OS HOME ASSIGNMENT

VAISHALI UPADHYAY

121059

1.

Parent sees i = 0

Parent sees i = 1

Child sees i = 0

Child sees i = 1

Parent sees i = 2

Child sees i = 2

2.

#include<pthread.h>

int a = 0;

void \*print\_fn(void \*ptr)

{

int tid = \*(int \*)ptr;

int b = 0;

a++; b++;

printf("id: %d a: %d b: %d\n", tid, a, b);

// while (1); // Spin-wait here forever

}

int main()

{

pthread\_t t1, t2;

int tid1 = 1;

int tid2 = 2;

int ret1, ret2;

a++;

printf("Parent says a: %d\n", a);

ret1 = pthread\_create(&t1, NULL, print\_fn, (void \*)&tid1);

ret2 = pthread\_create(&t2, NULL, print\_fn, (void \*)&tid2);

if (ret1 || ret2) {

//fprintf(stderr, "ERROR: pthread\_create failed\n");

exit(1);

}

if (pthread\_join(t1, NULL)) {

perror("join of t1");

exit(1);

}

if (pthread\_join(t2, NULL)) {

perror("join of t2");

exit(1);

}

printf("Thread 1 and 2 complete\n");

}

Thread headers were not provided.

while(1) cause infinite loop.

ANSWER

Parent says a: 1

id: 1 a: 2 b: 1

id: 2 a: 3 b: 1

Thread 1 and 2 complete

3.

Each process has its own address space, file descriptors, and thread of execution, at kernel level. The processes contain codes to create, destroy, schedule and synchronise user level threads to support multiple threads at user level whcich can be thought of as mulitplexing many user-level threads onto the single kernel thread, all managedÂ within the process. The scheduler can run any arbitrary scheduling algorithms, and is independent of the kernel's scheduler.