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);

}

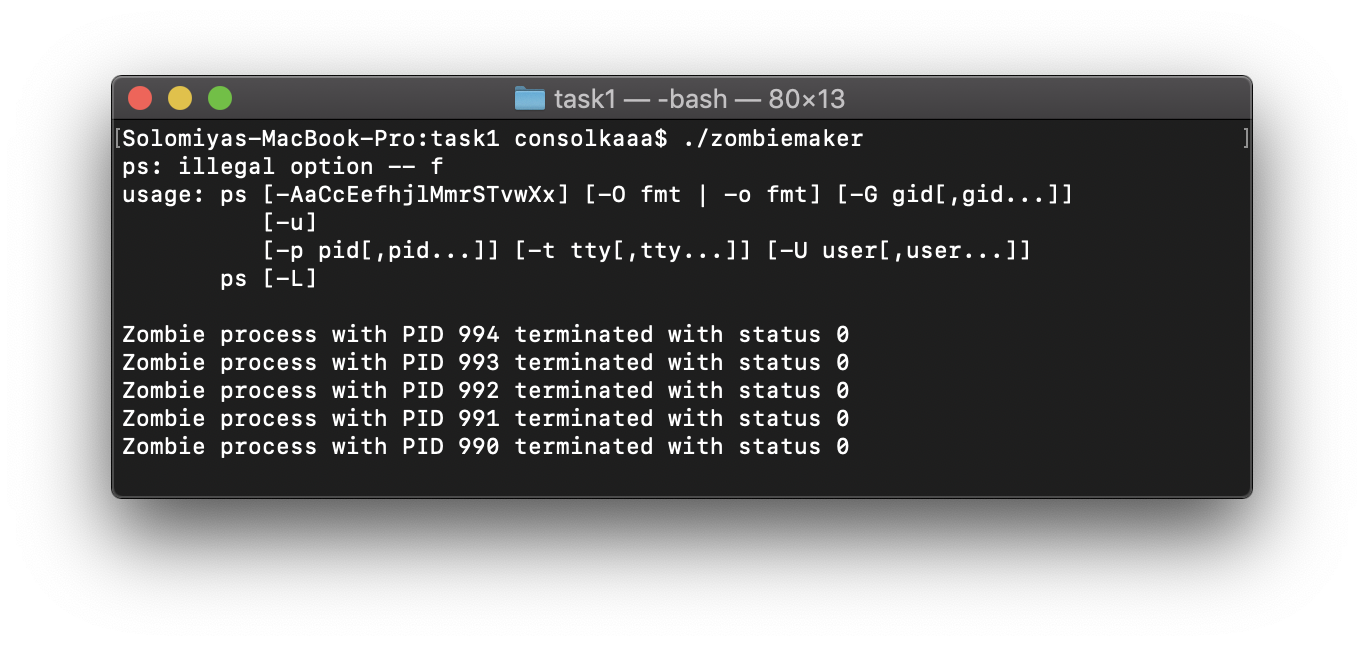
}

return 0;

}

* **Результати виконання програми:**



****

**Завдання №2.** Розробіть простий командний інтерпретатор для Linux і Windows. Він повинен видавати підказку (наприклад, «>»), обробляти введений 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);

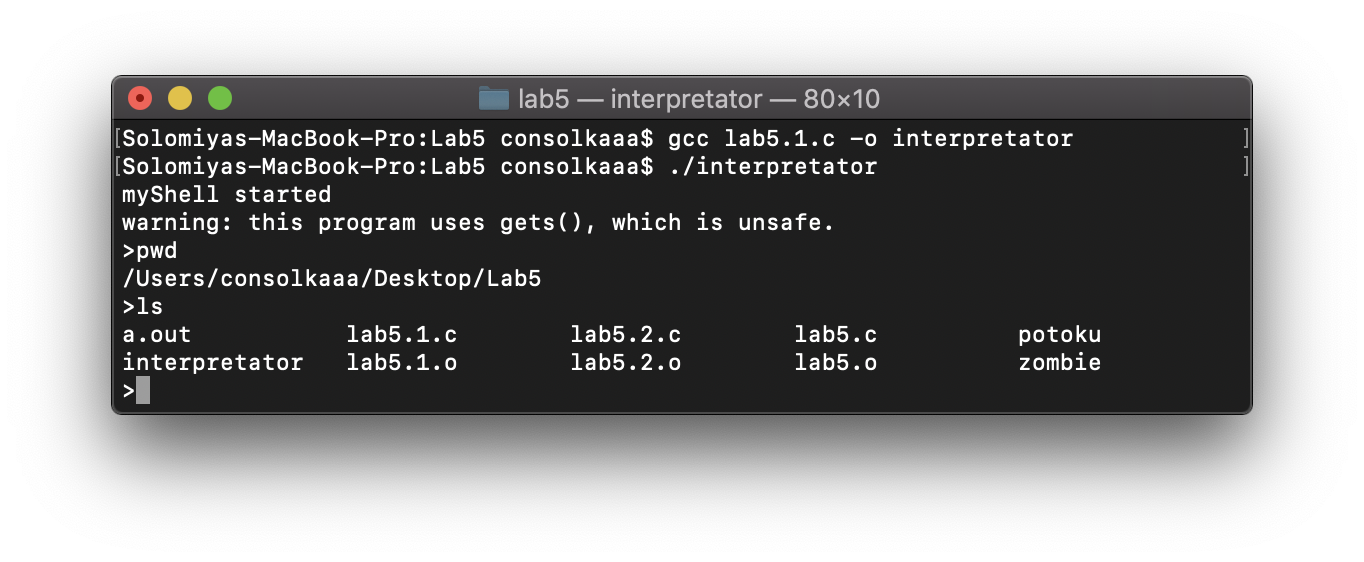
}

printf("You are exit!\n");

return 0;

}

* **Результат виконання програми:**

****

**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 T , потік t – w t ). Після завершення циклу потік T 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);

}

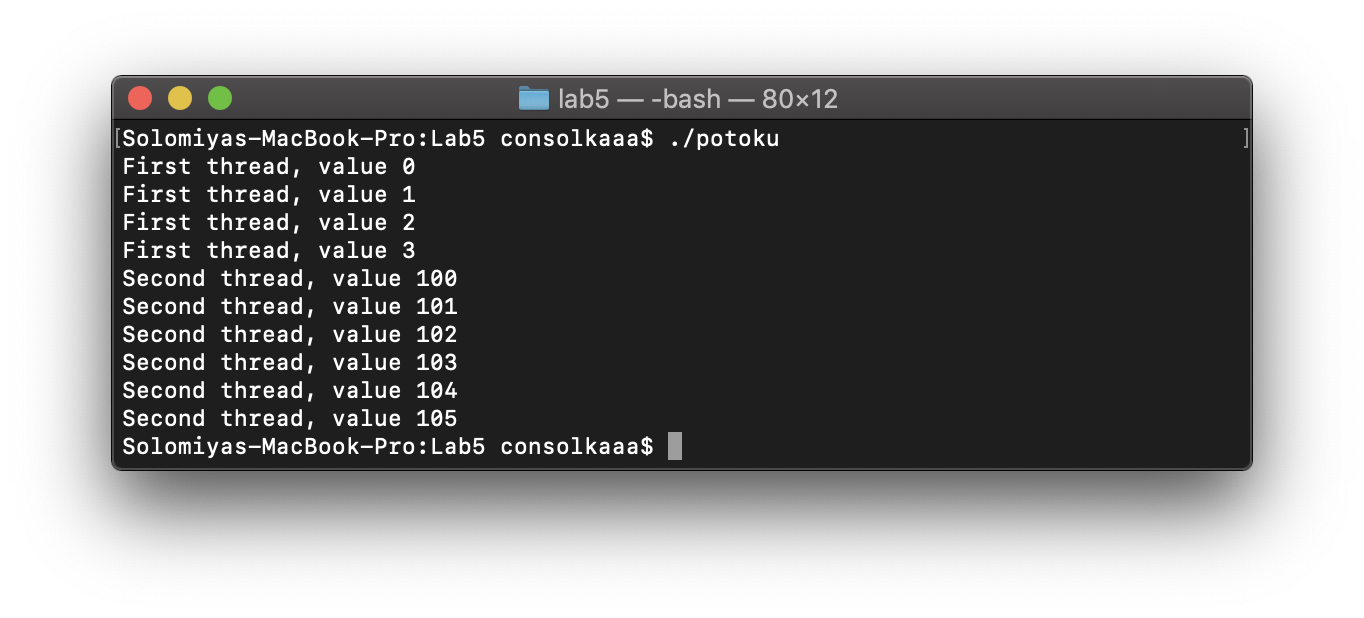
pthread\_join(thread,NULL);

return 0;

}

* **Результат виконання програми:**

****

****

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