Operating systems

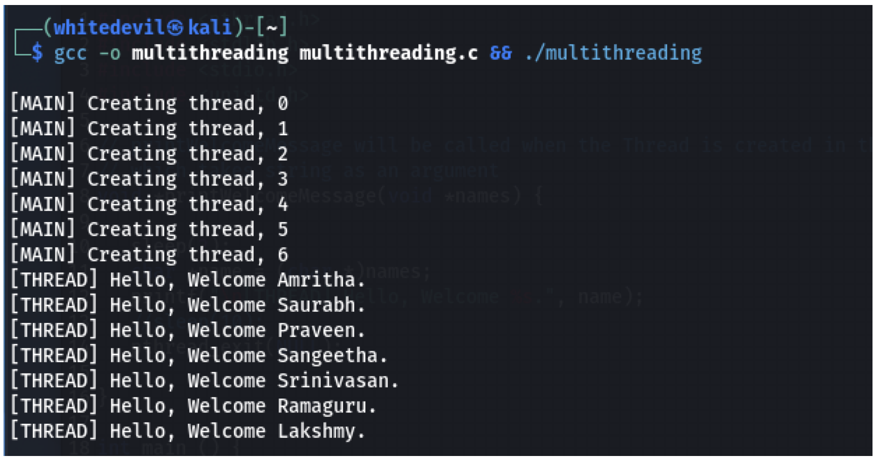
LAB-06

Name: G H Hem Sagar
Roll no: CB.EN.U4CYS21016

1. Multithreading in C:

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

// printWelcomeMessage will be called when the Thread is created in the main function
// which takes string as an argument
void *printWelcomeMessage(void *names) {

  sleep(2);
  char *name = (char *)names;
  printf("\n[THREAD] Hello, Welcome %s.", name);
  //sleep(10);
  pthread_exit(NULL);

}

int main () {

  // thread defintion
  pthread_t threads[7];

  // parameter to be passed to the called function - printWelcomeMessage
  char names[10][15] =
{"Amritha","Praveen","Saurabh","Sangeetha","Lakshmy","Srinivasan","Ramaguru"};

  int result;

  for(int i = 0; i < 7; i++ ) {

    printf("\n[MAIN] Creating thread, %d", i);
    //printf("%d",i);

    // Creating the threading and thus calling the function with parameter passed to it
    result = pthread_create(&threads[i], NULL, printWelcomeMessage, (void *)names[i]);
   // printf("\t thread id : %d \t result: %d",threads[i],result);
    if (result) {

      printf("Error in creating thread, %d ", result);
      exit(-1);
    }

  }

  // Exit the thread
  pthread_exit(NULL);
}
```

Output:

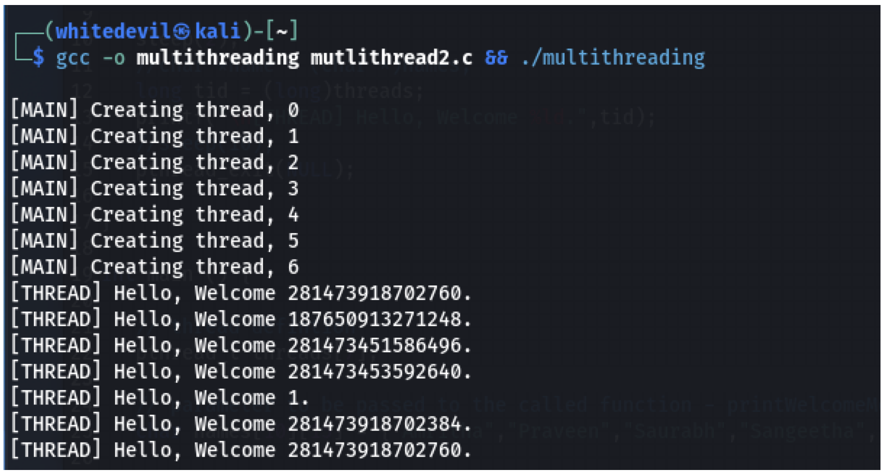


```
┌──(whitedevil㉿kali)-[~]
└─$ gcc -o multithreading multithreading.c && ./multithreading

[MAIN] Creating thread, 0
[MAIN] Creating thread, 1
[MAIN] Creating thread, 2
[MAIN] Creating thread, 3
[MAIN] Creating thread, 4
[MAIN] Creating thread, 5
[MAIN] Creating thread, 6
[THREAD] Hello, Welcome Amritha.
[THREAD] Hello, Welcome Saurabh.
[THREAD] Hello, Welcome Praveen.
[THREAD] Hello, Welcome Sangeetha.
[THREAD] Hello, Welcome Srinivasan.
[THREAD] Hello, Welcome Ramaguru.
[THREAD] Hello, Welcome Lakshmy.
```

2. Multithreading displaying thread id:

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

// printWelcomeMessage will be called when the Thread is created in the main function
// which takes string as an argument
void *printWelcomeMessage(void *threads) {

  sleep(2);
  //char *name = (char *)names;
  long tid = (long)threads;
  printf("\n[THREAD] Hello, Welcome %ld.",tid);
  //sleep(10);
  pthread_exit(NULL);

}

int main () {

  // thread defintion
  pthread_t threads[7];

  // parameter to be passed to the called function - printWelcomeMessage
  char names[10][15] =
{"Amritha","Praveen","Saurabh","Sangeetha","Lakshmy","Srinivasan","Ramaguru"};

  int result;

  for(int i = 0; i < 7; i++ ) {

    printf("\n[MAIN] Creating thread, %d", i);
    //printf("%d",i);

    // Creating the threading and thus calling the function with parameter passed to it
    result = pthread_create(&threads[i], NULL, printWelcomeMessage, (void *)threads[i]);
   // printf("\t thread id : %d \t result: %d",threads[i],result);
    if (result) {

      printf("Error in creating thread, %d ", result);
      exit(-1);
    }

  }

  // Exit the thread
  pthread_exit(NULL);
}
```

Output:



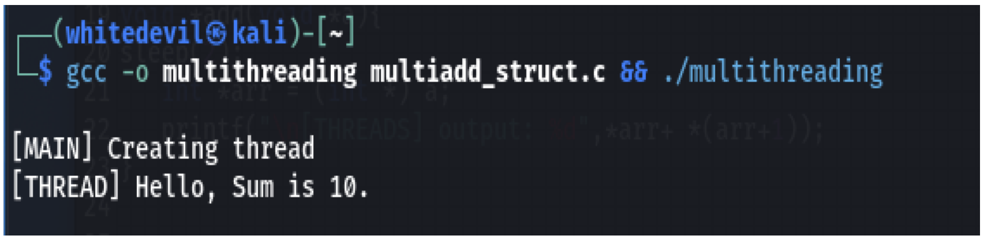
```
┌──(whitedevil㉿kali)-[~]
└─$ gcc -o multithreading mutlithread2.c && ./multithreading

[MAIN] Creating thread, 0
[MAIN] Creating thread, 1
[MAIN] Creating thread, 2
[MAIN] Creating thread, 3
[MAIN] Creating thread, 4
[MAIN] Creating thread, 5
[MAIN] Creating thread, 6
[THREAD] Hello, Welcome 281473918702760.
[THREAD] Hello, Welcome 187650913271248.
[THREAD] Hello, Welcome 281473451586496.
[THREAD] Hello, Welcome 281473453592640.
[THREAD] Hello, Welcome 1.
[THREAD] Hello, Welcome 281473918702384.
[THREAD] Hello, Welcome 281473918702760.
```

3. Multithreading with addition:

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

struct add {
   int a;
   int b;
};
```

```
void *printWelcomeMessage(void * var) {

    sleep(1);
    struct add *obj = var;
    int sum = obj->a + obj->b;
    printf("\n[THREAD] Hello, Sum is %d.", sum);
    pthread_exit(NULL);

}

int main () {

   // thread defintion
   pthread_t threads;

   struct add var;
   var.a = 5;
   var.b = 5;

   int result;


      printf("\n[MAIN] Creating thread");
      // Creating the threading and thus calling the function with parameter passed to it
      result = pthread_create(&threads, NULL, printWelcomeMessage, &var);


   // Exit the thread
   pthread_exit(NULL);
  return 0;
}
```

Output:



```
┌──(whitedevil㉿kali)-[~]
└─$ gcc -o multithreading multiadd_struct.c && ./multithreading

[MAIN] Creating thread
[THREAD] Hello, Sum is 10.
```