CS266 Practice Lab 4

NAME:

ARCHIT AGRAWAL

ROLL NO. :

202051213

SECTION:

2

**Code 1**

//thread\_example1.c

#include<stdio.h>

#include<pthread.h>

#include<unistd.h>

#include<sys/time.h>

//#include<sys/types.h>

void \*kidfunc(void \*p)

{

    printf("Thread ID is ---> %u\n",pthread\_self());

    printf("Kid ID is ---> %d\n",getpid( ));

}

void main( )

{

    pthread\_t kid;

    pthread\_create(&kid, NULL, kidfunc, NULL);

    printf("Parent ID is ---> %d\n", getpid( ));

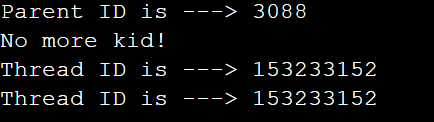
    //pthread\_join(kid, NULL);

    printf("No more kid!\n");

    //return 0;

}

**Output**





**Code 2**

#include <stdio.h>

#include <pthread.h>

int glob\_data = 5 ;

void \*kidfunc(void \*p)

{

    printf ("Kid here. Global data was %d.\n", glob\_data);

    glob\_data = 15;

    printf("Kid Again. Global data was now %d.\n", glob\_data);

}

int main( )

{

    pthread\_t kid;

    pthread\_create (&kid, NULL, kidfunc, NULL);

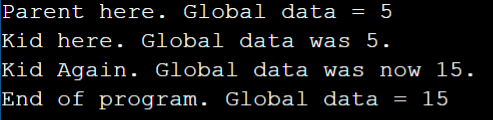
    printf("Parent here. Global data = %d\n",glob\_data);

    pthread\_join(kid,NULL) ;

    printf("End of program. Global data = %d\n", glob\_data);

}

**Output**



**Code 3**

#include<pthread.h>

#include<stdio.h>

//#include<sys/types.h>

//#include<unistd.h>

#include<stdlib.h>

int sum; /\*This data is shared by the thread(s) \*/

void \*runner(void \*param); /\* the thread \*/

void main(int argc, char \*argv[]){

    pthread\_t tid; /\* the thread identifier \*/

    pthread\_attr\_t attr; /\* set of thread attributes \*/

    if(argc != 2)

    {

        fprintf(stderr,"usage: a.out <integer value>\n");

        exit(0);

    }

    if(atoi(argv[1]) < 0)

    {

        fprintf(stderr, "%d must be >= 0 \n", atoi(argv[1]));

        exit(0);

    }

/\* get the default attributes \*/

pthread\_attr\_init(&attr);

/\*create the thread \*/

pthread\_create(&tid,&attr,runner,argv[1]);

/\* Now wait for the thread to exit \*/

pthread\_join(tid,NULL);

printf("sum = %d\n",sum);

}

/\*The thread will begin control in this function \*/

void \*runner(void \*param)

{

    int upper = atoi(param);

    int i;

    sum=0;

    if(upper > 0)

    {

        for(i=1; i <= upper;i++)

        sum += i;

    }

    pthread\_exit(0);

}

**Output**



**Code 4**

#include<pthread.h>

#include<stdio.h>

#include<stdlib.h>

#include<sys/types.h>

#include<unistd.h>

#define NUM\_THREADS 5

void \*PrintHello(void \*threadid)

{

    printf("\n%d: Hello World!\n", threadid);

    //pthread\_exit(NULL);

}

int main( )

{

    pthread\_t threads [NUM\_THREADS];

    int rc, t;

    for(t=0; t < NUM\_THREADS; t++)

    {

        printf ("Creating thread %d\n", t);

        rc = pthread\_create (&threads[t], NULL, PrintHello, (void \*) t );

        if (rc)

        {

            printf("ERROR; return code from pthread\_create() is %d\n",rc);

            exit(-1);

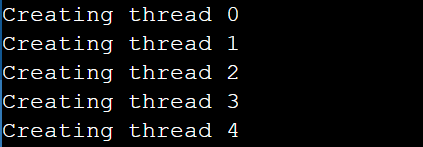
    }

}

//pthread\_exit(NULL);

}

**Output**



**Code 5**

//thread\_example5.c

#include <stdio.h>

#include <pthread.h>

#include <unistd.h>

#include<stdlib.h>

int this\_is\_global;

void thread\_func( void \*ptr );

int main( )

{

    int local\_main; int pid, status;

    pthread\_t thread1, thread2;

    printf("First, we create two threads to see better what context they share...\n");

    this\_is\_global=1000;

    printf("Set this\_is\_global=%d\n",this\_is\_global);

    pthread\_create( &thread1, NULL, (void\*)&thread\_func, (void\*) NULL);

    pthread\_create(&thread2, NULL, (void\*)&thread\_func, (void\*) NULL);

    pthread\_join(thread1, NULL); pthread\_join(thread2, NULL);

    printf("After threads, this\_is\_global=%d\n",this\_is\_global);

    printf("\n");

    printf("Now that the threads are done, let's call fork..\n");

    local\_main=17; this\_is\_global=17;

    printf("Before fork(), local\_main=%d, this\_is\_global=%d\n",local\_main,

    this\_is\_global);

    pid=fork();

    if (pid == 0)

    {

        /\* this is the child \*/

        printf("In child, pid %d: &global: %X, &local: %X\n", getpid(), &this\_is\_global, &local\_main);

        local\_main=13;

        this\_is\_global=23;

        printf("Child set local main=%d, this\_is\_global=%d\n",local\_main,

this\_is\_global);

        exit(0);

    }

    else

    {

        /\* this is parent \*/

        printf("In parent, pid %d: &global: %X, &local: %X\n", getpid(),

&this\_is\_global, &local\_main);

        wait(&status);

        printf("In parent, local\_main=%d, this\_is\_global=%d\n",local\_main,

this\_is\_global);

    }

    exit(0);

}

void thread\_func(void \*dummy)

{

    int local\_thread;

    printf("Thread %d, pid %d, addresses: &global: %X, &local: %X\n",

    pthread\_self(),getpid(),&this\_is\_global, &local\_thread);

    this\_is\_global++;

    printf("In Thread %d, incremented this\_is\_global=%d\n", pthread\_self(),this\_is\_global);

    pthread\_exit(0);

}

**Output**

