2- Практическое задание – Использование Функций в C++

Table of Contents

Введение в Функции

Программа на C++, которая определяет функцию для сложения двух чисел и вызывает эту функцию:

```cpp
#include <iostream>
#include <windows.h>
using namespace std;

// Объявление функции
int addNumbers(int a, int b);

int main() {
    SetConsoleOutputCP(CP_UTF8);
    SetConsoleCP(CP_UTF8);

    int num1, num2, sum;

    // Ввод пользователем двух чисел
    cout << "Введите первое число: ";
    cin >> num1;
    cout << "Введите второе число: ";
    cin >> num2;

    // Вызов функции для сложения чисел
    sum = addNumbers(num1, num2);

    // Вывод результата
    cout << "Сумма = " << sum << endl;

    return 0;
}

// Определение функции
int addNumbers(int a, int b) {
    return a + b;
}
```

Эта программа состоит из:

1. Функции **addNumbers**, которая принимает два целых числа в качестве входных данных и возвращает их сумму.
2. Функции **main**, где пользователь вводит два числа. Эти числа затем передаются в функцию **addNumbers**.
3. Результат затем отображается на экране

Примечание

Определение функции может быть размещено как до, так и после функции main. Если оно находится до main, отдельное объявление не обязательно (поскольку определение также выполняет роль объявления). Если оно находится после main, объявление требуется до main, чтобы обеспечить, что компилятор знает о функции, когда он встречает вызов этой функции в main.

Задача самостоятельной реализации 1

Измените приведенный выше пример, добавив три функции: первая для вычитания чисел, вторая для умножения чисел и последняя для деления чисел. И в главной функции **main** позвольте пользователю выбрать, какую из этих функций он хочет реализовать

Параметры Функций и Типы Возвращаемых Значений

В целом, эта программа служит простым учебным примером или примером того, как определяются и используются функции с различными типами возвращаемых значений в C++:

```
#include <iostream>
#include <windows.h>

using namespace std;

// Функция void: Не возвращает никакого значения
void printMessage() {
          cout << "Это функция void. Она не возвращает никакого значения." << endl;
}

// Функция bool: Возвращает булево значение
bool isEven(int num) {
          return (num % 2 == 0);
}

// Функция int: Возвращает целочисленное значение
int doubleNumber(int num) {
          return num * 2;
}

// Функция double: Возвращает значение с плавающей точкой
double halfNumber(double num) {
          return num / 2.0;
}

int main() {
          // Set console to UTF-8 to support Cyrillic characters
          SetConsoleOutputCP(CP_UTF8);
          SetConsoleCP(CP_UTF8);

          // Демонстрация функции void
          printMessage();

          // Демонстрация функции bool
          int number = 4;
          cout << "Является ли " << number << " четным? " << (isEven(number) ? "Да" : "Нет") << endl;

          // Демонстрация функции int
          int value = 5;
          cout << "Удвоенное значение " << value << " равно " << doubleNumber(value) << endl;

          // Демонстрация функции double
          double num = 8.5;
          cout << "Половина от " << num << " равна " << halfNumber(num) << endl;


          return 0;
}
```

1- **Подключение библиотек и пространства имен:**
   - #include <iostream>: Подключает стандартную библиотеку потоков ввода/вывода.
   - #include <windows.h>: Подключает заголовочный файл Windows, специфичный для функций Windows API.
   - using namespace std;: Позволяет программе использовать объекты и функции стандартной библиотеки без указания префикса std::.
2- **Объявления функций:**
   - void printMessage(): Функция, которая не возвращает значение (void). Просто выводит сообщение в консоль.
   - bool isEven(int num): Функция, возвращающая булево значение. Проверяет, является ли переданное целое число (num) четным.
   - int doubleNumber(int num): Функция, возвращающая целое число. Удваивает значение переданного целого числа (num).
   - double halfNumber(double num): Функция, возвращающая значение с плавающей точкой. Делит пополам значение переданного числа с плавающей точкой (num).
3- **Главная функция (int main()):**
   - Устанавливает кодовую страницу консоли UTF-8 с помощью SetConsoleOutputCP(CP_UTF8) и SetConsoleCP(CP_UTF8), чтобы корректно отображать кириллические символы.
   - Вызывает printMessage() для демонстрации функции void.
   - Демонстрирует функцию bool isEven(), проверяя, является ли жестко заданное целое число (number = 4) четным.
   - Демонстрирует функцию int doubleNumber(), удваивая жестко заданное целое число (value = 5).
   - Демонстрирует функцию double halfNumber(), деля пополам жестко заданное значение с плавающей точкой (num = 8.5).
   - Программа завершается с return 0;, указывая на успешное выполнение.

Задача самостоятельной реализации 2

Создайте программу на C++, которая демонстрирует использование циклов и функций. Программа должна включать следующие компоненты:

1- **Функция bool**: isPrime(int num), которая принимает целое число и возвращает true, если число является простым, и false в противном случае. Чтобы помочь вам, код этой функции выглядит следующим образом:

```
bool isPrime(int num) {
    if (num <= 1) return false;
    for (int i = 2; i <= sqrt(num); ++i) {
        if (num % i == 0) return false;
    }
    return true;
}
```

2- **Функция void**: Создайте функцию printMultiplicationTable(int num), которая принимает целое число и печатает таблицу умножения для этого числа до 10.
3- **Главная функция:**
   - Попросите пользователя ввести целое число.
   - Используйте цикл для повторения процесса пять раз.
   - Для каждого ввода:
     - ❖ Проверьте, является ли число простым с помощью isPrime().
     - ❖ Напечатайте таблицу умножения числа с помощью printMultiplicationTable().

Примечание

1- **sqrt(num)**: Эта функция принимает один аргумент num и возвращает квадратный корень из num. Если num отрицательно, результатом будет ошибка домена, поскольку квадратный корень из отрицательного числа не определен в множестве вещественных чисел.

2- **#include <cmath>**: Эта строка подключает заголовочный файл <cmath>. Заголовочный файл <cmath> предоставляет набор функций для выполнения математических операций, таких как квадратные корни (sqrt), тригонометрические функции (например, sin, cos, tan), экспоненциальные функции (exp) и логарифмы (log). Подключив этот заголовочный файл, вы делаете эти математические функции доступными для использования в вашей программе.

Используя глобальные и локальные переменные, рекурсию, проверьте ввод пользователя

Ниже представлена программа на C++, демонстрирующая использование локальных и глобальных областей видимости с помощью нескольких функций. В неё включены функция для ввода и проверки числа, рекурсивная функция для вычисления факториала, и функция для отображения результата.

Описание Программы

Эта программа будет:

1. Принимать целочисленный ввод от пользователя.
2. Проверять ввод (он должен быть положительным).
3. Вычислять факториал числа.
4. Отображать результат.

```
#include <iostream>
#include <windows.h>

using namespace std;

// Глобальная переменная
int globalNumber = 0;

// Функция для ввода и проверки числа
void inputNumber() {
    int number;
    do {
        cout << "Введите положительное число: ";
        cin >> number;
        if (number <= 0) {
            cout << "Неверный ввод. Пожалуйста, введите положительное число." << endl;
        }
    } while (number <= 0);

    // Присваиваем допустимое число глобальной переменной
    globalNumber = number;
}

// Рекурсивная функция для вычисления факториала
int calculateFactorial(int num) {
    if (num == 0 || num == 1) {
        return 1;
```

```
    }
    else {
        return num * calculateFactorial(num - 1);
    }
}

// Функция для отображения результата
void displayResult(int result) {
    cout << "Факториал числа " << globalNumber << " равен " << result << endl;
}

int main() {
    SetConsoleOutputCP(CP_UTF8);
    SetConsoleCP(CP_UTF8);

    // Получаем ввод пользователя
    inputNumber();

    // Вычисляем факториал
    int factorial = calculateFactorial(globalNumber);

    // Отображаем результат
    displayResult(factorial);

    return 0;
}
```

Объяснение

1. Глобальная Переменная: globalNumber объявлена вне любой функции, что делает её глобальной переменной, доступной для всех функций в программе.

2. Функция inputNumber(): Эта функция считывает целое число от пользователя и убеждается, что оно положительное. Проверенное число затем сохраняется в globalNumber.

3. Функция calculateFactorial(int num): Рекурсивная функция, которая вычисляет факториал заданного числа. Она демонстрирует локальную область видимости, так как num является локальной переменной этой функции.

4. Функция displayResult(int result): Эта функция принимает целое число (результат факториала) и отображает его. Она использует глобальную переменную globalNumber для отображения исходного вводимого числа.

5. Функция main(): Координирует ход программы, вызывая inputNumber(), calculateFactorial() и displayResult() последовательно.

Перегрузка Функций

В C++ перегрузка функций позволяет иметь несколько функций с одинаковым именем, но с разными параметрами.

Пример Программы

Эта программа будет включать перегруженные функции add для следующих сценариев:

1. Сложение двух целых чисел.
2. Сложение трех целых чисел.
3. Сложение двух чисел с плавающей запятой (тип double).
4. Сложение целого числа и числа с плавающей запятой.

```
#include <iostream>

using namespace std;

// Функция для сложения двух целых чисел
```

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

// Функция для сложения трех целых чисел
int add(int a, int b, int c) {
    return a + b + c;
}

// Функция для сложения двух чисел с плавающей запятой
double add(double a, double b) {
    return a + b;
}

// Функция для сложения целого числа и числа с плавающей запятой
double add(int a, double b) {
    return a + b;
}

int main() {
    // Демонстрация сложения двух целых чисел
    cout << "Сложение двух целых чисел: " << add(5, 3) << endl;

    // Демонстрация сложения трех целых чисел
    cout << "Сложение трех целых чисел: " << add(5, 3, 2) << endl;

    // Демонстрация сложения двух чисел с плавающей запятой
    cout << "Сложение двух чисел с плавающей запятой: " << add(2.5, 3.4) << endl;

    // Демонстрация сложения целого числа и числа с плавающей запятой
    cout << "Сложение целого числа и числа с плавающей запятой: " << add(5, 2.3) << endl;

    return 0;
}
```

Объяснение

1. Первая функция add складывает два целых числа.
2. Вторая функция add складывает три целых числа.
3. Третья функция add складывает два числа с плавающей запятой.
4. Четвертая функция add складывает целое число и число с плавающей запятой.

Каждая функция имеет разную сигнатуру (то есть количество и тип параметров), что позволяет C++ различать их. Эта программа ясно демонстрирует, как можно использовать перегрузку функций для выполнения похожих операций с различными типами данных или разным количеством аргументов.

Задача: реализовать, изучить и объяснить

Ваша задача - реализовать следующий код, изучить его самостоятельно и объяснить практическим инструкторам, как работает каждая функция и как код проверяет ввод пользователя, есть ли в нем рекурсия, где и как, есть ли перегрузка функций, где и как:

```
#include <iostream>
#include <windows.h>

using namespace std;

int power(int base, int exponent) {
    if (exponent == 0) return 1;
    return base * power(base, exponent - 1);
}

double power(double base, int exponent) {
    if (exponent == 0) return 1.0;
    return base * power(base, exponent - 1);
}
```

```cpp
int main() {

    SetConsoleOutputCP(CP_UTF8);
    SetConsoleCP(CP_UTF8);

    int intBase, exponent;
    double doubleBase;
    char baseType;

    cout << "Введите 'i' для целочисленного основания или 'd' для основания с плавающей запятой: ";
    cin >> baseType;

    cout << "Введите показатель степени (неотрицательное целое число): ";
    cin >> exponent;

    if (exponent < 0) {
        cout << "Показатель степени должен быть неотрицательным." << endl;
        return 1;
    }

    if (baseType == 'i') {
        cout << "Введите целочисленное основание: ";
        cin >> intBase;
        cout << "Результат: " << power(intBase, exponent) << endl;
    }
    else if (baseType == 'd') {
        cout << "Введите основание с плавающей запятой: ";
        cin >> doubleBase;
        cout << "Результат: " << power(doubleBase, exponent) << endl;
    }
    else {
        cout << "Неверный тип основания." << endl;
    }

    return 0;
}
```