



ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА
(САМАРСКИЙ УНИВЕРСИТЕТ)»

Институт _____ Информатики и кибернетики
Кафедра _____ Программных систем
Дисциплина _____ Технологии промышленного программирования

ОТЧЁТ

к лабораторной работе
«Запуск и синхронизация нитей»

Обучающийся группы 6231-020302D _____ Гижевская В.Д.

Преподаватель _____ Баландин А.В.

Самара 2023

СОДЕРЖАНИЕ

1	Постановка задачи.....	3
2	Результаты работы	4
	ПРИЛОЖЕНИЕ А Листинг модуля M1	5

1 Постановка задачи

Разработать приложение, состоящее из одного процесса с тремя запущенными нитями:

1. M(main).
2. T1(F1).
3. T2(F2).

В качестве нити M(main) выступает функция main(). Нити T1(F1) и T2(F2) запускаются нитью M(main) на базе соответственно функций F1() и F2(). Все три нити, работая параллельно, должны совместно динамически сформировать текст вида: "Text0, Text1, Text2.\n".

Порядок работы приложения

Вначале нить M(main) запускает первой нить T1(F1), затем - T2(F2), передавая им в качестве параметра указатель совместно формируемого текста.

Далее нить M(main), записывая в текст букву за буквой, формирует свою часть текста: "Text0, ".

После формирования нитями всего текста нить M(main) выдаёт его на печать и завершает свою работу.

Запущенная нить T1(F1) должна тем же способом добавить в формируемый текст свою часть: "Text1, ".

После завершения записи своей части текста нить T1(F1) ожидает записи в буфер своей части текста нитью T2(F2), после чего завершает свою работу.

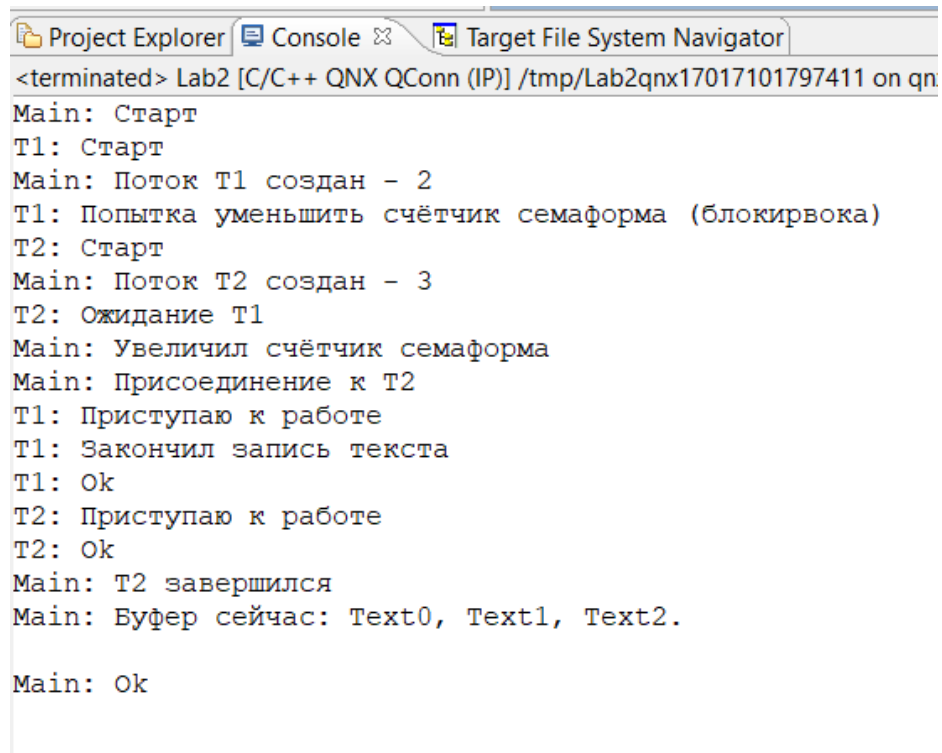
Запущенная нить T2(F2) должна так же добавить в формируемый текст свою часть: "Text2.\n ", после чего завершает свою работу.

Замечание. Для имитации времени записи в текст одной буквы использовать пустой цикл в 1000 итераций.

Необходимо использовать: Именованные семафоры. Ждущие блокировки. Присоединение.

2 Результаты работы

Результаты работы представлены в виде вывода на консоль сообщений во время выполнения программы.



```
<terminated> Lab2 [C/C++ QNX QConn (IP)] /tmp/Lab2qnx17017101797411 on qn
Main: Старт
T1: Старт
Main: Поток T1 создан - 2
T1: Попытка уменьшить счётчик семафора (блокировка)
T2: Старт
Main: Поток T2 создан - 3
T2: Ожидание T1
Main: Увеличил счётчик семафора
Main: Присоединение к T2
T1: Приступаю к работе
T1: Закончил запись текста
T1: Ok
T2: Приступаю к работе
T2: Ok
Main: T2 завершился
Main: Буфер сейчас: Text0, Text1, Text2.
Main: Ok
```

Рисунок 1 – Результат выполнения программы в консоли

ПРИЛОЖЕНИЕ А

Листинг модуля M1

```
#include <cstdlib>
#include <iostream>

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <pthread.h>
#include <pthread.h>

#include <semaphore.h>
#include <fcntl.h>

#include <time.h>

#include <unistd.h>
#define GetCurrentDir getcwd

using std::cout;
using std::endl;

// Именованные семафоры. Ждущие блокировки. Присоединение

#define SEMAPHORE_NAME "/lab2_semaphore_name"

// Запись одного символа с имитацией времени записи
void appendChar(char *text, char c);
// Функция потока T1
void* funcT1(void* args);
// Функция потока T2
void* funcT2(void* args);

// Семафор //
sem_t* sem;

// Ждущие блокировки //
// Флаг завершения записи нитью T1
bool isDoneT1 = false;

// Записываемый текст
char* mainText = (char*) "Text0, ";
char* t1Text = (char*) "Text1, ";
char* t2Text = (char*) "Text2.\n ";

int main(int argc, char *argv[]) {
    cout << "Main: Старт" << endl;

    // Буфер формируемого текста
    char textBuf[200] = {};

    // имя, управляющий флаг, уровень доступа, начальное значение
    sem = sem_open(SEMAPHORE_NAME, O_CREAT, 0777, 0);
    if (sem == SEM_FAILED) {
        perror("Main: Ошибка создания именованного семафора");
        return EXIT_FAILURE;
    }

    // дескриптор нити, атрибутная запись, функция нити, аргументы - буфер
    текста
```

```

pthread_t threadT1;
int threadT1Res = pthread_create(&threadT1, NULL, funcT1, (void*)
textBuf);
if(threadT1Res != 0){
    cout << "Main: Ошибка старта T1 " << strerror(threadT1Res) <<
endl;
    return EXIT_FAILURE;
}
cout << "Main: Поток T1 создан - " << threadT1 << endl;

// дескриптор нити, атрибутная запись, функция нити, аргументы - буфер
текста
pthread_t threadT2;
int threadT2Res = pthread_create(&threadT2, NULL, funcT2, (void*)
textBuf);
if(threadT2Res != 0){
    cout << "Main: Ошибка старта T2 " << strerror(threadT2Res) <<
endl;
    return EXIT_FAILURE;
}
cout << "Main: Поток T2 создан - " << threadT2 << endl;

for (unsigned int i = 0; i < strlen(mainText); i++) {
    appendChar(textBuf, mainText[i]);
}

cout << "Main: Увеличил счётчик семафора" << endl;
int semPost = sem_post(sem);

cout << "Main: Присоединение к T2" << endl;
pthread_join(threadT2, NULL);
cout << "Main: T2 завершился" << endl;

size_t p = strlen(textBuf);
cout << "Main: Буфер сейчас: ";
for(int i = 0; i < p; i++){
    cout << textBuf[i];
}
cout << endl;

sem_close(sem);
cout << "Main: Ok" << endl;
return EXIT_SUCCESS;
}

// Функция потока T1
void* funcT1(void* args) {
    cout << "T1: Старт" << endl;

    char* textBuf = (char*) args;

    cout << "T1: Попытка уменьшить счётчик семафора (блокировка)" << endl;
    sem_wait(sem) ;

    pthread_sleepon_lock();
    cout << "T1: Приступаю к работе" << endl;

    for (unsigned int i = 0; i < strlen(t1Text); i++) {
        appendChar(textBuf, t1Text[i]);
    }

    cout << "T1: Закончил запись текста" << endl;
    isDoneT1 = true;
}

```

```

pthread_sleepon_signal(&isDoneT1);

pthread_sleepon_unlock();

cout << "T1: Ok" << endl;
return EXIT_SUCCESS;
}

// Функция потока T2
void* funcT2(void* args) {
    cout << "T2: Старт" << endl;

    char* textBuf = (char*) args;

    pthread_sleepon_lock();
    // Проверка надо ли ждать и ожидание смены флага
    while(isDoneT1 == false){
        cout << "T2: Ожидание T1" << endl;
        pthread_sleepon_wait(&isDoneT1);
    }

    pthread_sleepon_unlock();

    cout << "T2: Приступаю к работе" << endl;
    for (unsigned int i = 0; i < strlen(t2Text); i++) {
        appendChar(textBuf, t2Text[i]);
    }

    cout << "T2: Ok" << endl;
    return EXIT_SUCCESS;
}

// Запись одного символа с имитацией времени записи
void appendChar(char *text, char c) {
    size_t p = strlen(text);

    text[p] = c;
    text[p + 1] = '\0';

    for (int i = 0; i < 1000000000; i++);
    sleep(1);
}

```