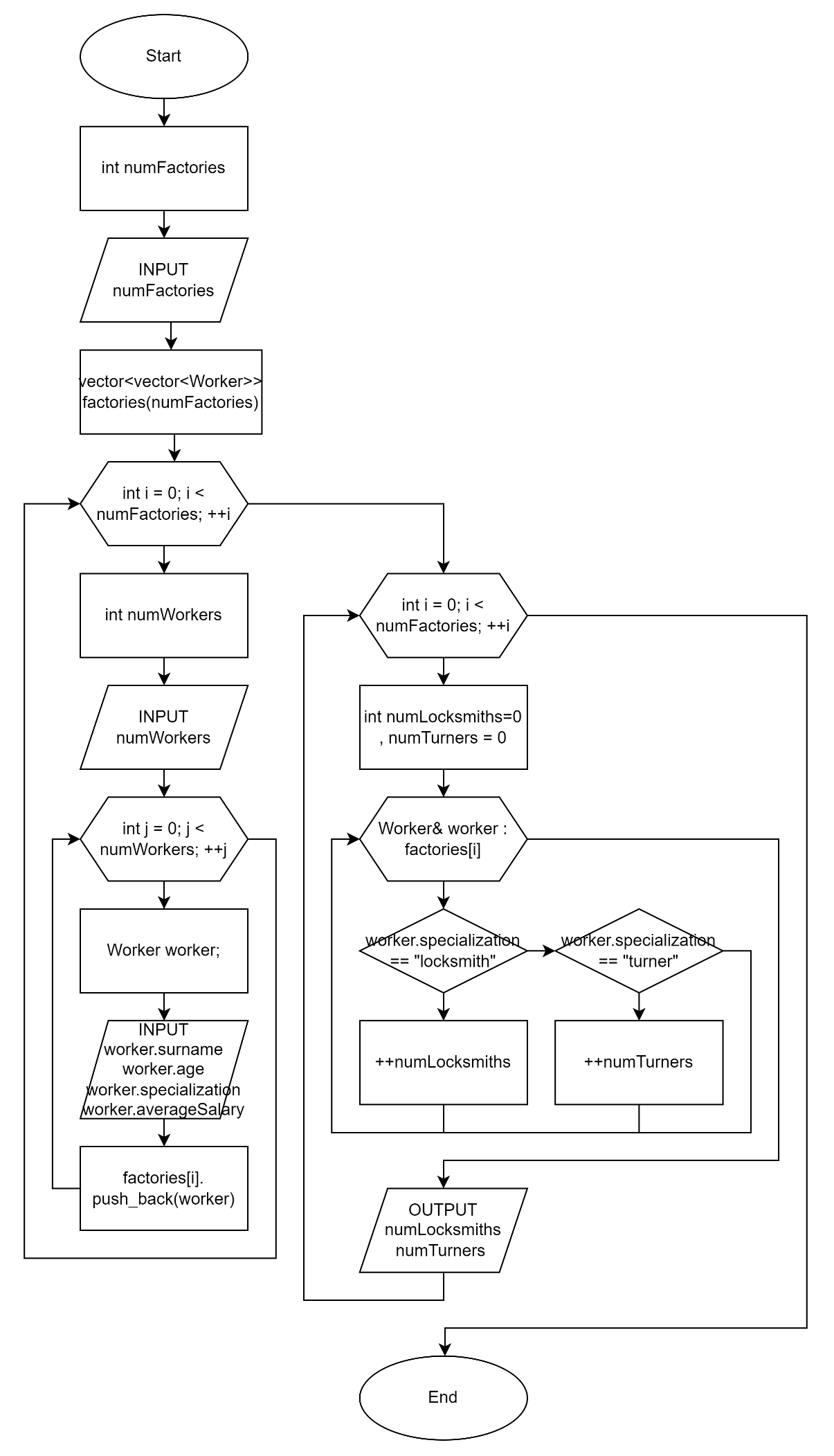
Информация по работникам N заводов задается структурой с полями: фамилия, возраст, специальность, средний оклад. Ввести информацию по заводам, посчитать количество слесарей и токарей. Вывести эти значения на консоль.

**Исходные данные:**

N (numFactories) – количество заводов.

struct Worker {  
 string surname;  
 int age;  
 string specialization;  
 double averageSalary;  
};

**Блок-схемы:**

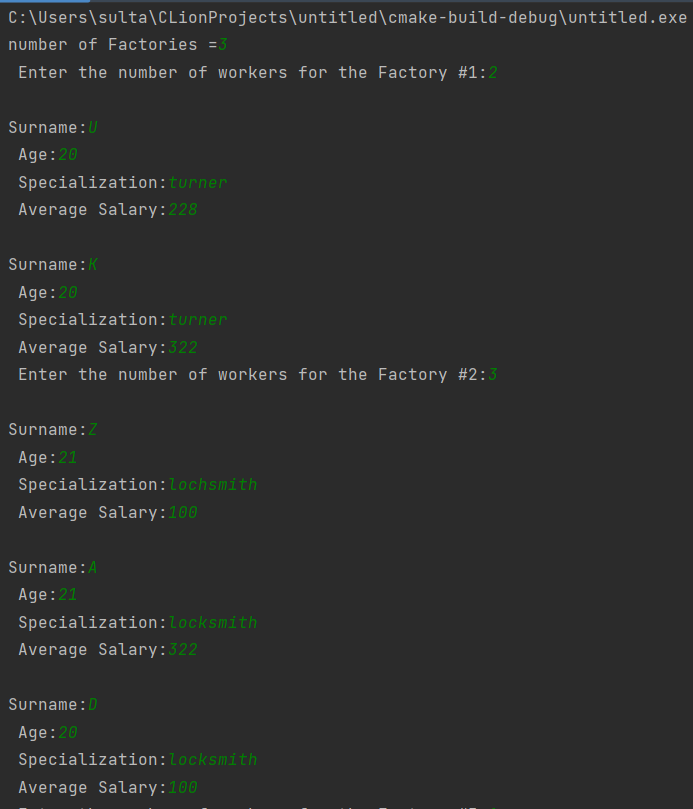


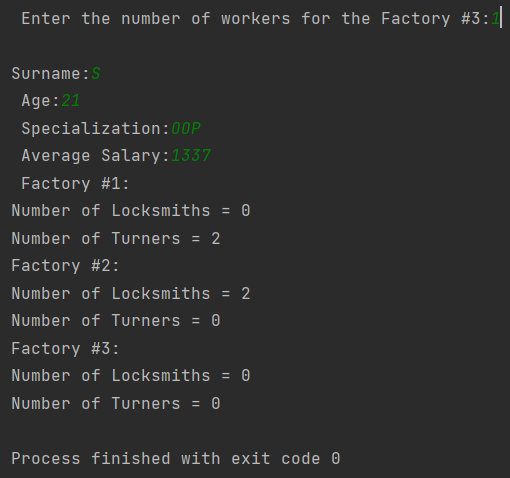
**Исходный код программы на C++:**

#include <bits/stdc++.h>  
  
using namespace std;  
struct Worker {  
 string surname;  
 int age;  
 string specialization;  
 double averageSalary;  
};  
  
int main() {  
 int numFactories;  
 cout<<"number of Factories = ";cin>>numFactories;  
 vector<vector<Worker>> factories(numFactories);  
  
 for (int i = 0; i < numFactories; ++i) {  
 int numWorkers;  
 cout<<"Enter the number of workers for the Factory #"<<i+1<<": ";  
 cin >> numWorkers;  
  
 for (int j = 0; j < numWorkers; ++j) {  
 Worker worker;  
 cout<<endl;  
 cout << "Surname: ";  
 cin >> worker.surname;  
 cout << "Age: ";  
 cin >> worker.age;  
 cout << "Specialization: ";  
 cin >> worker.specialization;  
 cout << "Average Salary: ";  
 cin >> worker.averageSalary;  
  
 factories[i].push\_back(worker);  
 }  
 }  
  
 for (int i = 0; i < numFactories; ++i) {  
 int numLocksmiths = 0; //Слесари  
 int numTurners = 0; //Токари  
  
 for (const Worker& worker : factories[i]) {  
 if (worker.specialization == "locksmith") {  
 ++numLocksmiths;  
 } else if (worker.specialization == "turner") {  
 ++numTurners;  
 }  
 }  
  
 cout << "Factory #" << i + 1 << ":" << endl;  
 cout << "Number of Locksmiths = " << numLocksmiths << endl;  
 cout << "Number of Turners = " << numTurners << endl;  
  
 }  
  
 return 0;  
}

**Тесты:**

**Тест1**



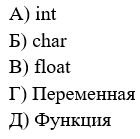
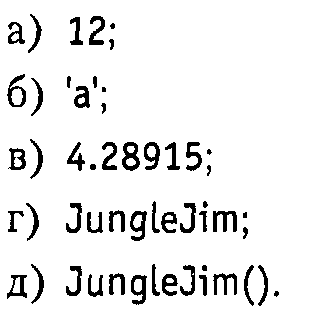


**Ответы на вопросы:**

**1) Какие стандартные типы используются в С++? Сколько под них резервируется памяти?**

1. **int (целочисленный тип):**
   * Размер: обычно 4 байта (32 бита) на большинстве систем.
   * Пример: **int x = 10;**
2. **float (тип с плавающей запятой одинарной точности):**
   * Размер: обычно 4 байта.
   * Пример: **float y = 3.14f;**
3. **double (тип с плавающей запятой двойной точности):**
   * Размер: обычно 8 байт.
   * Пример: **double z = 2.71828;**
4. **char (символьный тип):**
   * Размер: обычно 1 байт.
   * Пример: **char c = 'A';**
5. **bool (логический тип):**
   * Размер: зависит от реализации, но обычно 1 байт.
   * Пример: **bool flag = true;**
6. **short (короткое целое):**
   * Размер: обычно 2 байта.
   * Пример: **short s = 42;**
7. **long (длинное целое):**
   * Размер: обычно 4 байта (но может быть больше на некоторых системах).
   * Пример: **long l = 1234567890;**
8. **long long (очень длинное целое):**
   * Размер: обычно 8 байт.
   * Пример: **long long ll = 9876543210;**

**2) К каким элементам программы относятся следующие:**



1. **Что такое функция? Какова роль функций в языке С++?**

В языке программирования C++, функция представляет собой блок кода, который имеет имя и может выполнять определенные действия. Функции в C++ играют ключевую роль, позволяя разбивать программы на более мелкие и легко управляемые части. Они способствуют структурированию кода, повторному использованию, и улучшают читаемость программ.

1. **Напишите синтаксис функции.**

тип\_возвращаемого\_значения имя\_функции(параметры\_функции) {

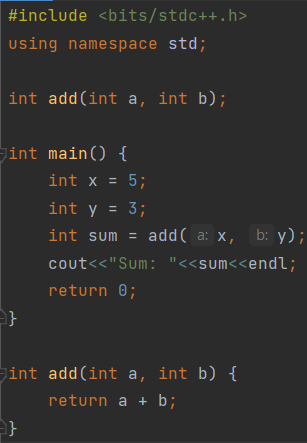
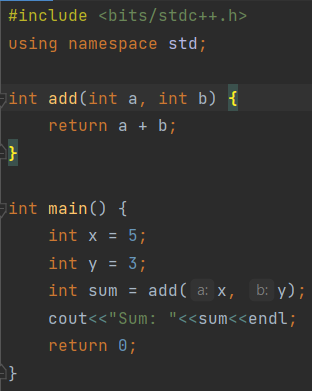
// тело функции

// код выполняемых операций

// оператор return (если функция возвращает значение)

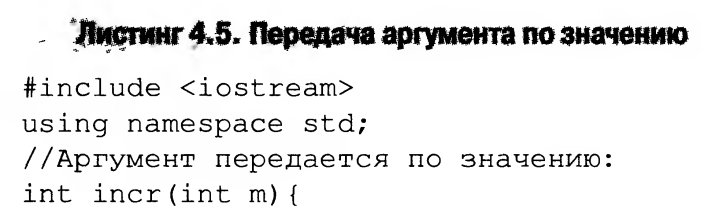
}

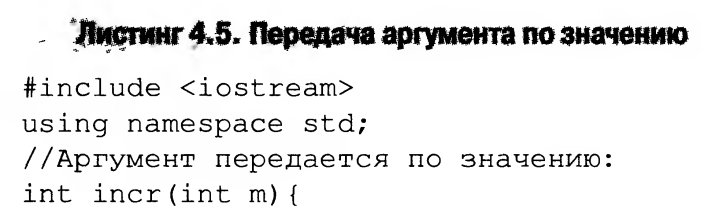
1. **Опишите способы использования функций в программах с объявлением функций и без объявления функций.**

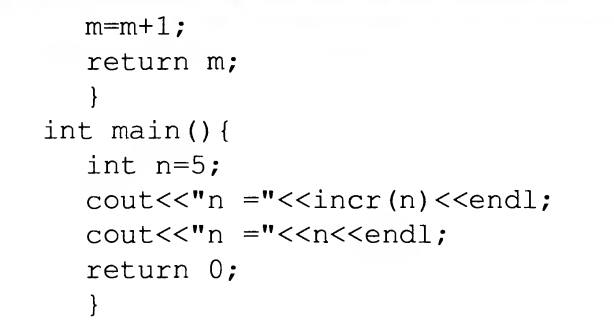


1. **Объясните механизмы передачи аргументов по значению и по ссылке в функцию.**

**Объясните результаты работы программ.**







2)





1. **По значению:**

При передаче аргументов по значению в функцию создается копия значения аргумента, и функция работает с этой копией. Исходное значение переменной, переданной в функцию, не изменяется после завершения функции.

В этой программе используется передача по значению. Функция incr получает копию значения переменной n, увеличивает ее на 1, но это изменение не влияет на исходное значение n. Поэтому результатом будет вывод n = 6, но значение переменной n в main останется равным 5.

1. **По ссылке:**

При передаче аргументов по ссылке в функцию передается адрес памяти (ссылка) на переменную, а не ее копия. Функция работает с самой переменной, и изменения, внесенные внутри функции, затрагивают исходное значение переменной в вызывающей программе.

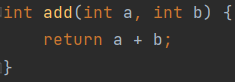
В этой программе используется передача по ссылке. Функция incr принимает переменную n по ссылке, и изменения, внесенные внутри функции, напрямую влияют на исходное значение n. Поэтому результатом будет вывод n = 6, и значение переменной n в main также станет равным 6.

1. **Назовите разновидности аргументов, которые могут быть переданы параметрам функции?**

**Обязательные аргументы:**

Это аргументы, которые функция обязана принять. Если вы вызываете функцию с определенными параметрами, эти значения должны быть предоставлены.

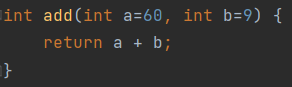




**Параметры по умолчанию:**

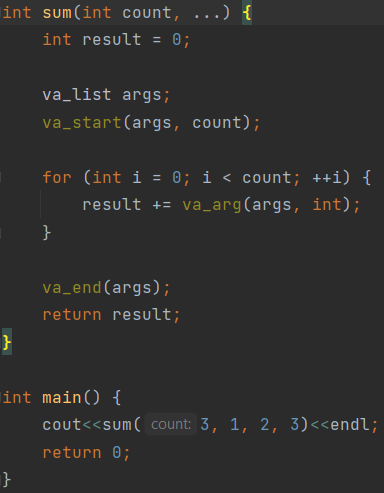
Параметры функции могут иметь значения по умолчанию, что позволяет вызывающему коду опускать их при вызове





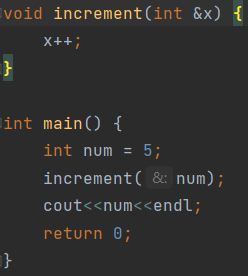
**Аргументы переменной длины:**

Функции также могут принимать переменное количество аргументов с использованием многоточия ... или с использованием стандартной библиотеки <cstdarg>.



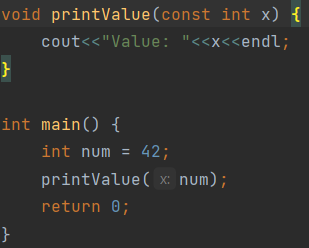
**Ссылочные аргументы:**

Параметры функции могут быть переданы по ссылке, что позволяет функции изменять значения аргументов в вызывающем коде.



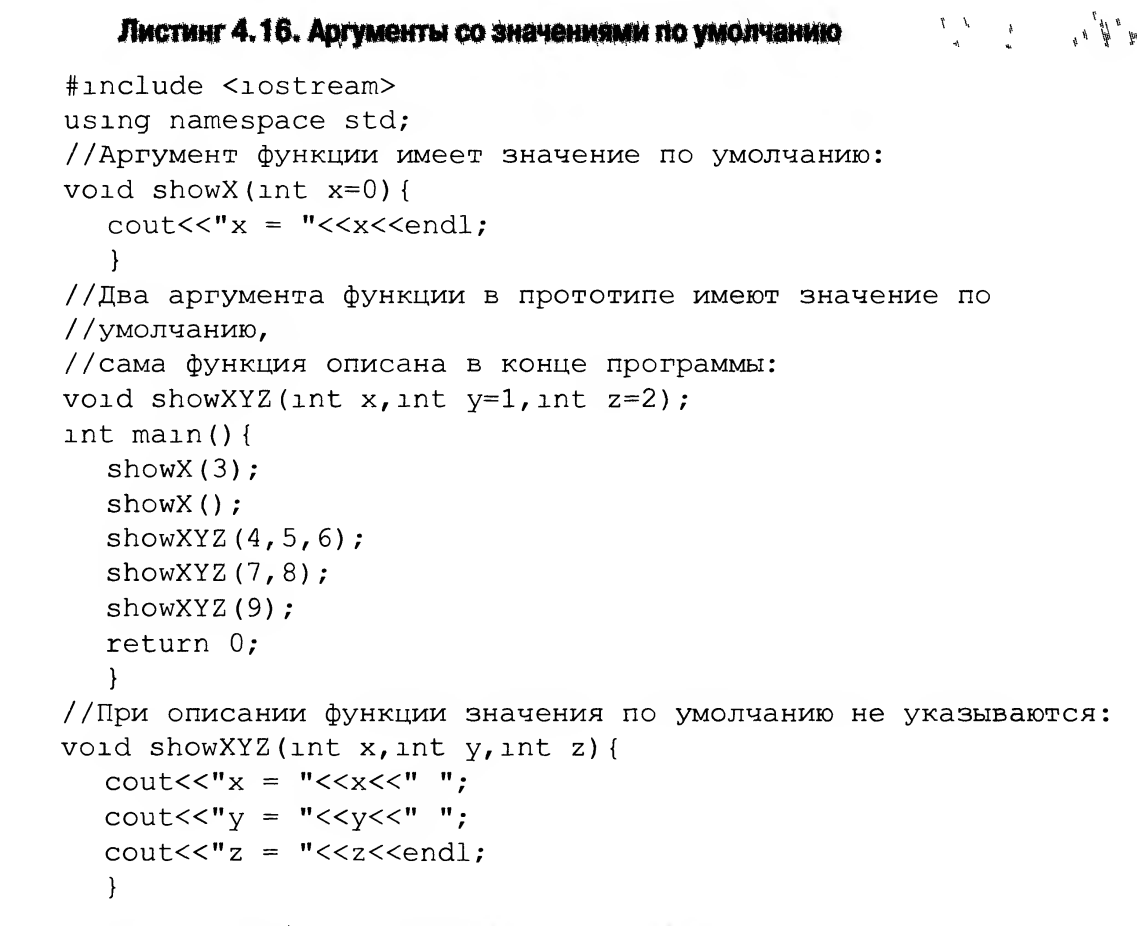
**Константные аргументы:**

Параметры функции могут быть объявлены как константные, что предотвращает их изменение внутри функции.



1. **Что такое аргументы по умолчанию?**

Объясните результат работы программы:



Аргументы по умолчанию в C++ представляют собой значения, которые могут быть предварительно заданы для параметров функции. Если значение не передано при вызове функции, то используется значение по умолчанию, указанное при определении функции.

**Объяснение:**

showX(3); Вызов функции showX с аргументом 3. Выводится "x = 3".

showX(); Вызов функции showX без аргумента. Используется значение по умолчанию (0). Выводится "x = 0".

showXYZ(4, 5, 6); Вызов функции showXYZ с явными значениями для всех параметров. Выводится "x = 4 y = 5 z = 6".

showXYZ(7, 8); Вызов функции showXYZ с явным значением для x и y, а значение z используется по умолчанию (2). Выводится "x = 7 y = 8 z = 2".

showXYZ(9); Вызов функции showXYZ с явным значением только для x, значения y и z используются по умолчанию. Выводится "x = 9 y = 1 z = 2".

1. **Что такое перегрузка функций?**

Перегрузка функции в C++ — это возможность определить несколько функций с одинаковыми именами, но различающимися в их сигнатуре (наборе параметров). При вызове функции компилятор выбирает подходящую функцию на основе переданных аргументов.

1. **Чем отличается структура (struct) от массива?**

Структура - это пользовательский тип данных, который может содержать переменные различных типов данных, объединенные под общим именем.

Массив - это упорядоченный набор элементов одного и того же типа данных.

**Отличия**:

Типы данных: В структуре элементы могут иметь разные типы данных, в массиве элементы имеют одинаковый тип.

Доступ к данным: В структуре доступ к данным осуществляется по именам полей, в массиве - по индексам.

Размер: Размер массива фиксирован, определен при объявлении. Размер структуры определяется суммой размеров ее полей.