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

ИНСТИТУТ ИНФОРМАТИКИ, МАТЕМАТИКИ И ЭЛЕКТРОНИКИ  
Факультет информатики  
Кафедра программных систем

Дисциплина

**Технологии промышленного программирования**

**ОТЧЁТ**  
по лабораторной работе  
**Запуск и синхронизация нитей**

Студент: В.А. Артамонов

Группа: № 6231-020402D

Преподаватель: Баландин А.В.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2022

# Задание

Разработать приложение, состоящее из одного процесса с тремя запущенными нитями: М(main), Т1(F1), 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".

При разработке приложения необходимо обязательно применить следующие методы синхронизации: неименованные семафоры, условные переменные, присоединение. Все методы синхронизации нужно применить даже если их применение будет выглядеть несколько искусственным.

Вариант 7, Задание 1

|  |  |
| --- | --- |
| 7. | Неименованные семафоры. Условные переменные. Присоединение. |

# Порядок выполнения 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.

# Код программы

**#include** <cstdlib>

**#include** <iostream>

**#include** <string.h>

**#include** <semaphore.h>

**#include** <pthread.h>

// Семафор для согласования Мейн и T1

sem\_t SEMAPHORE;

// Условная переменная и мутекс для согласования T1 и T2

pthread\_mutex\_t COND\_MUTEX;

pthread\_cond\_t COND\_VAR;

**bool** HAS\_T1\_FINISHED = **false**;

// Индекс вписывания символа в текст.

**int** TEXT\_INDEX\_FOR\_SYMBOL = 0;

// "Вписывание" символа в конец текста

**void** **append\_symbol\_to\_text**(**char**\* text, **char** symbol) {

// Ждём тысячу итераций

**for** (**int** i = 0; i < 1000; i++);

// Индекс от начала текста, затем разыменование указателя

\*(text + TEXT\_INDEX\_FOR\_SYMBOL) = symbol;

}

**void**\* **t1\_func**(**void**\* args) {

**char**\* text = (**char**\*) args;

**const** **char** T1\_TEXT[] = "Text1, ";

// T1 ждёт, когда Мейн увеличит счётчик, чтобы вписать свою строку

std::cout << "T1 ожидает Мейн" << std::endl;

**sem\_wait**(&SEMAPHORE);

// Захват мутекса для согласования с T2

**pthread\_mutex\_lock**(&COND\_MUTEX);

std::cout << "T1 записывает" << std::endl;

**for** (**unsigned** **int** i = 0; i < strlen(T1\_TEXT); i++) {

append\_symbol\_to\_text(text, T1\_TEXT[i]);

TEXT\_INDEX\_FOR\_SYMBOL++;

}

// Текущая нить закончила ввод своих символов.

// Увеличить счётчик семафора на 1 для согласования хода нити с нитью Мейн

**sem\_post**(&SEMAPHORE);

// Освобождение мутекса для T2

HAS\_T1\_FINISHED = **true**;

// Отправка уведомления по условной переменной

**pthread\_cond\_signal**(&COND\_VAR);

**pthread\_mutex\_unlock**(&COND\_MUTEX);

// Оповещение о конце работы F1

std::cout << "T1 закончил запись" << std::endl;

**return** EXIT\_SUCCESS;

}

**void**\* **t2\_func**(**void**\* args) {

**char**\* text = (**char**\*) args;

**const** **char** T2\_TEXT[] = "Text2.\n\0";

// T2 ждёт, когда T1 освободит мутекс

std::cout << "T2 ожидает T1" << std::endl;

**pthread\_mutex\_lock**(&COND\_MUTEX);

// Ждать, пока T1 нить не закончит запись

**while** (!HAS\_T1\_FINISHED) {

**pthread\_cond\_wait**(&COND\_VAR, &COND\_MUTEX);

}

std::cout << "T2 записывает" << std::endl;

**for** (**unsigned** **int** i = 0; i < strlen(T2\_TEXT) + 1; i++) {

append\_symbol\_to\_text(text, T2\_TEXT[i]);

TEXT\_INDEX\_FOR\_SYMBOL++;

}

// Отправка уведомления по условной переменной

**pthread\_cond\_signal**(&COND\_VAR);

**pthread\_mutex\_unlock**(&COND\_MUTEX);

std::cout << "T2 закончил запись" << std::endl;

**return** EXIT\_SUCCESS;

}

**int** **main**(**int** argc, **char** \*argv[]) {

// "Общая" строка трёх нитей.

// В конце программы должна содержать "Text0, Text1, Text2.\n".

// 22 символов, потому что вместе с признаком конца строки '\0'.

**char** text[100];

// Создание семафора.

// 0 – семафор не разделяется нитями других процессов,

// 0 – значение семафора.

**sem\_init**(&SEMAPHORE, 0, 0);

// Создание мутекса и условной переменной

**pthread\_mutex\_init**(&COND\_MUTEX, NULL);

**pthread\_cond\_init**(&COND\_VAR, NULL);

// Создание и запуск нити T1

pthread\_t t2\_thread;

**pthread\_create**(&t2\_thread, NULL, (**void**\*(\*)(**void**\*)) t2\_func, (**void**\*) text);

pthread\_t t1\_thread;

**pthread\_create**(&t1\_thread, // дескриптор нити

NULL, // атрибутная запись

(**void**\*(\*)(**void**\*)) t1\_func, // функция нити

(**void**\*) text); // аргументы функции нити

// Создание и запуск нити T2

// Символы, которые должна вписать в text текущая нить main

**const** **char** M\_TEXT[] = "Text0, ";

// Запись символов в текст нитью Мейн.

// Цикл по кол-ву символов в M\_TEXT ("Text0, ")

std::cout << "Мейн записывает" << std::endl;

**for** (**unsigned** **int** i = 0; i < strlen(M\_TEXT); i++) {

append\_symbol\_to\_text(text, M\_TEXT[i]);

// В след. раз вписать в след. позицию

TEXT\_INDEX\_FOR\_SYMBOL++;

}

// Текущая нить закончила ввод своих символов,

// увеличить счётчик семафора на 1

**sem\_post**(&SEMAPHORE);

//Мейн ждёт, когда T1 впишет свои символы

std::cout << "МЕйн ожидает T1" << std::endl;

**sem\_wait**(&SEMAPHORE);

// Удалить семафор

**sem\_destroy**(&SEMAPHORE);

// Мейн ждёт, когда T2 впишет символы,

// согласование Мейн и T2

**void** \*value\_ret;

std::cout << "Мейн ожидает T2" << std::endl;

**pthread\_join**(t2\_thread, &value\_ret);

// Вывод результата согласованной работы трёх нитей

std::cout << text << std::endl;

// Оповещение о конце работы нити Мейн

std::cout << "Мейн закончил запись" << std::endl;

**return** EXIT\_SUCCESS;

}