Name: Shangirne Kharbanda

Registration Number: 20BAI1154

OS LAB-6

1) **Create two threads thread1 and thread2 and call functions fun1 and fun2 respectively.**

Code:

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

void *print_message_function(void *ptr);

int main(){

pthread_t thread1, thread2;
char *message1 = "Thread 1";
char *message2 = "Thread 2";
int iret1, iret2;

iret1 = pthread_create(&thread1, NULL, print_message_function, (void*) message1);
iret2 = pthread_create(&thread2, NULL, print_message_function, (void*) message2);

pthread_join(thread1, NULL);
pthread_join(thread2, NULL);

printf("Thread 1 returns: %d\n", iret1);
printf("Thread 2 returns: %d\n", iret2);
exit(0);
}

void *print_message_function(void *ptr){
char *message;
message= (char *) ptr;
printf("%s \n", message);
}

```

Output:

```
alaric@alaric-virtual-machine:~/Desktop$ gcc ques1.c -lpthread
alaric@alaric-virtual-machine:~/Desktop$ ./a.out
Thread 1
Thread 2
Thread 1 returns: 0
Thread 2 returns: 0
```

2) Create two threads thread1 and thread2 and call functions fun1 and fun2 respectively. Compute and print Fibonacci in fun1 and square of a number in fun2.

Code:

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

void *fun1(){
 int i, n;
  int t1 = 0, t2 = 1;
  int nextTerm = t1 + t2;

  printf("Enter the number of terms: ");
  scanf("%d", &n);

  printf("Fibonacci Series: %d, %d, ", t1, t2);

  for (i = 3; i <= n; ++i) {
    printf("%d, ", nextTerm);
    t1 = t2;
    t2 = nextTerm;
    nextTerm = t1 + t2;
    }

  }

 void *fun2(){
 int n, square;
 printf("Enter the number that you want to square: ");
 scanf("%d", &n);


 square= n*n;
 printf("Square of %d is %d", n, square);

}

  int main(){
  pthread_t thread1, thread2;
   int iret1, iret2;

  iret1= pthread_create(&thread1, NULL, fun1, NULL);
  iret2= pthread_create(&thread2, NULL, fun2, NULL);

  pthread_join(thread1, NULL);
  pthread_join(thread2, NULL);


  return 0;
  }
```

Output:

```
alaric@alaric-virtual-machine:~/Desktop$ gcc ques2.c -lpthread
alaric@alaric-virtual-machine:~/Desktop$ ./a.out
Enter the number of terms: 5
Enter the number that you want to square: Fibonacci Series: 0, 1, 1, 2, 3, 4
Square of 4 is 16alaric@alaric-virtual-machine:~/Desktop$
```

3) Create two threads thread1 and thread2 and call functions fun1 and fun2 respectively. Compute and print factorial in fun1 and prime number in fun2.

Code:

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

void *fun1(){
 int i,fact=1,number;
 printf("Enter a number: ");
  scanf("%d",&number);
    for(i=1;i<=number;i++){
      fact=fact*i;
  }
  printf("Factorial of %d is: %d",number,fact);
  printf("\n");

  }

void *fun2(){
int n,i,m=0,flag=0;
printf("Enter the number to check prime:");
scanf("%d",&n);
m=n/2;
for(i=2;i<=m;i++)
{
if(n%i==0)
{
printf("Number is not prime");
flag=1;
break;
}
}
if(flag==0)
printf("Number is prime\n");
printf("%d\n", n);

}

  int main(){
  pthread_t thread1, thread2;
   int iret1, iret2;


  iret1= pthread_create(&thread1, NULL, fun1, NULL);
  iret2= pthread_create(&thread2, NULL, fun2, NULL);

  pthread_join(thread1, NULL);
  pthread_join(thread2, NULL);

  return 0;
  }

```

Output:

```
alaric@alaric-virtual-machine:~/Desktop$ gcc ques3.c -lpthread
alaric@alaric-virtual-machine:~/Desktop$ ./a.out
Enter a number: 9
Enter the number to check prime:Factorial of 9 is: 362880
7
Number is prime
7
```

4) Create two threads thread1 and thread2 and call functions fun1 and fun2 respectively. Compute and print Armstrong number or not in fun1 and reverse number in fun2.

Code:

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

void *fun1(){
int n,r,sum=0,temp;
printf("Enter the number=");
scanf("%d",&n);
temp=n;
while(n>0)
{
r=n%10;
sum=sum+(r*r*r);
n=n/10;
}
if(temp==sum)
printf("Armstrong  number\n");
else
printf("Not an armstrong number\n");

}

 void *fun2(){
 int n, reverse = 0, remainder;
  printf("\n");
  printf("Enter an integer: ");
  scanf("%d", &n);

  while (n != 0) {
    remainder = n % 10;
    reverse = reverse * 10 + remainder;
    n /= 10;
  }

  printf("Reversed number = %d", reverse);


}

```

```
40  int main(){
41  pthread_t thread1, thread2;
42   int iret1, iret2;
43
44  iret1= pthread_create(&thread1, NULL, fun1, NULL);
45  iret2= pthread_create(&thread2, NULL, fun2, NULL);
46
47  pthread_join(thread1, NULL);
48  pthread_join(thread2, NULL);
49
50  return 0;
51 }
```

Output:

```
alaric@alaric-virtual-machine:~/Desktop$ gcc ques4.c -lpthread
alaric@alaric-virtual-machine:~/Desktop$ ./a.out
Enter the number=
153
Enter an integer: Armstrong  number
97
Reversed number = 79alaric@alaric-virtual-machine:~/Desktop$
```