**Отчет лабораторной работы №2**

**Тема « Ожидаемые и отсоединенные потоки»**

**Султанов Шакир**

Студенты должны знать:

1. Атрибуты потока;
2. Ожидаемые и отсоединенные потоки;
3. Назначение фунций: pthread\_detach(), pthread\_attr\_setdetachstate() , pthread\_attr\_init(), pthread\_attr\_destroy().

**Задания**

1. Написать консольное параллельное приложение Alarm­\_clock (будильник). Главный поток в цикле считывает с консоли целые числа. Если введено отрицательное число, главный поток выходит из цикла, ждет нажатия произвольной клавиши, после чего завершает работу приложения. Если введено положительное число x, главный поток запускает параллельный отсоединенный поток, передавая в поток через параметр потоковой функции число x. Созданный поток «засыпает» на x секунд, затем выводит сообщение на консоль и завершает работу.
2. Используя потоковые данные, реализовать в написанном параллельном приложении следующие требования. Каждый созданный параллельный поток должен:

* вывести в отдельный файл сообщение о начале работы и число x;
* сообщение о завершении работы;
* закрыть файл при завершении.

**Код:**

**#include <pthread.h>**

**#include <stdio.h>**

**unsigned int sleep(unsigned int seconds);**

**pthread\_key\_t fkey;**

**void write\_to\_file(char\* message) {**

**FILE\* file = (FILE\*) pthread\_getspecific(fkey);**

**fprintf(file, "%s\n", message);**

**}**

**void close\_file(void\* file) {**

**fclose((FILE\*) file);**

**}**

**void\* alarm(void\* c) {**

**char filename[25];**

**FILE\* file;**

**sprintf(filename, "thread%d.log", (int)pthread\_self());**

**file = fopen(filename, "w");**

**pthread\_setspecific(fkey, file);**

**int count = \*(int\*)c;**

**write\_to\_file("Thread started working...\n");**

**sleep(count);**

**char message[25];**

**sprintf(message, "Waited for %d seconds\n", count);**

**printf("%s", message);**

**write\_to\_file(message);**

**return NULL;**

**}**

**int main() {**

**pthread\_key\_create(&fkey, close\_file);**

**int num = 0;**

**pthread\_t t1;**

**pthread\_attr\_t attr1;**

**pthread\_attr\_init(&attr1);**

**pthread\_attr\_setdetachstate(&attr1, PTHREAD\_CREATE\_DETACHED);**

**scanf("\n %d", &num);**

**while (num >= 0) {**

**pthread\_create(&t1, &attr1, alarm, (void\*)&num);**

**scanf("%d", &num);**

**}**

**pthread\_attr\_destroy(&attr1);**

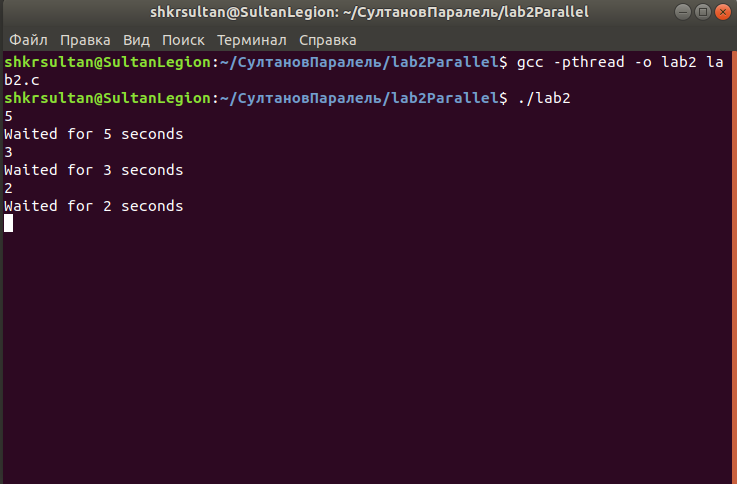
**printf("Press to finish\n");**

**scanf("%d", &num);**

**return 0;**

**}**

**Результат:**

****