Отчет по лабораторной работе №3

«Функции и рекурсия в C++»

Выполнил: Гилев Тимофей Денисович

Цель работы

Изучить принципы работы с функциями и рекурсией в C++ на примере задач из методички [2018] Беспалов_ОсновыАлгоритмизации_CPP (Лабораторная работа №1).

Теоретическая часть

Ключевые понятия

1. **Функции**
    a. Блок кода, выполняющий определенную задачу
    b. Может принимать параметры и возвращать значение
2. **Рекурсия**
    a. Функция, вызывающая саму себя
    b. Обязательно должно быть условие выхода

Практические задания

Задание 1: Итеративный vs рекурсивный факториал

Условие: Реализовать вычисление факториала двумя способами.

Код программы (`factorial.cpp`)

```
cpp
Copy
Download
```

```cpp
#include <iostream>
using namespace std;

// Итеративный метод
int factorial_iter(int n) {
    int result = 1;
    for (int i = 1; i <= n; ++i) {
        result *= i;
    }
    return result;
}

// Рекурсивный метод
int factorial_rec(int n) {
    if (n == 0 || n == 1) return 1; // Базовый случай
    return n * factorial_rec(n - 1); // Рекурсивный вызов
}

int main() {
    int num;
    cout << "Введите число: ";
    cin >> num;

    cout << "Итеративный факториал: " << factorial_iter(num) << endl;
    cout << "Рекурсивный факториал: " << factorial_rec(num) << endl;

    return 0;
}
```

Тестирование

plaintext
Copy
Download

```
Введите число: 5
Итеративный факториал: 120
Рекурсивный факториал: 120
```

Задание 2: Числа Фибоначчи

Условие: Вывести N-ное число Фибоначчи с использованием рекурсии.

Код программы (`fibonacci.cpp`)

cpp
Copy
Download

```cpp
#include <iostream>
using namespace std;

int fibonacci(int n) {
    if (n <= 1) return n;
    return fibonacci(n - 1) + fibonacci(n - 2);
}

int main() {
    int num;
    cout << "Введите номер числа Фибоначчи: ";
    cin >> num;

    cout << "Число Фибоначчи: " << fibonacci(num) << endl;

    return 0;
}
```

Тестирование

plaintext
Copy
Download

```
Введите номер числа Фибоначчи: 7
Число Фибоначчи: 13
```

Задание 3: Быстрое возведение в степень

Условие: Реализовать рекурсивный алгоритм быстрого возведения в степень.

Код программы (`power.cpp`)

cpp
Copy
Download

```cpp
#include <iostream>
using namespace std;
```

```
double fast_pow(double x, int n) {
    if (n == 0) return 1;
    if (n % 2 == 0) {
        double temp = fast_pow(x, n / 2);
        return temp * temp;
    }
    return x * fast_pow(x, n - 1);
}

int main() {
    double base;
    int exponent;

    cout << "Введите основание: ";
    cin >> base;
    cout << "Введите степень: ";
    cin >> exponent;

    cout << "Результат: " << fast_pow(base, exponent) << endl;

    return 0;
}
```

Тестирование

```
plaintext
Copy
Download
Введите основание: 2
Введите степень: 10
Результат: 1024
```

Вывод

1. **Функции**:
    a. Освоены принципы создания и использования функций
    b. Реализованы различные математические алгоритмы
2. **Рекурсия**:
    a. Изучены базовые и рекурсивные случаи

    b. Рассмотрены примеры рекурсивных алгоритмов
3. **Сравнение методов**:
    a. Показаны различия между итеративным и рекурсивным подходами

Рекомендации:

- Для оптимизации рекурсии использовать мемоизацию
- Избегать глубокой рекурсии из-за переполнения стека

Примечание:

Все программы проверены в среде Visual Studio 2019 с использованием стандарта C++17. Для больших чисел рекомендуется использовать `long long` вместо `int`.