Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación

SISTEMAS OPERATIVOS I

PRACTICA 6: HILOS

Docente:
Prof. Marcos González Flores

Alumno:	Matricula:
Jesús Huerta Aguilar	202041509
Alex Abdiel Ruano Flores	202075025

NRC: 46152
Sección: 003

QUINTO SEMESTRE

Puebla, Pue. 02/03/2022

Practica No 6

Nombre: Hilos

Objetivos: Aprender a crear hilos a crear atributos y a pasar parámetros

Desarrollo:

PRACTICA 1

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

void *functionThread(void *parametro);

int contador = 100;

main(){
	pthread_t iHilo;
	int error;
	int i;
	/*Pasamos atributos del nuevo thread NULL para que  los atrib
		por default
	Pasamos la funcion que se ejecutara en el nuevo hilo
	Pasamos NULL como parmaetro para esa funcion*/

	error = pthread_create(&iHilo,NULL,functionThread,NULL);

	if(error != 0){
		perror("No puedo crear thread");
		exit(-1);
	}

	for(i=0;i<10;i++){
		contador++;
		printf("\nPadre: %d\n",contador);
		sleep(1);
	}
	exit(0);
}
	/* Funcion que se ejecuta en el thread hijo*/

void *functionThread(void *parametro){
	int i;
	/*Bucle infinito para decrementar contado y mostrarlo en pantalla*/
	for(i=0;i<10;i++){
		contador--;
		printf("\nHijo: %d\n", contador);
		sleep(1);
	}
	exit(0);
}
```

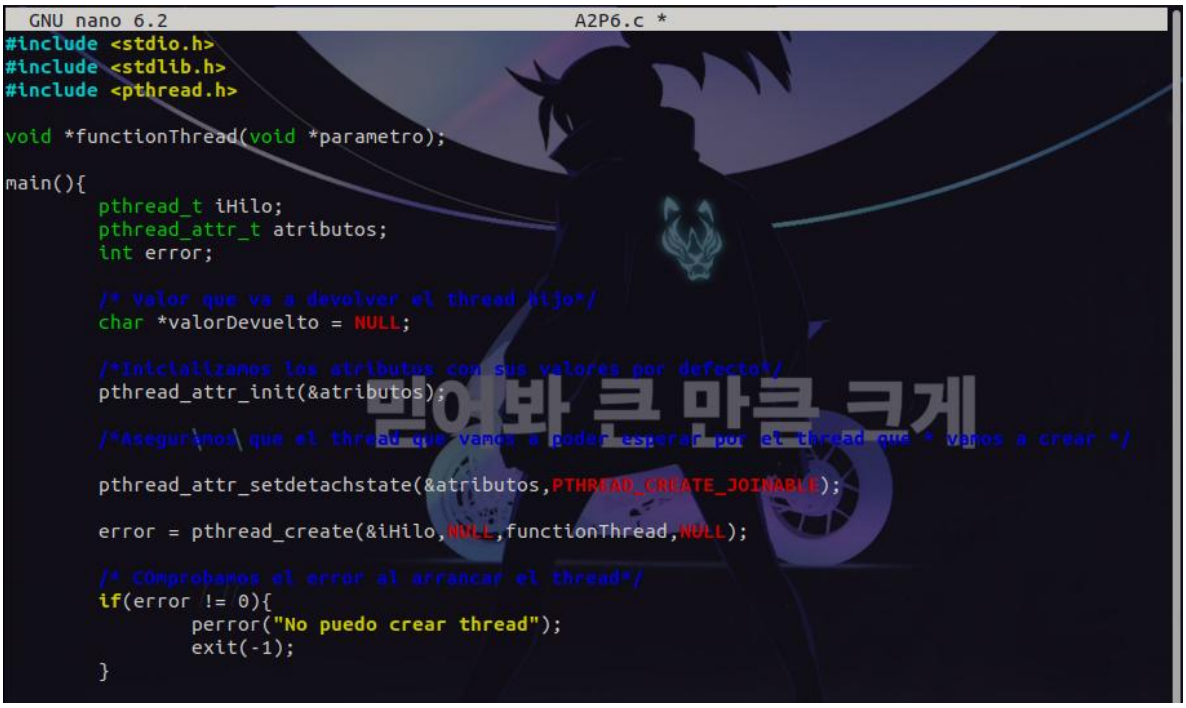
TRANSCRITO + EJECUCIÓN:


Padre: 101
Hijo: 100
Padre: 101
Hijo: 100
Padre: 101
Hijo: 100
Padre: 101
Hijo: 100
Padre: 101
Hijo: 100
Padre: 101
Hijo: 100
Padre: 101
Hijo: 100
Padre: 101
Hijo: 100
planisphere@planisphere-VirtualBox:~$
믿어봐 큰 만큼 크게

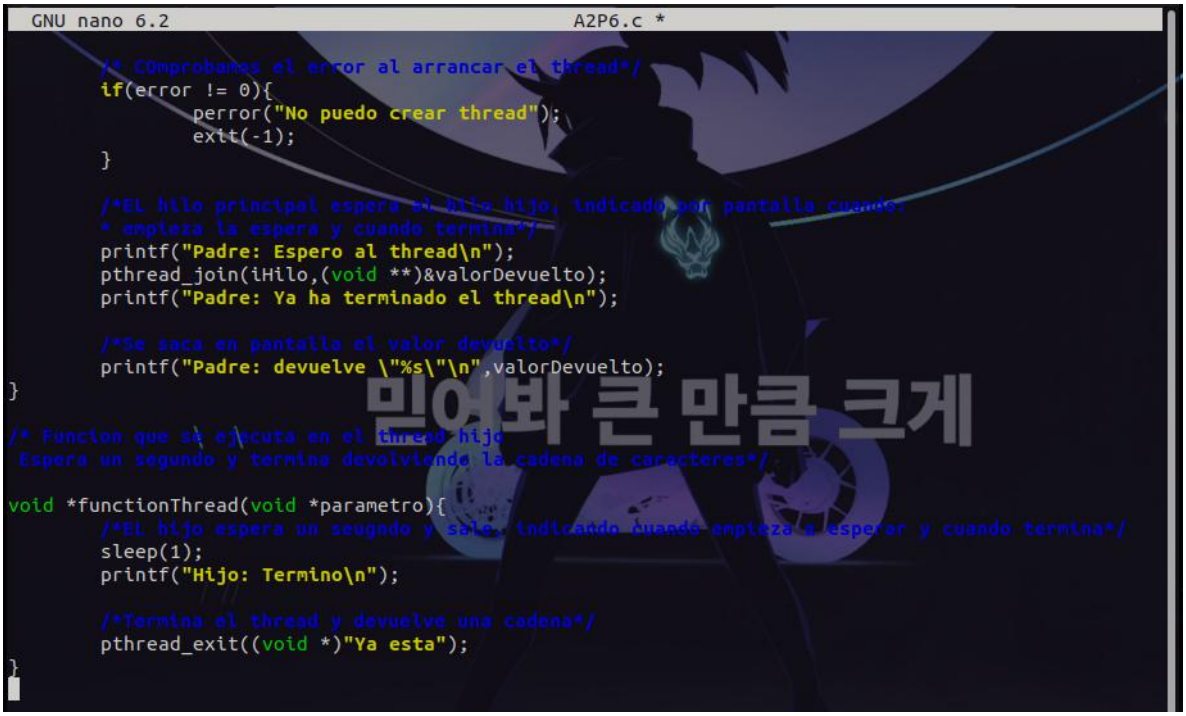
PRACTICA 2

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

void *functionThread(void *parametro);

main(){
	pthread_t iHilo;
	pthread_attr_t atributos;
	int error;

	/* Valor que va a devolver el thread hijo*/
	char *valorDevuelto = NULL;

	/*Inicializamos los atributos con sus valores por defecto*/
	pthread_attr_init(&atributos);

	/*Aseguramos que el thread que vamos a poder esperar por el thread que *
vamos a crear */

	pthread_attr_setdetachstate(&atributos,PTHREAD_CREATE_JOINABLE);

	error = pthread_create(&iHilo,NULL,functionThread,NULL);

	/* COmprobamos el error al arrancar el thread*/
	if(error != 0){
		perror("No puedo crear thread");
		exit(-1);
	}

	/*EL hilo principal espera al hilo hijo, indicado por pantalla cuando:
	* empieza la espera y cuando termina*/
	printf("Padre: Espero al thread\n");
	pthread_join(iHilo,(void **)&valorDevuelto);
	printf("Padre: Ya ha terminado el thread\n");

	/*Se saca en pantalla el valor devuelto*/
	printf("Padre: devuelve \"%s\"\n",valorDevuelto);
}

/* Funcion que se ejecuta en el thread hijo
 Espera un segundo y termina devolviendo la cadena de caracteres*/

void *functionThread(void *parametro){
	/*EL hijo espera un seugndo y sale, indicando cuando empieza a esperar y
cuando termina*/
	sleep(1);
	printf("Hijo: Termino\n");

	/*Termina el thread y devuelve una cadena*/
	pthread_exit((void *)"Ya esta");
}
```

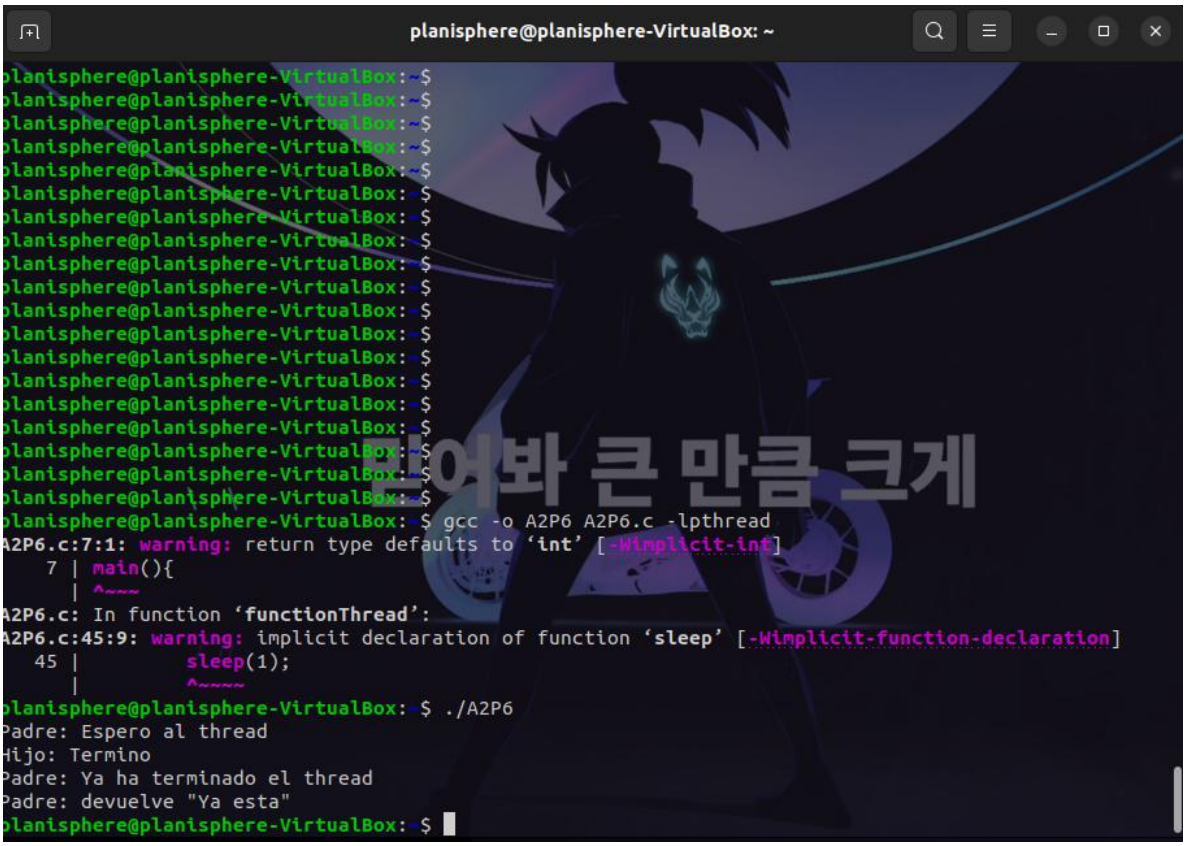
TRANSCRITO + EJECUCIÓN:



```
 GNU nano 6.2                                  A2P6.c *
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

void *functionThread(void *parametro);

main(){
        pthread_t iHilo;
        pthread_attr_t atributos;
        int error;

        /* Valor que va a devolver el thread hijo*/
        char *valorDevuelto = NULL;

        /*Inicializamos los atributos con sus valores por defecto*/
        pthread_attr_init(&atributos);

        /*Aseguramos que el thread que vamos a poder esperar por el thread que * vamos a crear */

        pthread_attr_setdetachstate(&atributos,PTHREAD_CREATE_JOINABLE);

        error = pthread_create(&iHilo,NULL,functionThread,NULL);

        /* COmprobamos el error al arrancar el thread*/
        if(error != 0){
                perror("No puedo crear thread");
                exit(-1);
        }
```



```
 GNU nano 6.2                                  A2P6.c *
        /* COmprobamos el error al arrancar el thread*/
        if(error != 0){
                perror("No puedo crear thread");
                exit(-1);
        }

        /*El hilo principal espera al hilo hijo, indicado por pantalla cuando
        * empieza la espera y cuando termina*/
        printf("Padre: Espero al thread\n");
        pthread_join(iHilo,(void **)&valorDevuelto);
        printf("Padre: Ya ha terminado el thread\n");

        /*Se saca en pantalla el valor devuelto*/
        printf("Padre: devuelve \"%s\"\n",valorDevuelto);
}

/* Funcion que se ejecuta en el thread hijo
 Espera un segundo y termina devolviendo la cadena de caracteres*/

void *functionThread(void *parametro){
        /*El hijo espera un segundo y sale, indicando cuando empieza a esperar y cuando termina*/
        sleep(1);
        printf("Hijo: Termino\n");

        /*Termina el thread y devuelve una cadena*/
        pthread_exit((void *)"Ya esta");
}
```


planisphere@planisphere-VirtualBox: ~
planisphere@planisphere-VirtualBox:~$ gcc -o A2P6 A2P6.c -lpthread
A2P6.c:7:1: warning: return type defaults to 'int' [-Wimplicit-int]
7 | main(){
| ^~~~
A2P6.c: In function 'functionThread':
A2P6.c:45:9: warning: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
45 | sleep(1);
| ^~~~~
planisphere@planisphere-VirtualBox:~$ ./A2P6
Padre: Espero al thread
Hijo: Termino
Padre: Ya ha terminado el thread
Padre: devuelve "Ya esta"
planisphere@planisphere-VirtualBox:~$

PRACTICA 3

```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#define NUM_THREADS 5

// La funcion que es utilizada en un hijo
// Debera de retornar un puntero sin tipo
void *ImprimeSaludo(void *threadid){
      long tid;
      // Aqui se castea lo que fue pasado
      // Al hilo como atributos
      tid = (long)threadid;
      printf("Hola esto se ejecuta en el hilo #%ld\n",tid);
      pthread_exit(NULL);
}

// ---Aqui finaliza la funcion que
// sera lamada en el hilo

int main (int argc, char *argv[]){
      pthread_t threads[NUM_THREADS];
      int rc;
      long t;
      for(t=0; t < NUM_THREADS; t++){
            printf("En Funcion main: creando el hilo %ld\n",t);
            rc = pthread_create(&threads[t],NULL,ImprimeSaludo,(void
*)t);
            if(rc){
                  printf("ERROR: Return code from pthread_create() is
&dn",rc);
                  exit(-1);
            }
      }
      pthread_exit(NULL);
}
```

TRANSCRITO + EJECUCIÓN:

```
  GNU nano 6.2                                  A3P6.c
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#define NUM_THREADS 5

// La funcion que es utilizada en un hijo
// Debera de retornar un puntero sin tipo
void *ImprimeSaludo(void *threadid){
        long tid;
        // Aqui se castea lo que fue pasado
        // Al hilo como atributos
        tid = (long)threadid;
        printf("Hola esto se ejecuta en el hilo #%ld\n",tid);
        pthread_exit(NULL);
}

// ---Aqui finaliza la funcion que
// sera lamada en el hilo

int main (int argc, char *argv[]){
        pthread_t threads[NUM_THREADS];
        int rc;
        long t;
        for(t=0; t < NUM_THREADS; t++){
                printf("En Funcion main: creando el hilo %ld\n",t);
                rc = pthread_create(&threads[t],NULL,ImprimeSaludo,(void *)t);
                if(rc){
                        printf("ERROR: Return code from pthread_create() is &dn",rc);
                        exit(-1);
                }
        }
        pthread_exit(NULL);
```

```
planisphere@planisphere-VirtualBox: ~
planisphere@planisphere-VirtualBox:~$ nano A3P6.c
planisphere@planisphere-VirtualBox:~$ gcc -o A3P6 A3P6.c
A3P6.c: In function ‘main’:
A3P6.c:28:32: warning: too many arguments for format [-Wformat-extra-args]
   28 |                         printf("ERROR: Return code from pthread_create() is &dn",rc);
      |                                ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
planisphere@planisphere-VirtualBox:~$ ./A3P6
En Funcion main: creando el hilo 0
En Funcion main: creando el hilo 1
En Funcion main: creando el hilo 2
En Funcion main: creando el hilo 3
En Funcion main: creando el hilo 4
Hola esto se ejecuta en el hilo #1
Hola esto se ejecuta en el hilo #2
Hola esto se ejecuta en el hilo #3
Hola esto se ejecuta en el hilo #4
Hola esto se ejecuta en el hilo #0
planisphere@planisphere-VirtualBox:~$
```