**National University of Computer & Emerging Sciences, Karachi  
Fall-2018 CS-Department**Fast

**Lab Final**

|  |  |  |
| --- | --- | --- |
| **Course Code: CL205** | **Course Name: Operating Systems Lab** | |
| **Instructor Name: Sumaiyah Zahid** | | |
| **Student Roll No:** | | **Section:** |

***"If there is something, you don’t know today. You will surely learn afterwards. Life is not an exam hall."  
BEST OF LUCK!***

Instructions

* Rules are made to break them. So, invent yours and I’ll break.

**Time**: 90 minutes **Max Marks:** 40 points

**This program will create \_\_\_\_ child processes and \_\_\_\_ threads?**  **(5 marks)**

int main() **Output**

{

printf(“OS\n”);

fork();

pthread\_create(&tid, NULL, thread, NULL);

fork();

printf(“OS\n”);

pthread\_create(&tid, NULL, thread, NULL);

fork();

pthread\_create(&tid, NULL, thread, NULL);

return 0;

}

**Output**

int main() {

printf("%d\n", getpid());

a=fork();

printf("%d\n", getpid());

if (a==0){

printf("%d\n", getpid());

fork();

printf("%d\n", getpid());

}

printf(" Done!\n");

return 0;

}

**Write appropriate system calls in the blanks**  **(5 marks)**

int main(void) {

int shmid;

key\_t key;

char \*shm, \*s;

key = 2211;

fflush(stdin);

if((shmid = \_\_\_\_\_\_\_\_\_\_\_\_\_(key, MAXSIZE, IPC\_CREAT | 0666)) < 0)

die("error");

if((shm = \_\_\_\_\_\_\_\_\_\_\_\_\_ ( \_\_\_\_\_\_\_\_\_, NULL, 0)) == (char\*) -1)

die("error");

for(s = shm; \*s != '\0'; s++)

putchar(\*s);

\*shm = '\*';

printf("\n");

exit(0);

}

**Advantage of FIFO over pipe is**

1. related processes can communicate
2. unrelated processes can communicate
3. all of the mentioned
4. none of the mentioned

**Which is Fastest IPC?**

1. Message Queue
2. shared memory
3. Socket
4. All of the mentioned

**What are the two basic function for any module?** **(5 marks)**

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Command for compiling module \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Command for module details \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**What is the output on the terminal after compiling?**

printk(KERN\_INFO "Hello World. \n");

printk(KERN\_INFO "Final Paper of OS”);

printk("GoodBye");

return 0;

In which pattern pthread\_create and pthread\_join can create a serial execution of threads and parallel. Illustrate by writing code for 3 threads. **(2 marks)**

True or false: Code in an OpenMP program that is not covered by a pragma is executed by all threads. **(1 marks)**

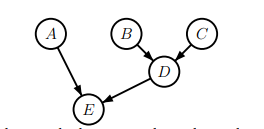
Procom has 4 volunteers on their front desk.

* Volunteer 1 manages On day registration
* Volunteer 2 handles announcements
* Volunteer 3 handles sponsors
* Volunteer 4 resolve queries of participants

**Implement this system using OpenMP for total 100 participants. Asuming 25 participants for each volunteer. (5 marks)**

Write a sketch of a C program that uses Pthreads to execute the five functions in a way that is maximally parallel, but adheres to the above dependency graph.

The edge from node B to node D means that functionB must be called, and must return, before functionD can be called. **(2 marks)**



**Write all possible output on executing the code below? (3 marks)**

sem\_t mutex; **Output**

void\* thread(void\* arg)

{

Int a= \* ((int\*)arg);

printf("\nEntering..\n");

sem\_wait(&mutex);

printf("\n %d Entered..\n",a);

sleep(4);

printf("\nJust Exiting...\n");

sem\_post(&mutex);

}

int main()

{

sem\_init(&mutex, 0, 1);

pthread\_t t1,t2;

pthread\_create(&t1,NULL,thread,&0);

pthread\_create(&t2,NULL,thread,&1);

pthread\_join(t1,NULL);

pthread\_join(t2,NULL);

sem\_destroy(&mutex);

return 0;

}

The classic problems of producers (such as CPUs) and consumers (such as a printers) concerns one or more process data that one or more process consumes later through a single buffer. Systems must make sure that the producer won’t try to add data to full a buffer, and the consumer won’t try to make withdrawals from an empty buffer. And for the integrity of data only one process must be allowed to access the buffer at a time. Assume buffer contain 5 files maximum, design the procedures and consumers’ processes using semaphores. **(5 marks)**

Write a code snippet which sets default behavior of ctrl+\, ignores ctrl+Z, assign funcA to ctrl+C.and func B to floating point error. **(5 marks)**

**Write output on executing the code below? (2 marks)**

int main(void) **Output**

{

int child\_pid, i;

child\_pid = fork();

if (child\_pid == 0)

{

for (i = 0; i < 20000000; i++)

{

}

cout << "Bye from Child!" << endl;

}

else

{

sleep(1);

kill(child\_pid, SIGINT);

cout << "Bye from Parent " << endl;

}

}