THREAD PROGRAMS

```
//Simple multitreading program; to compile - gcc th1.c -pthread
//Execute the program with and without pthread_join()
#include<stdio.h>
#include<unistd.h>
#include<pthread.h>
void * threadfun(); // the Thread routine with no parameters
int i, j, n;
void main()
{
pthread_t T1; //thread id
pthread_create(&T1, NULL, threadfun, NULL);//create thread
//pthread_create(&tid, &attr, threadfun, args);
//pthread_join(T1,NULL);//the main thread waits for child thread to complete
printf("Inside main thread\n");
for(i=10; i<=15;i++)
{
printf("i=%d\n",i);
sleep(1);
}
}
void * threadfun()
{
printf("inside thread\n");
for(j=1; j<=5;j++)
{
printf("j=%d\n",j);
sleep(1);
}
}
```

```
//program with two threads and main thread; also passing parameters to child
threads
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<pthread.h>
void *sum(void *arg);
void *mul(void *arg);
int sm=0, prod=1;
void main(int argc, char * argv[])
{
pthread_t T1,T2;
pthread_attr_t attr;//set of thread attributes
pthread_attr_init(&attr);//get the default attributes
pthread_create(&T1, &attr, sum, argv[1]);
```

```
pthread_create(&T2, &attr, mul, argv[1]);
pthread_join(T1,NULL);
pthread_join(T2,NULL);
printf("Inside main thread\n");
printf("sum=%d\n",sm);
printf("product=%d\n",prod);
}
void *sum(void *parm)
{
int i, n;
n = atoi(parm);//string to integer i.e., ASCII to int
printf("inside sum thread\n");
for(i=1; i<=n;i++)
{
sm+=i;
}
printf("sum thread completed\n");
}
void *mul(void *parm)
{
int i, n;
n = atoi(parm);
printf("inside mul thread\n");
for(i=2; i<=n;i++)
{
prod =prod *i;
}
printf("mul thread completed product\n");
}

//the main thread catching the returned status of child thread
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<pthread.h>
void *sum(void *arg);
int a[5]={1,2,3,4,5};
int sm=0;
void *res;
void main()
{
pthread_t T1;
pthread_attr_t attr;
pthread_attr_init(&attr);
pthread_create(&T1, &attr, sum, (void *)a);
pthread_join(T1,&res);//&res is the pointer to the location where the exit status of
the thread mentioned in T1 is stored
```

```
printf("Inside main thread\n");
printf("sum=%d\n",sm);
printf("thread returned: %s\n",(char *)res);
}
void *sum(void *parm)
{
int i;
int *x=parm;
printf("inside thread\n");
for(i=0;i<5;i++)
sm+=x[i];
pthread_exit("sum calculated");

}
```