KHUSHBU PATEL (150034)

LINUX : MULTI-THREADING

QUE : 1) Write a pthread application where main task terminated but pending pthreads task still execute.

```
#include<pthread.h>
#include<stdio.h>
void thread_fun()
{
printf("Thread function is calling and executing...\n");
sleep(2);
printf("The thread is ends here...\n");
}
int main()
{
pthread_t t1;
printf("Thread is creating now... \n");
pthread_create(&t1,NULL,thread_fun,NULL);
sleep(2);
printf("Main Task ends here...\n");
pthread_exit(NULL);
}
```

```
khushbu@khushbu-VirtualBox:~/backup/Assign_4_multithreading$ ./a1
Thread is creating now...
Thread function is calling and executing...
Main Task ends here...
The thread is ends here...
```

QUE : 2) Write a program where a structure of information passed to pthread task function, and display structure of information.

```
#include <stdio.h>
#include <pthread.h>
struct printf_info
{
char information;
};
void *print_char_num(void *parameters)
{
struct printf_info *s = (struct printf_info *)parameters;
printf("Data prints here = = %c \n", s->information);
return NULL;
}
void main()
{
pthread_t t1;
struct printf_info arg1;
arg1.information = 'A';
pthread_create(&t1, NULL, print_char_num, &arg1);
pthread_join(t1, NULL);
}
```

```
khushbu@khushbu-VirtualBox:~/backup/Assign_4_multithreading$ gcc assign_4_que2.c -o a2 -lpthread
khushbu@khushbu-VirtualBox:~/backup/Assign_4_multithreading$ ./a2
Data prints here = = A
```

QUE : 3) Write a pthread program that implements simple initialization code.

```
#include<pthread.h>
#include<stdio.h>
pthread_once_t once = PTHREAD_ONCE_INIT;
void *initfunction()
{
printf("\nThis is initialization function\n");
}
void *threadfunction(void *i)
{
printf("\nI am into thread = %d\n",(int *)i);
pthread_once(&once,(void *)initfunction);
printf("\n exit from mythread = %d\n",(int *)i);
}
int main()
{
        pthread_t thread;
        pthread_create(&thread,NULL,threadfunction,(void *)1);
        pthread_exit(NULL);
}
```

```
khushbu@khushbu-VirtualBox:~/backup/Assign_4_multithreading$ ./a3
I am into thread = 1
This is initialization function
 exit from mythread = 1
```

QUE : 4) 4.write a program, which get and set pthread scheduling policy and priority.

```
#include<stdio.h>

#include<pthread.h>

int main()

{

struct sched_param scheduling_param;

int priority,policy;

int i;

i=pthread_getschedparam(pthread_self(),&policy,&scheduling_param);

printf("\n INITIAL POLICY AND PRIORITY OF THREAD :- \t Policy:%d \t Priority
:%d\n",policy,scheduling_param.sched_priority);

policy=SCHED_FIFO;

scheduling_param.sched_priority=5;

i=pthread_setschedparam(pthread_self(),policy,&scheduling_param);


i=pthread_getschedparam(pthread_self(),&policy,&scheduling_param);


printf("\n AFTER SET THE PRIORITY          :-  \t Policy:%d \t Priority
:%d\n",policy,scheduling_param.sched_priority);

}
```

```
khushbu@khushbu-VirtualBox:~/backup/Assign_4_multithreading$ sudo su
root@khushbu-VirtualBox:/home/khushbu/backup/Assign_4_multithreading# gcc assign_4_que4.c -o a4 -lpthr
ead
root@khushbu-VirtualBox:/home/khushbu/backup/Assign_4_multithreading# ./a4

 INITIAL POLICY AND PRIORITY OF THREAD :-        Policy:0        Priority :0

 AFTER SET THE PRIORITY                :-        Policy:1        Priority :5
root@khushbu-VirtualBox:/home/khushbu/backup/Assign_4_multithreading#
```

QUE : 5) 5.Write a program that implements threads synchronization using pthread spinlock techniques.

```
#include<stdio.h>
#include<stdlib.h>
#include<errno.h>
#include<unistd.h>
#include<pthread.h>
#include<sys/types.h>
#include<bits/types.h>
static pthread_spinlock_t spinlock;
volatile int slock;
void *spinlockThread(void *i)
{
        int a;
printf("enter thread %d getting spin lock \n",(int *)i);
a = pthread_spin_lock(&slock);
printf("%d Thread unlock the spin lock\n",(int *)i);
a = pthread_spin_unlock(&slock);
printf("%d thread complete\n",(int *)i);
return NULL;
}
int main()
{
        int a=0;
        int b=0;
        pthread_t  thread1,thread2;
        if(pthread_spin_init(&slock,PTHREAD_PROCESS_PRIVATE)!=0)
```

```
        perror("init");
        a = pthread_spin_lock(&slock);
        printf("spin lock created for thread 1..\n");
        a = pthread_create(&thread1 , NULL,spinlockThread , (int *)1);
        printf("thread 1 , now unlock the spin lock\n");
        a = pthread_spin_unlock(&slock);
        if(a==0)
{
        printf("\n  thread 1 successfully unlocked..\n");

}
else
{
        printf("\n thread1 unsuccessfully unlocked..\n");
}

printf("main, wait for the thread to end\n");
        b = pthread_spin_lock(&slock);
        printf("spin lock created for thread 2..\n");
        b = pthread_create(&thread2 , NULL,spinlockThread , (int *)2);
        printf("thread 2 , now unlock the spin lock\n");
        b = pthread_spin_unlock(&slock);
        if(b==0)
{
        printf("\n  thread2 successfully unlocked..\n");
}
else
```

```
{
        printf("\n thread2 unsuccessfully unlocked..\n");
}
a = pthread_join(thread1,NULL);
b = pthread_join(thread2,NULL);
return 0;
}
```

```
khushbu@khushbu-VirtualBox:~/backup/Assign_4_multithreading$ ./a5
spin lock created for thread 1..
thread 1 , now unlock the spin lock

  thread 1 successfully unlocked..
main, wait for the thread to end
spin lock created for thread 2..
thread 2 , now unlock the spin lock

  thread2 successfully unlocked..
enter thread 1 getting spin lock
1 Thread unlock the spin lock
1 thread complete
enter thread 2 getting spin lock
2 Thread unlock the spin lock
2 thread complete
```