LAB 4 ASSIGNMENT

Name: Birva Babaria

Roll no.: CE010

ID: 19CEUON064

Aim: Thread creation and Termination. Synchronization using mutex lock and unlock. (Use of pthread_create, ptread_join library functions of Pthread library).

Description:

1) ‘pthread_create’

Library: #include<pthread.h>

Syntax: int pthread_create(pthread_t *thread, const pthread_attr_t *attr, void *(*start_routine) (void *), void *arg);

Description: pthread_create() function creates a new thread in the current calling process. To strart the execution of the new thread start_routine() function is invoked. The arg argument passes the argument to the thread function. If arg is NULL then the thread is created with the default attributes.

On success, pthread_create function returns 0. On error, it returns an error number and the contents of the thread are undefined.

Example: int status = pthread_create(&thread, NULL, hello, NULL);

2) ‘pthread_join’

Library: #include<pthread.h>

Syntax: int pthread_join(pthread_t thread, void **retval);

Description: pthread_join() function waits for the thread specified by the thread to terminate. When the thread terminates, the pthread_join() function returns immediately.

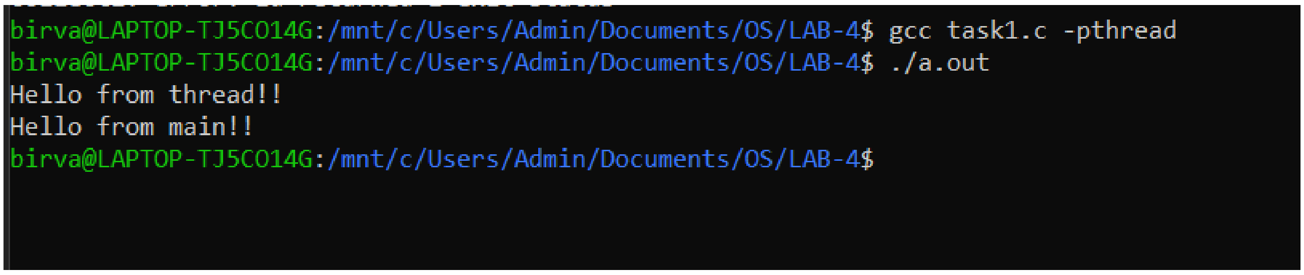
Example: pthread_join(thread,NULL);

1). Write a program to create a thread using pthread_create.

CODE:

```
//display hello from thread function

#include<stdio.h>

#include<pthread.h>

void *hello()

{

    printf("Hello from thread!!\n");

}

void main()

{

    pthread_t thread;

    int status = pthread_create(&thread, NULL, hello, NULL);

    pthread_join(thread,NULL);

    printf("Hello from main!!\n");

}
```

OUTPUT:



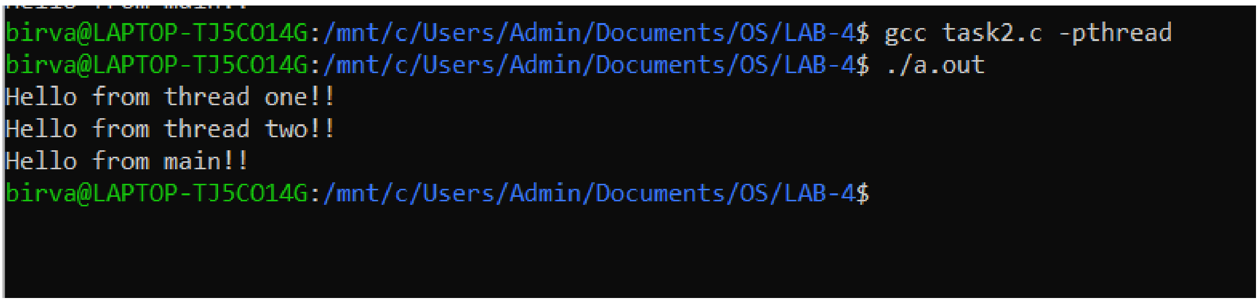
```
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ gcc task1.c -pthread
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ ./a.out
Hello from thread!!
Hello from main!!
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$
```

2). Write a program to call two different thread functions from different threads.

CODE:

```
//create two threads and each thread should invoke different
function

#include<stdio.h>

#include<pthread.h>

void *hello1()

{

    printf("Hello from thread one!!\n");

}

void *hello2()

{

    printf("Hello from thread two!!\n");

}

void main()

{

    pthread_t thread;

    int status1 = pthread_create(&thread, NULL, hello1, NULL);

    int status2 = pthread_create(&thread, NULL, hello2, NULL);

    pthread_join(thread,NULL);

    printf("Hello from main!!\n");

}
```

OUTPUT:



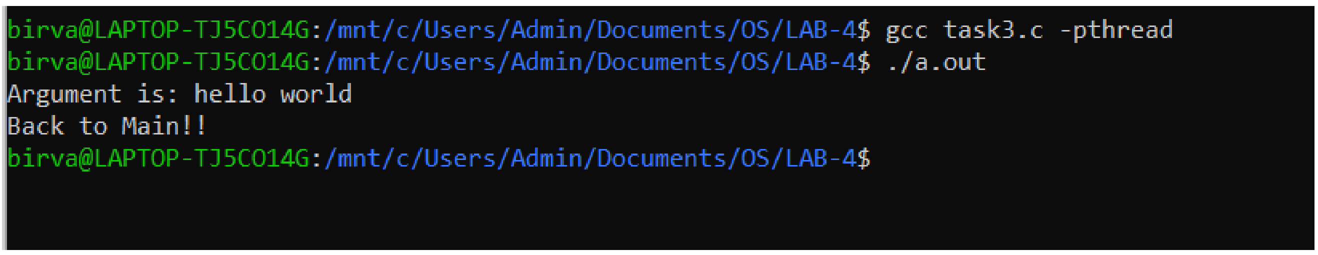
```
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ gcc task2.c -pthread
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ ./a.out
Hello from thread one!!
Hello from thread two!!
Hello from main!!
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$
```

3). Write a program to pass a character string to the threaded function.

CODE:

```
//pass a character string to the threaded function
#include<stdio.h>
#include<pthread.h>
void *func(void *arg)
{
    char *str = (char *)arg;
    printf("Argument is: %s\n",str);
}
void main()
{
    pthread_t thread;
    int status = pthread_create(&thread, NULL, func, (void *)"hello
world");
    pthread_join(thread,NULL);
    printf("Back to Main!!\n");
}
```

OUTPUT:



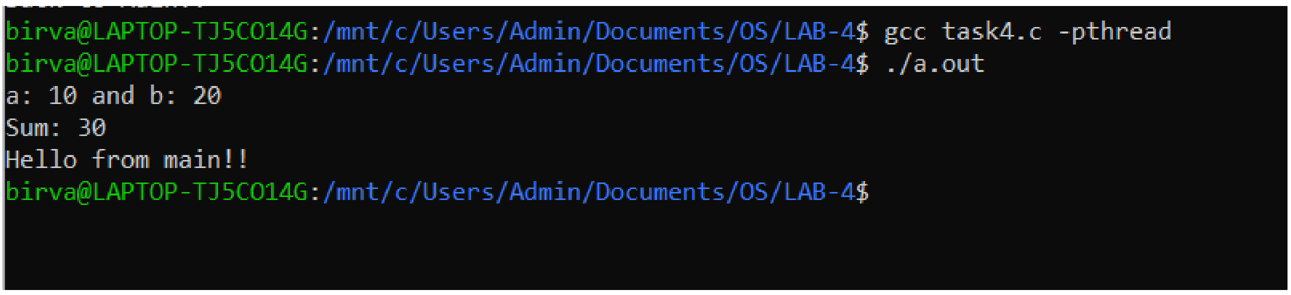
```
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ gcc task3.c -pthread
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ ./a.out
Argument is: hello world
Back to Main!!
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$
```

4). Write a program pass multiple arguments to the threaded function using struct.

CODE:

```
//pass multiple arguments to the threaded function using struct
#include<stdio.h>
#include<pthread.h>
struct mystruct{
	int a;
	int b;
};
void *add(void *arg)
{
	struct mystruct *strct = (struct mystruct *)arg;
	int sum = strct->a + strct->b;
	printf("a: %d and b: %d\n",strct->a,strct->b);
	printf("Sum: %d\n",sum);
}
void main()
{
	pthread_t thread;
	struct mystruct *str;
	str->a = 10;
	str->b = 20;
	int status = pthread_create(&thread, NULL, add, (void *)str);
	pthread_join(thread,NULL);
	printf("Hello from main!!\n");
}
```

OUTPUT:



```
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ gcc task4.c -pthread
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ ./a.out
a: 10 and b: 20
Sum: 30
Hello from main!!
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$
```

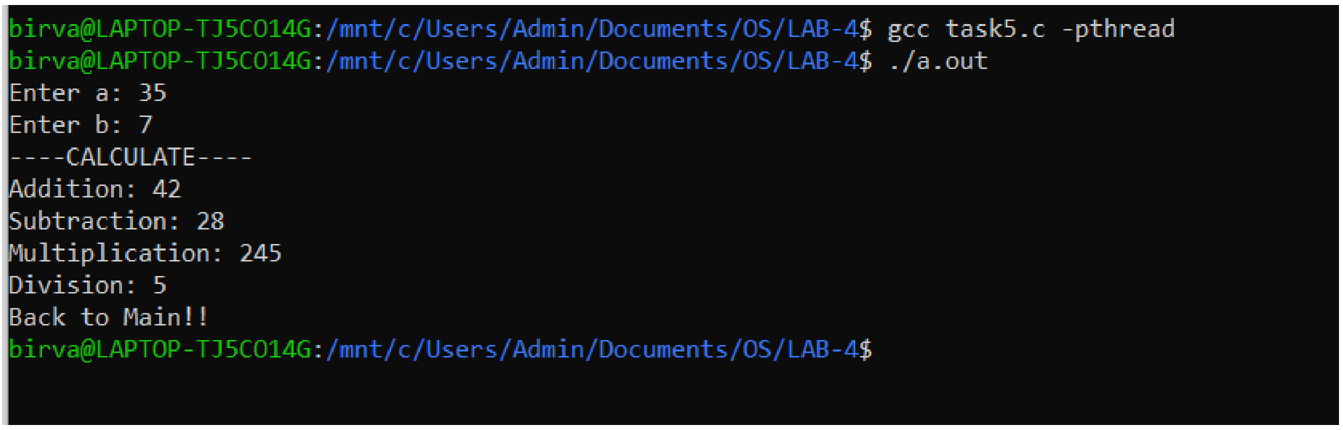
5). Write a program to implement simple calculator using threads.

CODE:

```
//construct simple calculator
#include<stdio.h>
#include<pthread.h>
struct mystruct{
    int a;
    int b;
};
void *add(void *arg)
{
    struct mystruct *strct = (struct mystruct *)arg;
    int sum = strct->a + strct->b;
    printf("Addition: %d\n",sum);
}
void *sub(void *arg)
{
    struct mystruct *strct = (struct mystruct *)arg;
    int sub = strct->a - strct->b;
    printf("Subtraction: %d\n",sub);
}
void *mul(void *arg)
```

```
{
     struct mystruct *strct = (struct mystruct *)arg;
     int mul = strct->a * strct->b;
     printf("Multiplication: %d\n",mul);
}
void *div(void *arg)
{
     struct mystruct *strct = (struct mystruct *)arg;
     int div = strct->a / strct->b;
     printf("Division: %d\n",div);
}
void main()
{
     pthread_t thread;
     struct mystruct *str;
     int a,b;
     printf("Enter a: ");
     scanf("%d",&a);
     printf("Enter b: ");
     scanf("%d",&b);
     str->a = a;
     str->b = b;
     printf("----CALCULATE----\n");
     int status1 = pthread_create(&thread, NULL, add, (void *)str);
     int status2 = pthread_create(&thread, NULL, sub, (void *)str);
     int status3 = pthread_create(&thread, NULL, mul, (void *)str);
     int status4 = pthread_create(&thread, NULL, div, (void *)str);
     pthread_join(thread,NULL);
     printf("Back to Main!!\n");
}
```

OUTPUT:



```
birva@LAPTOP-TJ5C014G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ gcc task5.c -pthread
birva@LAPTOP-TJ5C014G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ ./a.out
Enter a: 35
Enter b: 7
----CALCULATE----
Addition: 42
Subtraction: 28
Multiplication: 245
Division: 5
Back to Main!!
birva@LAPTOP-TJ5C014G:/mnt/c/Users/Admin/Documents/OS/LAB-4$
```

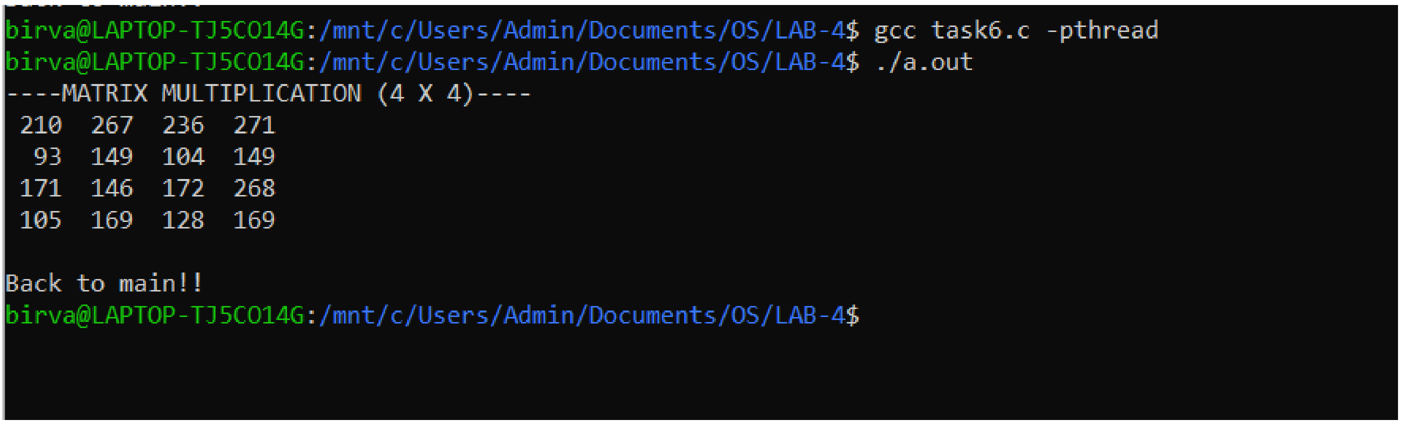
6). Write a program to multiply two matrices.

CODE:

```
//multiply two matrices (4 X 4)
#include<stdio.h>
#include<pthread.h>
int a[4][4] = {{5,7,9,10},
                 {2,3,3,8},
                 {8,10,2,3},
                 {3,3,4,8}
                };
int b[4][4] = {{3,10,12,18},
                 {12,1,4,9},
                 {9,10,12,2},
                 {3,12,4,10}
                };
void *mul(void *arg)
{
     int i = *(int *)arg;
     int arr[4] = {0};
     for(int j=0;j<4;j++)
```

```
    {
        for(int k=0;k<4;k++)
        {
            arr[j] += (a[i][k] * b[k][j]);
        }
    }
    for(int j=0;j<4;j++)
    {
        printf(" %3d ",arr[j]);
    }
    printf("\n");
}
void main()
{
    pthread_t thread;
    int a=0,b=1,c=2,d=3;
    printf("----MATRIX MULTIPLICATION (4 X 4)----\n");
    int status1 = pthread_create(&thread, NULL, mul, (void *)&a);
    pthread_join(thread,NULL);
    int status2 = pthread_create(&thread, NULL, mul, (void *)&b);
    pthread_join(thread,NULL);
    int status3 = pthread_create(&thread, NULL, mul, (void *)&c);
    pthread_join(thread,NULL);
    int status4 = pthread_create(&thread, NULL, mul, (void *)&d);
    pthread_join(thread,NULL);
    printf("\nBack to main!!\n");
}
```

OUTPUT:



```
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ gcc task6.c -pthread
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$ ./a.out
----MATRIX MULTIPLICATION (4 X 4)----
 210  267  236  271
  93  149  104  149
 171  146  172  268
 105  169  128  169

Back to main!!
birva@LAPTOP-TJ5CO14G:/mnt/c/Users/Admin/Documents/OS/LAB-4$
```