

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](#_Toc147860736)

[2 Результаты работы 4](#_Toc147860737)

[ПРИЛОЖЕНИЕ А Листинг модуля M1 5](#_Toc147860738)

1. Постановка задачи

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

1. М(main).

2. Т1(F1).

3. Т2(F2).

В качестве нити М(main) выступает функция main(). Нити Т1(F1) и Т2(F2) запускаются нитью М(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 М(main) запускает первой нить Т1(F1), затем - Т2(F2), передавая им в качестве параметра указатель совместно формируемого текста.

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

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

Запущенная нить Т1(F1) должна тем же способом добавить в формируемый текст свою часть: "Text1, ".

После завершения записи соей части текста нить Т1(F1) ожидает записи в буфер своей части текста нитью Т2(F2), после чего завершает свою работу.

Запущенная нить Т2(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.

1. Результаты работы

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

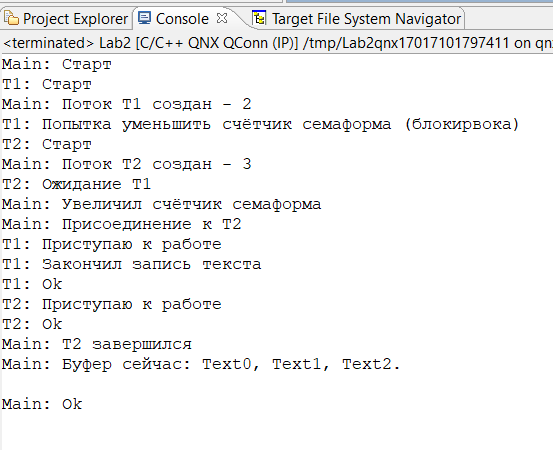


Рисунок 1 – Результат выполнения программы в консоли

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 семаформа");

**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 семаформа" << 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 семаформа (блокирвока)" << 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 < 100000000; i++);

**sleep**(1);

}