

Міністерство освіти і науки України
Львівський національний університет імені Івана Франка

Факультет електроніки та комп'ютерних технологій
Кафедра радіоелектронних і комп'ютерних систем

Звіт

Про виконання лабораторної роботи №5
«Керування процесами і потоками»

Виконала:
Студентка групи ФЕІ-23
Лісова С.О.
Викладач:
Сінькевич О.О.

Львів – 2019

Тема: Керування процесами і потоками

Мета: Вивчення та застосування програмних інтерфейсів ОС для керування процесами та потоками.

Завдання №1. Напишіть функцію, виклик якої приведе до знищення всіх процесів-зомбі, створених поточним процесом.

Порядок виконання роботи:

- Створюю програму з реалізацією функцій для створення та видалення потоків зомбі.
- Код програми:

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/wait.h>
#include <unistd.h>

void killZombie(){
    pid_t pid;
    int status;
    printf("\n");
    while((pid = wait(&status)) > 0){
        printf("Zombie process with PID %d terminated with status %d\n", pid, status);
    }
    printf("\n");
}

void zombieList(){
    pid_t pid;

    if ((pid = fork()) == -1)
        exit(-1);

    if (pid == 0){ //child
        char* args[] = {"ps", "uf", "-C", "zombie", NULL};
        execvp("ps", args);
        // exec return if someting wrong
        exit(-1);
    } else { //parent
        int status;
        waitpid(pid, &status, NULL);
        if (!WIFEXITED(status)){
            exit(-1);
        }
    }
}

int main() {
    pid_t pid;
    int i;

    for (i = 1; ; ++i){
```

```

        if ((pid = fork()) == -1) {
            return -1;
        }
    else if (pid > 0) { //parent
        if (i == 5){
            zombieList();
            killZombie();
            zombieList();

            exit(0);
        }
    } else { //child
        printf("");
        exit(0);
    }
}
return 0;
}

```

- **Результати виконання програми:**

```
Solomiyas-MacBook-Pro:task1 consolkaaa$ gcc zombiemaker.c -o zombiemaker
```

```

task1 — -bash — 80x13
Solomiyas-MacBook-Pro:task1 consolkaaa$ ./zombiemaker
ps: illegal option -- f
usage: ps [-AaCcEefhjLMmrSTvwXx] [-O fmt | -o fmt] [-G gid[,gid...]]
        [-u]
        [-p pid[,pid...]] [-t tty[,tty...]] [-U user[,user...]]
        ps [-L]

Zombie process with PID 994 terminated with status 0
Zombie process with PID 993 terminated with status 0
Zombie process with PID 992 terminated with status 0
Zombie process with PID 991 terminated with status 0
Zombie process with PID 990 terminated with status 0

```

Завдання №2. Розробіть простий командний інтерпретатор для Linux і Windows. Він повинен видавати підказку (наприклад, «>»), обробляти введенний користувачем командний рядок (що містить ім'я виконуваного файлу програми та її аргументи) і запускати задану програму.

Порядок виконання роботи:

- Створюю простий командний інтерпретатор для Linux і Windows.
- **Код програми:**

```
#include <stdio.h>
```

```

#include <signal.h>
#include <stdlib.h>
#include <string.h>
#include <sys/wait.h>
#include <unistd.h>

pid_t pid;

void exeCommand(char* command){
    char* argv[20];
    int beakground;

    char* token;
    token = strtok(command, " ");
    int i=0;
    for(;token!=NULL;i++){
        argv[i] = token;
        token = strtok(NULL, " ");
    }
    if(!strcmp(argv[i-1],"&")){
        beakground = 1;
        argv[i-1] = NULL;
    }
    else{
        beakground = 0;
        argv[i] = NULL;
    }

    if((pid=fork())== -1) exit(1);
    if(pid==0){
        execvp(argv[0],argv);
        exit(1);
    }
    if(!beakground) wait(NULL);
}

static void catch_function(int signal){
    kill(pid,SIGINT);
    printf("\n");
}

int main(){
    printf("myShell started\n");
    signal(SIGINT, catch_function);
    char command[5];
    while(1){
        printf(">");
        //fgets(command, 3, stdin);/////
        gets(command);

        if(command[0]=='\0') continue;
        if(!strcmp(command, "exit")) break;
        exeCommand(command);
    }
    printf("You are exit!\n");
    return 0;
}

```

- **Результат виконання програми:**

```
lab5 — interpretator — 80x10
Solomiyas-MacBook-Pro:Lab5 consolkaaa$ gcc lab5.1.c -o interpretator
Solomiyas-MacBook-Pro:Lab5 consolkaaa$ ./interpretator
myShell started
warning: this program uses gets(), which is unsafe.
>pwd
/Users/consolkaaa/Desktop/Lab5
>ls
a.out          lab5.1.c      lab5.2.c      lab5.c        potoku
interpretator  lab5.1.o      lab5.2.o      lab5.o        zombie
>
```

Завдання №3. Розробіть застосування для Linux і Windows, що реалізує паралельне виконання коду двома потоками. Основний потік застосування Т створює потік t Далі кожен із потоків виконує цикл (наприклад, до 30). На кожній ітерації циклу він збільшує значення локального лічильника на одиницю, відображає це значення з нового рядка і призупиняється на деякий час (потік Т– на час w Т , потік t – w t). Після завершення циклу потік Т приєднує t.

Порядок виконання роботи:

- Створюю застосування в текстовому редакторі gedit.
- Код програми:

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

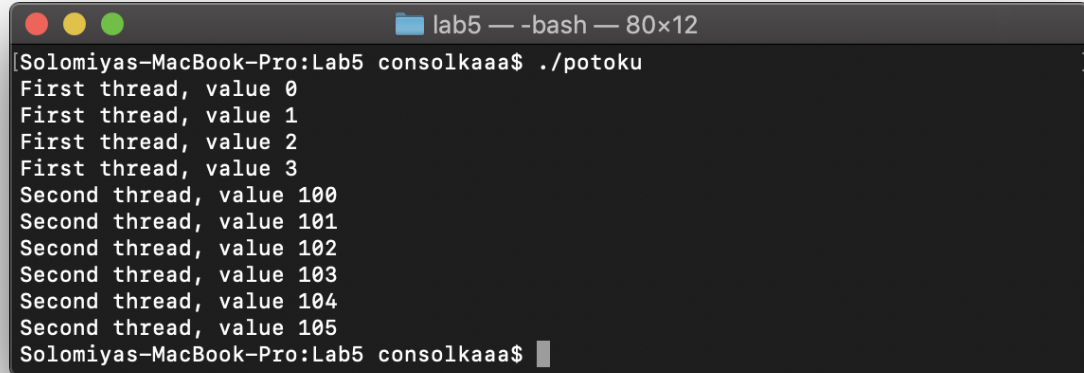
void *anotherThread(void *value){
    int i=0;
    for(i=100;i<106;i++){
        printf("Second thread, value %d\n",i);
        sleep(2);
    }
}

int main(){
    pthread_t thread;
    pthread_create(&thread,NULL,anotherThread,NULL);

    int i=0;
    for(i=0;i<4;i++){
        printf("First thread, value %d\n",i);
        sleep(1);
    }
    pthread_join(thread,NULL);
    return 0;
}
```

- Результат виконання програми:

```
Solomiyas-MacBook-Pro:Lab5 consolkaaa$ gcc -pthread lab5.2.c -o potoku
```

A terminal window titled 'lab5 — -bash — 80x12' showing the execution of the 'potoku' program. The output displays two threads: 'First thread' and 'Second thread'. The first thread prints values 0, 1, 2, and 3. The second thread prints values 100, 101, 102, 103, 104, and 105. The prompt returns to the shell.

```
Solomiyas-MacBook-Pro:Lab5 consolkaaa$ ./potoku
First thread, value 0
First thread, value 1
First thread, value 2
First thread, value 3
Second thread, value 100
Second thread, value 101
Second thread, value 102
Second thread, value 103
Second thread, value 104
Second thread, value 105
Solomiyas-MacBook-Pro:Lab5 consolkaaa$
```

Висновок: виконуючи лабораторну роботу, я навчилась працювати з потоками та написала програму, яка виводить числа в два потоки.