**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

Лабораторная работа №4

по курсу

«Операционные системы и сети»

Выполнили:

студенты группы ИКПИ-14

Хохлов Т.В.

Принял:

доцент кафедры ПИиВТ

Дагаев А.В

Санкт-Петербург

2023 г.

**Цель работы**

Целью работы является разработка программы на языке C++ с использованием библиотеки потоков <thread>.

# Описание программы

Программа является многопоточным приложением, реализованным с использованием библиотеки Windows API (WinAPI). Два потока выполняются параллельно: один генерирует случайное число, а другой пытается угадать это число. Третий поток отвечает за ввод пользователя. Для синхронизации потоков и для эксклюзивного доступа к ресурсам используются мьютексы.

Описание кода

**Основные компоненты:**

Критические секции:

**generateSecretNumber():**

* Функция потока, генерирующего случайное число в диапазоне от 1 до 100.
* Число обновляется каждые 2 секунды.

**guessNumber():**

* Функция потока, который пытается угадать сгенерированное число.
* Попытка происходит каждые 500 миллисекунд.

**userInput():**

* Функция потока, ожидающего ввода пользователя.
* Программа завершается по команде пользователя ("s").

Мьютекс:

**HANDLE mtx;**

* В функции **generateSecretNumber**, перед генерацией нового секретного числа, поток "захватывает" мьютекс, чтобы обновить **secretNumber** и вывести информацию о загаданном числе.
* В функции **guessNumber**, перед угадыванием числа, поток "захватывает" мьютекс, чтобы обновить **numberGuessAttempts** и вывести информацию о задуманном числе.
* В функции **userInput**, после пользовательского ввода, поток "захватывает" мьютекс, чтобы обновить **programStop** и вывести информацию о завершении программы.

Основная функция:

**main():**

* Создаются три потока с возвращением их дескрипторов используя функцию **CreateThread**.
* Ожидание завершения всех потоков с использованием **WaitForMultipleObjects**.
* Закрытие дескрипторов потоков с использованием **CloseHandle**.
* После завершения потоков выводится статистика:
  + Общее количество сгенерированных чисел.
  + Количество угаданных чисел.
  + Общее количество попыток угадывания.
  + Процент успешных и неудачных попыток.

**Завершение программы:**

Программа завершается, когда пользователь вводит от клавиатуры **“s”** в терминал. По завершении выводится статистика и сообщение "Program finished!".

**Статистика:**

Статистика включает в себя общее количество сгенерированных чисел, количество угаданных чисел, общее количество попыток угадывания, а также процент успешных и неудачных попыток.

**Графики:**





**Выводы**

В ходе проведённой лабораторной работы были получены следующие результаты:

1. Освоена библиотека **WINAPI** языка C++.

2. Создана программа с использованием параллельных потоков.

Компиляция программы

g++ -std=c++11 test4.cpp -o test4

Исходный код программы

#include <iostream>

#include <random>

#include <chrono>

#include <string>

#include <windows.h>

HANDLE mtx;

bool programStop = false;

int secretNumber = 0;

float numbersGenerated = 0;

float numbersGuessed = 0;

float numberGuessAttempts = 0;

DWORD WINAPI generateSecretNumber(LPVOID lpParam) {

    std::random\_device rd;

    std::mt19937 gen(rd());

    std::uniform\_int\_distribution<> distr(1, 100);

    while (!programStop) {

        WaitForSingleObject(mtx, INFINITE);

        secretNumber = distr(gen);

        std::cout << "Thread 1: New secret number is generated: " << secretNumber << std::endl;

        numbersGenerated++;

        ReleaseMutex(mtx);

        Sleep(2000);

    }

    return 0;

}

DWORD WINAPI guessNumber(LPVOID lpParam) {

    std::random\_device rd;

    std::mt19937 gen(rd());

    std::uniform\_int\_distribution<> distr(1, 100);

    while (!programStop) {

        Sleep(500);

        WaitForSingleObject(mtx, INFINITE);

        int guess = distr(gen);

        std::cout << "Thread 2: Guessing " << guess << std::endl;

        numberGuessAttempts++;

        if (guess == secretNumber) {

            std::cout << "Thread 2: Guessed the number " << secretNumber << std::endl;

            numbersGuessed++;

        }

        ReleaseMutex(mtx);

    }

    return 0;

}

DWORD WINAPI userInput(LPVOID lpParam) {

    std::string input;

    while (true) {

        std::getline(std::cin, input);

        if (input == "s") {

            WaitForSingleObject(mtx, INFINITE);

            programStop = true;

            ReleaseMutex(mtx);

            break;

        }

    }

    return 0;

}

int main() {

    mtx = CreateMutex(NULL, FALSE, NULL);

    HANDLE threads[3];

    threads[0] = CreateThread(NULL, 0, generateSecretNumber, NULL, 0, NULL);

    threads[1] = CreateThread(NULL, 0, guessNumber, NULL, 0, NULL);

    threads[2] = CreateThread(NULL, 0, userInput, NULL, 0, NULL);

    WaitForMultipleObjects(3, threads, TRUE, INFINITE);

    CloseHandle(threads[0]);

    CloseHandle(threads[1]);

    CloseHandle(threads[2]);

    CloseHandle(mtx);

    std::cout << "Program finished!" << std::endl;

    std::cout << "Number of times the number was generated: " << numbersGenerated << std::endl;

    std::cout << "Number of times the number was guessed: " << numbersGuessed << std::endl;

    std::cout << "Number of times the number was attempted to guess: " << numberGuessAttempts << std::endl;

    std::cout << std::endl << "Statistics: " << std::endl;

    std::cout << "Success attempts: " << numbersGuessed / numberGuessAttempts << "freq" << " or " << (numbersGuessed / numberGuessAttempts) \* 100 << "%" << std::endl;

    std::cout << "Failure attempts: " << (numberGuessAttempts - numbersGuessed) / numberGuessAttempts << "freq" << " or " << ((numberGuessAttempts - numbersGuessed) / numberGuessAttempts) \* 100 << "%" << std::endl;

    return 0;

}