L-3/T-2/CSE **Date : 02/10/2023**

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY, DHAKA

L-3/T-2 B. Sc. Engineering Examinations 2021-2022

Sub : **CSE 313** (Operating System)

Full Marks : 210 Time : 3 Hours

The figures in the margin indicate full marks.

USE SEPARATE SCRIPTS FOR EACH SECTION

SECTION – A

There are **FOUR** questions in this section. Answer any **THREE**.

1. (a) Consider the following workload: **(18)**

Process	Priority (Lowest Number has Highest Priority)	Duration (sec)	Arrival Time (sec)
P1	1	60	0
P2	1	25	30
P3	2	80	50
P4	3	20	70

Draw the Gantt chart and calculate the average turnaround time for each of the following scheduling algorithms:

- Shortest Remaining Time Next
- Priority Scheduling. Within the same priority class, schedule according to Round Robin with quantum 20 sec. Each priority class maintains separate FIFO queue.

(b) Four jobs have arrived at the same time in a batch system. Their expected run times are 9, 3, 5, and X. In what order should they be run to obtain optimal average turnaround time? (Hint: The jobs are non-preemptive) **(10)**

(c) Differentiate between an unsafe state and a deadlock state. **(7)**

2. (a) Peterson's solution for achieving mutual exclusion for using critical regions is presented in Figure for Question 2(a): **(12)**

(i) Discuss priority inversion problem with a high-priority process, H, and a low-priority process, L.

(ii) Does the same problem occur if round-robin scheduling is used instead of priority scheduling? Justify.

```
#define FALSE  0
#define TRUE   1
#define N      2                    /* number of processes */

int turn;                           /* whose turn is it? */
int interested[N];                  /* all values initially 0 (FALSE) */

void enter_region(int process);     /* process is 0 or 1 */
{
     int other;                     /* number of the other process */

     other = 1 - process;           /* the opposite of process */
     interested[process] = TRUE;    /* show that you are interested */
     turn = process;                /* set flag */
     while (turn == process && interested[other] == TRUE) /* null statement */ ;
}

void leave_region(int process)      /* process: who is leaving */
{
     interested[process] = FALSE;   /* indicate departure from critical region */
}
```

Figure for Question 2(a): Peterson's solution for 2 processes

Contd P/2

CSE 313

Contd. Q. No. 2(a)(ii)

(b) In the dining philosophers problem, let the following protocol be used: An even-numbered philosopher always picks up his left fork before picking up his right fork; an odd-numbered philosopher always picks up his right fork before picking up his left fork. Investigate whether this modified protocol prevents deadlock. **(10)**

(c) State the four requirements that must be present in a solution to support mutual exclusion among processes sharing resources. **(6)**

(d) Differentiate between microkernel and monolithic kernel architectures. **(7)**

3. (a) Consider a system with 4 processes: P1 through P4 and 3 resources types: A (9 units), B (3 units), C (6 Units). Current allocation of resources and maximum requirement of a process for each resource are given in Figure for Question 3(a). Determine whether the current state of this system is safe or not. **(15)**

	A	B	C
P1	1	0	0
P2	6	1	2
P3	2	1	1
P4	0	0	2

Current Allocation Matrix

	A	B	C
P1	3	2	2
P2	6	1	3
P3	3	1	4
P4	4	2	2

Maximum Requirement Matrix

Figure for Question 3(a)

(b) Suppose that there is a resource deadlock in a system. Give an example scenario to show that the set of processes deadlocked can include some processes that are not in the circular chain in the corresponding resource allocation graph. **(10)**

(c) Explain how to attack the following conditions to structurally prevent deadlock

(i) Hold and wait condition (ii) Circular wait condition. **(10)**

4. (a) Write down the steps in making a system call. **(7)**

(b) Differentiate between process context switching and thread context switching Identify which of the following are "per process item" and which are "per thread item"?

Program Counter, Stack, Address Space, Global Variables, Registers, Directories, Child Process, Local Variables **(13)**

Contd P/3

CSE 313

Contd ... Q. No. 4

(c) Illustrate the position of scheduler, process table and thread table in user-level thread implementation and kernel-level thread implementation with diagrams. With the help of your diagrams, justify that kernel-level thread implementation is better than user-level thread implementation with respect to blocking. **(15)**

SECTION – B

There are **FOUR** questions in this section. Answer any **THREE**.

5. (a) The following sequence of Virtual Page Numbers (VPN) has been referenced in your system. **(7+7+14=28)**

1, 2, 3, 4, 5, 2, 3, 1,2, 3, 4, 5, 1

Now, answer the following questions.

(i) Explain Belady's anomaly. Show that this anomaly occurs for cache size 3 and 4 in the above case.

(ii) Calculate hit and miss rate for the optimal page replacement algorithm.

(iii) If the TLB can keep at most 4 entries and employs LRU for TLB replacement and Page replacement uses CLOCK algorithm with cache size 4. Calculate the memory access time for the above memory access sequence. TLB access takes 5ns, memory access takes 60ns, and disk access takes 4ms.

Hint: The first 5 memory references in CLOCK algorithm gives the following cache states:

Access	Cache State (After)			
1	1(1)*	?(0)	?(0)	?(0)
2	1(1)	2(1)*	?(0)	?(0)
3	1(1)	2(1)	3(1)*	?(0)
4	1(1)	2(1)	3(1)	4(1)*
5	5(1)*	2(0)	3(0)	4(0)

Here,

"?" refers to unused cache,

the number in bracket refers to *use bit* and

"*" is the location of the *clock hand.*

(b) Both **ffs** and **lfs** try to optimize on **vsfs** in some ways. State and elaborate the difference in their optimization criteria. **(7)**

6. (a) The following function writes a sequence of n numbers to a given file descriptor **fd**. **(5×3+13=28)**

```
void write_fd(int fd, int n){
    char buf[10];
    for (int i = 0; i < n; i++) {
        itoa(i, buf, 10);
        write(fd, buf, strlen(buf));
    }
}
```

Contd P/4

CSE 313

Contd ... Q. No. 6(a)(i)

(i) For each of the following possible values of n, design a device driver where `write_fd` is called frequently with n. Explain your design decisions using illustrative pseudocodes.

(*a*) always 10

(*b*) always 1000000

(*c*) mixture of 10 and 1000000

(ii) What is the difference of output from the following two code blocks? Analyze with necessary figures and arguments showing what happens in the kernel.

Code block 1	**Code block 2**
`int fd1 = open("file.txt");`	`int fd1 = open("file.txt");`
`int fd2 = open("file.txt");`	`int fd2 = dup(fd1);`
`write_fd(fd1);`	`write_fd(fd1);`
`write_fd(fd2);`	`write_fd(fd2);`

(b) Memory allocation API consists mainly of the following two functions: **(7)**

```
void *malloc(size_t size);
void free(void *ptr);
```

With illustrative examples, show how the allocation API works. Mention the necessary bookkeeping structure(s) it requires.

7. (a) You have executed the following commands in the `root` of an **ffs** file system with 4 byte block number and 4 KB block size. The disk has an average disk-arm positioning time of 10 ms and max transfer rate of 100 MB/s. **(4×3+16=28)**

```
mkdir p
mv /q/foo.txt /p/bar.txt
```

Here, `foo.txt` is of size 1 GB.

Now answer the following questions.

(i) Given, bitmaps are in block 4, inodes are in block 5 and data for the root directory is in block 6. Also, the journal is kept in blocks 26 to 31. Determine the journaling timeline if the system uses:

(*a*) Data Journaling

(*b*) Metadata Journaling

(*c*) Metadata Journaling with Checksum Optimization

(ii) How will this file be laid out in **ffs**? How does **ffs** get the relevant information needed to decide that? Calculate how long it will take to sequentially read the whole file. *You can assume, there is enough space to save the file in ffs layout and only moving from one block group to another requires a disk-arm positioning.*

Contd P/5

CSE 313

Contd ... Q. No. 7

(b) What is dynamic relocation and segmentation in the context of memory management? Why would you choose one over another? Using pseudocodes, elaborate the address translation process used by these two systems. **(7)**

8. (a) You have set up a RAID system with 10 disks and 4KB block size. You are using **lfs** as the default file system where segment size is 64 MB. The disk config is as follows:**(3+9+6+5×2=28)**

Capacity	1TB
Rotation speed	10,000 RPM
Max seek time	12 ms
Max transfer rate	100 MB/s
Platters	2
Sector size	1 KB
Cache	16 MB

The following command has been executed in your file system.

```
rm lfsfile.txt
```

Now, answer the following questions.

(i) How is the inode number problem solved in **lfs**?

(ii) Write down the steps that will be executed while running the command. How will **lfs** invalidate any subsequent access to `lfsfile.txt`?

(iii) Elaborate the recovery action that **lfs** will take, if a crash occurs during the operation.

(iv) Calculate the throughput for read and writes in your file system (lfs) for the following RAID setups.

(*a*) RAID-0

(*b*) RAID-1

Hint:

- lfs reads 1 block at a time and writes 1 segment at a time.
- You can read/write one sector at a time on a disk.

(b) What is *thrashing* in the context of memory management? Why is it a problem? List some solutions for it. **(7)**

L-3/T-2/CSE **Date: 29/03/2023**

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY, DHAKA

L-3/T-2 B. Sc. Engineering Examinations 2020-2021

Sub: **CSE 313** (Operating System)

Full Marks: 210 Time: 3 Hours

USE SEPARATE SCRIPTS FOR EACH SECTION

The figures in the margin indicate full marks

SECTION – A

There are **FOUR** questions in this section. Answer any **THREE** questions.

1. (a) Draw the Processes State Transition Diagram. Include the following states: *Embryo, Running, Runnable, Zombie, Sleeping.* **(10)**

(b) Given a basic spinlock, Assume that locking the spinlock takes **A** time units (if no one is holding the lock); unlock also takes **A** time units. Assume further that a context switch takes **C** time units, and that a time slice is **T** time units long.

Assume this code sequence, executed by **two** threads on **one** processor at roughly the same time: **(5×3=15)**

```
mutex_lock() ;
do_something() ;   // takes no time to execute
mutex_unlock() ;
```

i. What is the best-case time for the two threads on one CPU to finish this code sequence?

ii. What is the worst-cast time for the two threads to finish this code sequence? Assume that only **three** context switches can occur at a maximum.

iii. If the spin lock is instead changed to a queue-based lock, how does that change the worst-case time?

(c) You are given a new atomic function, called **`FetchAndSubtract()`.** It executes as a single atomic instruction, and is defined as follows: **(10)**

```
int FetchAndSubtract  (int *location )  {
    int value = *location;        // read the value pointed to by location
    *location = value - 1;        // decrement it and store result back
    return value ;                // return old value
}
```

You are given the task: write the **`lock_init()`, `lock()`,** and **`unlock()`** functions (and also Define a **`lock_t`** structure) that use **`FetchAndSubtract()`** to implement a working lock.

Contd …… P/2

CSE 313

2. (a) The variable **counter**, is shared within process A and process B. The initial value is `counter = 0` before execution of either process. Here, `R0` is a register. **(7+8=15)**

Process A	Process B
LOAD (counter, R0)	LOAD (counter, R0)
ADD (R0, 1, R0)	ADD (R0, 2, R0)
STORE (R0, counter)	STORE (R0, counter)

i. Add semaphores (with initial values) so that the final value of counter **is 2.**

ii. Add semaphores (with initial values) so that the final value of counter is **not 3.**

(b) You wrote a piece of code with four threads (1-4) and four locks (A-D).

Thread 1 grabs Locks A and B (in some order); Thread 2 grabs Locks B and C (in some order);

Thread 3 grabs Locks C and D (in some order); Thread 4 grabs Locks D and A (in some order);

Is it possible that this code might result in deadlock? Briefly explain. **(5)**

(c) Consider the following program: **(5+10=15)**

```
1    int main{
2        int count =1;
3        int pid = 0, pid2 = 0;
4        if ((pid = fork())) {
5            count = count + 2;
6            printf("%d ", count);
7        }
8
9        if (count == 1) {
10           count++;
12             pid2=fork();
13           printf("%d ", count);
14       }
15
16       if (pid2) {
17           wait(pid2, NULL, 0);
18           count = count * 2;
19           printf("%d ", count);
20       }
21   }
```

i. How many processes are created during the execution of this program? Explain briefly.

ii. List all the possible outputs of the program.

3. (a) Let's examine a program having two threads: **(5)**

Thread 1	Thread 2

pending = 1;
while (pending) {
printf("hello\n");
}
	pending = 0;

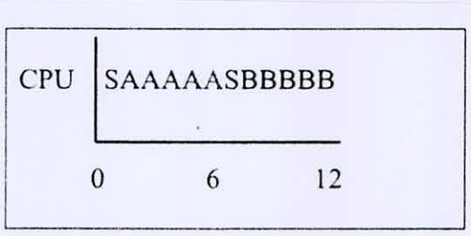
How could we re-write the code such that Thread 2 would only run after "hello" has been printed at least twice? You can use any synchronization primitive of your choice.

Contd P/3

CSE 313

Contd... Q. No. 3

(b) Scheduling policies can be easily depicted with some graphs. For example, let's say we run scheduler **S** for 1 time unit, job **A** for 5 time units, run scheduler **S** again for 1 time unit, and then run job **B** for time units. Our graph of this policy will look like this: **(5×3=15)**



i. Draw a similar graph of **ROUND-ROBIN** scheduling for jobs **A** (arriving at **T=0**), **B** (arriving at **T = 5**), and **C** (arriving at **T = 10**), each running for **6 time-units**. Assume a **2 time unit** time slice; also assume that the scheduler (**S**) takes 1 time unit to make a scheduling decision. Make sure to label the x-axis appropriately.

ii. What is the *average* ***RESPONSE TIME*** for jobs A, B and C?

iii. What is the *average* ***TURNAROUND TIME*** for jobs A, B and C?

(c) Consider the producer/consumer problem and the (broken) solution mentioned below. Briefly describe why solution is broken, and demonstrate it with a specific example of thread interleaving (*Hint: you can assume two consumers and one producer*). **(15)**

Producer	Consumer

`void *producer (void *arg) {`
`  int i;`
`  while (1) {`
`  mutex_lock (&mutex);  //p1`
`  if (count == MAX)  //p2`
`  cond_wait(&empty, &mutex);  //p3`
`  put (i);  //p4`
`  cond_signal (&full);  //p5`
`  mutex_unlock (&mutex);  //p6`
`  }`
`}`
	`void *consumer (void *arg) {`
	`  int i;`
	`  while (1) {`
	`  mutex_lock (&mutex);  //c1`
	`  if (count == 0)  //c2`
	`  cond_wait (&full, &mutex);  //c3`
	`  int tmp = get ();  //c4`
	`  cond_signal (&empty);  //c5`
	`  mutex_unlock (&mutex);  //c6`
	`  printf ("%d\n", tmp);`
	`  }`
	`}`

4. (a) A typical OS provides some APIs to create processes. **fork(), exec(),** and **wait()** can be used combinedly for that purpose. Write some code that uses these system calls to launch a new child process, have the child executed a program named "hello" (with no arguments), and have the parent wait for the child to complete. **(10)**

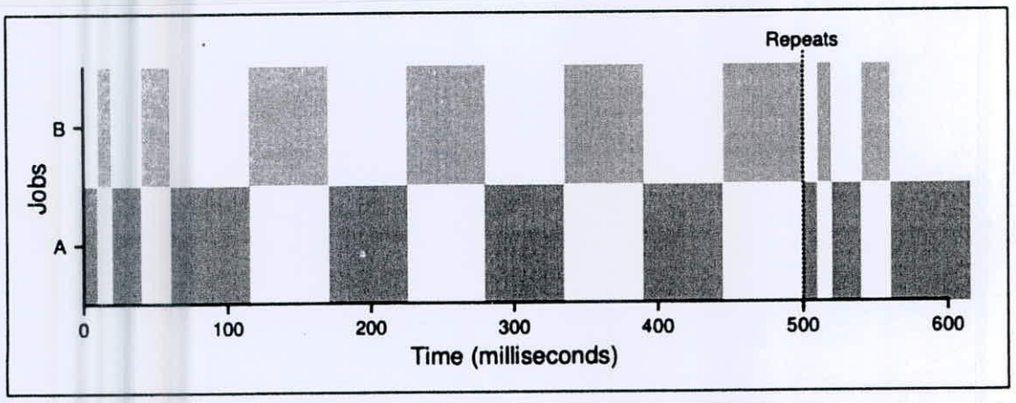
(b) Assume an OS with MLFQ (multi-level feedback queue) scheduler.

Here is a timeline of what happens when two CPU-bound (no I/O) jobs, **A** and **B** run: **(4×5=20)**

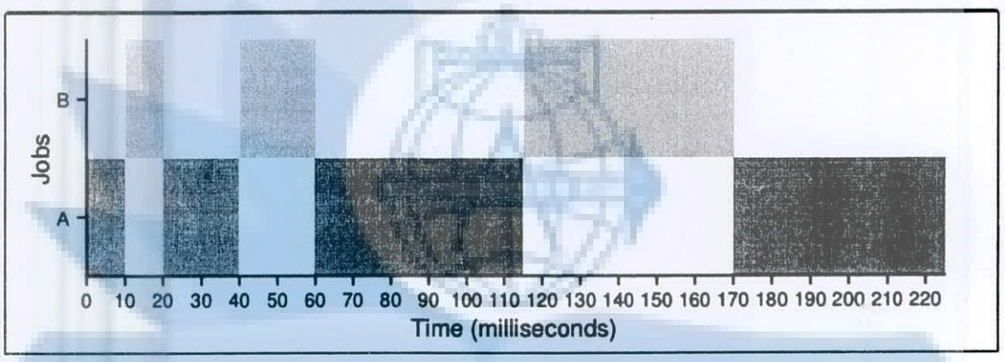
Contd P/4

CSE 313

Contd... Q. No. 4(b)



This figure shows when A and B run over time. Note that after 500 milliseconds, the behavior repeats, indefinitely (until the jobs are done). To help you further, a closeup of the first part of the graph is shown below:



Now, answer the following questions:

i. How many queues do you think there are in this MLFQ scheduler?

ii. How long is the time slice at the top-most (high priority) queue?

iii. How long is the time slice at the bottom-most (low priority) queue?

iv. How often do processes get moved back to the topmost queue?

v. Why does the scheduling policy MLFQ move processes to higher priority levels (i.e., the topmost queue) sometimes? Briefly explain.

(c) With the round robin (RR) scheduling policy, a question arises when a new job arrives in the system: should we put the job at the front of the RR queue, or the back? Does this subtle difference make a difference, or does RR behave pretty much the same way either way? Briefly explain. **(5)**

Contd P/5

CSE 313

SECTION – B

There are **FOUR** questions in this section. Answer any **THREE**.

5. (a) Alice has implemented a fast file system (FFS) with inodes having 12 direct pointers, 1 indirect pointer and 1 double indirect pointer. **(15)**

i. In a 1TB disk with 2KB blocks, how big of a file can be handled using Alice's file system?

ii. FFS divides the disk into multiple block groups. Now, can a 100 MB file be saved in Alice's file system? If so, then how all the data will be distributed on disk?

iii. If the disk spends 5ms time for positioning on the average and has 100MB/s transfer rate, how long will it take to read the file in (ii)?

(b) What is TLB in the context of memory virtualization? Show its structure. Mention its control flow using pseudocode. How does it manage context switches? **(10)**

(c) What is RPC? Suppose the user has written the following codes: **(10)**

```
int main() {
    …
    m = f1(x1, x2);
    …
}
int f1(char* a, char* b){
    …
    p = f2(*a);
}
float f2(int t){
    …
}
```


If the user wishes to run the function `f2` in a distributed manner, how will a RPC run-time library handle it? Write down which steps it needs to take and corresponding details that are needed to be taken care of in each step.

6. (a) Suppose the OS made the following 3 batch of block requests to a disk (each number represents block id): **(15)**

- 1, 40, 2, 15
- 10, 1, 13, 32, 2, 7
- 70, 49, 0, 6, 28

Contd …… P/6

CSE 313

Contd... Q. No. 6(a)

The requests were sent in such a way that one batch is requested after the previous one is handled. The disk has 1GB capacity, 4KB blocks, 8 blocks in each track and max seek time of 100ms. Now, calculate how much time the disk will spend on seeking, if the disk head is initially on the first track and the scheduling algorithm is

i. SSTF

ii. SCAN

iii. C-SCAN

(b) A program named `'test'` execute some code and writes some test to the console. The program is executed as: **(10)**

```
./test > /etc/out.txt
```

Now, write down the timeline for read/write operation in the file system. Assume that it is vsfs (very simple file system) and there is no cached value to aid.

(c) Bob has created the following files in an FFS (file sizes are written inside brackets). **(10)**

- `/a/f1 (2KB)`
- `/b/f2 (1KB)`
- `/c/b/f3 (10 KB)`
- `/c/f4 (7KB)`
- `/b/f5 (60KB)`

Suppose, the disk has 4KB blocks and 5 block group. If `/c/b` is symbolic link to the directory `/b`, show how the files will be saved with illustrative diagrams.

7. (a) You created an elegant device that has to deal with both big bursts of small I/O requests and very large I/O requests. While writing a device driver for it, what design decisions would you take and why? Elaborate the pros and cons of your design. **(15)**

(b) We need to write data in block 8, 11, 12 inode is in block 3, bitmap is in block 2. Journal is in blocks 24 to 31. Write down the journaling timeline if the system uses – **(10)**

i. Data Journaling

ii. Metadata Journaling

iii. Metadata Journaling with Checksum Optimization

(c) What is track skew? Why was it required in older disks? Why is it not good for newer disk models? **(5)**

(d) You know that RAID-4 has terrible Random Write performance. If the parity disk is mirrored in RAID-4, what would happen to its Random Write performance? What about Random Read or Sequential Read-Write? Explain your derivation. **(5)**

Contd P/7

CSE 313

8. (a) You ran the following rename operation: **(20)**

```
mv ~/a/foo.txt ~/b/bar.txt
```

Explain with illustrative figure what happens during this operation in-

i. Very Simple File System (VSFS)

ii. Log-structured File System (LFS)

(b) Using illustrative figures and pseudocodes, show how the OS talks to a canonical IO device while using DMA. **(10)**

(c) When using the swapping mechanism, the OS reserves some swap space on disk and works with directly. However, we know the OS already has File API to communicate with disk. Why do you think this discrepancy is there? **(5)**

L-3/T-2/CSE **Date: 18/04/2022**

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY, DHAKA

L-3/T-2 B. Sc. Engineering Examinations 2019-2020

Sub: **CSE 313** (Operating Systems)

Full Marks: 210 Time: 3 Hours

The figures in the margin indicate full marks

USE SEPARATE SCRIPTS FOR EACH SECTION

SECTION – A

There are **FOUR** questions in this section. Answer any **THREE** questions.

1. (a) Write down the steps of booting a computer. **(8)**

(b) Consider the code written in C shown in the following figure. Here `fork()` is an UNIX system call that creates a child process identical to the parent. Executing this code will generate a process tree. Each of the created processes will have its own copy of variable *i*. Your task is to draw this process tree. At each node of the tree, you have to mention the starting value of *i* for the corresponding process. Consider that the root process is called P0. **(10)**

```
int i=0;
int main(){
    for(;i<3;i++){
        fork();
    }
    return 0;
}
```

(c) Write down the steps in making a system call. **(10)**

(d) What are the advantages of hybrid implementation of threads? **(7)**

2. (a) State the four conditions of resource deadlock. **(6)**

(b) State the problem definition of the classical Dining Philosophers problem. Show that the problem meets all the four conditions for resource deadlock stated in 2 (a). **(12)**

(c) A solution to the Dining Philosophers problem is given in Figure for Q2(c). **(12)**

(i) Suppose that in the `put-forks(i)` function, variable `state[i]` was set to `THINKING` **after** the two calls to test, rather than before. How would this change affect the solution? Explain with an example.

(ii) Suppose that in the `test(i)` function, **`up`**`(&s[i])` is placed outside the "if" condition. How would this change affect the solution?

Contd P/2

CSE 313

Contd... Q. No.2

(d) Explain the race-condition that exists in the following solution to the producer-consumer problem. **(5)**

```
#define N = 100
int count = 0;
void producer (void) {
   int item;
   while (TRUE){
      item = produce();
      if (count == N) sleep();
      inset(item);
      count = count + 1;
      if (count == 1)
          wakeup(consumer);
}}
```

```
void consumer (void) {
int item;
while (TRUE){
     if (count == 0) sleep();
     item = remove-item();
     count = count - 1;
     if (count == N-1)
        wakeup(producer);
     consume(item);
}}
```

3. (a) A system has four processes and five allocatable resources. The current allocation and maximum needs are as follows: **(10)**

	Allocated	Maximum	Available
Process A	1 0 2 1 1	1 1 2 1 3	0 0 x 1 1
Process B	2 0 1 1 0	2 2 2 1 0	
Process C	1 1 0 1 0	2 1 3 1 0	
Process D	1 1 1 1 0	1 1 2 2 1	

What is the smallest value of x for which this is a safe state? Show the intermediate steps.

(b) What is the difference between livelock and starvation? **(5)**

(c) Draw the resource graph for the following scenario where A, B, C, D, and E denote processes and 1, 2, 3, 4, and 5 denote resource types. There exists only one resource of each type. Show the steps of the execution of the deadlock detection algorithm on the constructed graph starting from node B. **(20)**

(i) Process A holds 1 and 3, wants 2
(ii) Process B holds 4, wants 3 and 5
(iii) Process C holds nothing, wants 2
(iv) Process D holds nothing, wants 2
(v) Process E holds 5 and wants 1

4. (a) Discuss the difference between compute-bound and I/O bound process with figures. **(6)**

(b) Consider the following workload: **(24)**

Process	Priority	Duration (sec)	Arrival Time (sec)
P1	3	70	40
P2	1	40	0
P3	1	100	10
P4	2	50	70

Contd P/3

CSE 313

Contd... Q. No.4(b)

Draw the Gantt chart and calculate the average turnaround time for each of the following scheduling algorithms:

(i) Non-preemptive Shortest Job First

(ii) Round Robin with quantum 30 sec. Consider the processes enter FIFO queue according to their approval times.

(iii) Priority Scheduling. Within the same priority class, schedule according to the scheduling algorithm mentioned in (ii).

(c) Write down the goals of scheduling algorithms for Batch Systems. **(5)**

SECTION – B

There are **FOUR** questions in this section. Answer any **THREE** questions.

Unless otherwise specified, 1 k = 2^{10}, 1 M = 2^{10}, 1 G = 2^{30}.

5. (a) In 64-bit systems, 48 bit addressing is usually used. Calculate the amount of space needed (in GB) for a single-level page table for 48 bit addressing if the page size is 4 kB and each page table entry takes 4 bytes. Discuss the feasibility of such a page table and mention two (2) better alternatives. **(6+4=10)**

(b) Describe the advantages and disadvantages of memory-mapped I/O. **(10)**

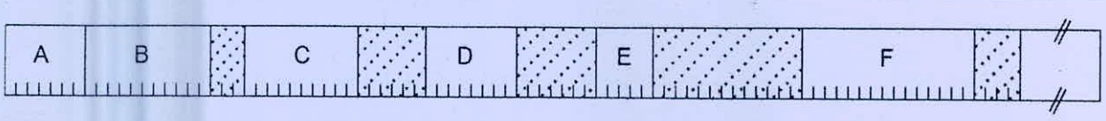
(c) Draw an omega switching network for 8 CPUs and 8 memory modules. Determine whether the following requests can be processed in parallel in this omega switching network and explain the reason. **(7+8=15)**

Request from CPU 000 to Memory 111 and

Request from CPU 110 to Memory 100

6. (a) The i-node of a Unix-like file system has 12 direct, one single-indirect and one double-indirect pointers. The disk block size is 4 kB and the disk block address is 32-bits long. Calculate the maximum possible file size for this file system. (Note that a single-indirect pointer points to a block of pointers that then point to blocks of the file's data. A double-indirect pointer points to a block of pointers that point to other blocks of the pointers that then point to blocks of the file's data.) **(10)**

(b) In the snapshot of memory given below, the dotted areas indicate holes and A, B, C, D, E, F are processes currently in memory. Create a linked list of processes and holes for swapping, assuming that there is no virtual memory. **(5+5=10)**



Contd P/4

CSE 313

Contd... Q. No.6(b)

A new process named G has to be allocated in memory now. The size of G is 4 allocation units. Mention the hole where G will be allocated if we follow:

(i) First-fit

(ii) Best-fit

(c) Define stable storage and mention the assumptions associated with it. Describe the three (3) basic operations a stable storage ensure. **(6+9=15)**

7. (a) Define precise interrupt and mention its four (4) properties. Describe the disadvantages of precise interrupts. **(5+5=10)**

(b) Mention the three (3) most essential properties of files. Explain how contiguous allocation of files may create both internal and external fragmentation in disk. **(3+7=10)**

(c) Mention the three key characteristics of a NUMA machine. In a directory-based NUMA multiprocessor having 1024 nodes, each memory address contains 48 bits. Each node consists of one CPU and 4 GB of RAM connected to the CPU via a local bus. Each cache line is 128 bytes long. The memory is statically allocated among the nodes. Calculate the directory overhead in percentage with respect to memory and discuss the feasibility of this system. **(3+12=15)**

8. (a) In a hypothetical machine, there are a total of 8 physical page frames. **(10)**

Page Table Index	Time of Last Use (t)	Referenced Bit	Modified Bit
0	214	1	1
1	381	0	1
2	402	1	0
3	289	1	1
4	409	0	0
5	160	1	1
6	315	1	0
7	387	0	1

At t = 430 and t = 440, two page faults occur. Using working set page replacement algorithm (with execution time approximation), identify the page frames that will be evicted. Assume the age threshold (τ) to be 50. Also assume that the clock interrupt to clear referenced bit last occurred at t = 425, and this interrupt occurs at 20 units time interval.

(b) Describe one (1) major advantage and one (1) major disadvantage of DMA. Briefly explain the major functions of device independent I/O software. **(6+14=20)**

(c) Write the full forms of the following abbreviations: **(5)**

NFTS, SMP, ECC, MBR, UDF

```
#define N           5                   /* number of philosophers */
#define LEFT        (i+N-1)%N           /* number of i's left neighbor */
#define RIGHT       (i+1)%N             /* number of i's right neighbor */
#define THINKING    0                   /* philosopher is thinking */
#define HUNGRY      1                   /* philosopher is trying to get forks */
#define EATING      2                   /* philosopher is eating */

typedef int semaphore;                  /* semaphores are a special kind of int */
int state[N];                           /* array to keep track of everyone's state */
semaphore mutex = 1;                    /* mutual exclusion for critical regions */
semaphore s[N];                         /* one semaphore per philosopher */

void philosopher(int i)                 /* i: philosopher number, from 0 to N-1 */
{
    while (TRUE) {                      /* repeat forever */
        think();                        /* philosopher is thinking */
        take_forks(i);                  /* acquire two forks or block */
        eat();                          /* yum-yum, spaghetti */
        put_forks(i);                   /* put both forks back on table */
    }
}

void take_forks(int i)                  /* i: philosopher number, from 0 to N-1 */
{
    down(&mutex);                       /* enter critical region */
    state[i] = HUNGRY;                  /* record fact that philosopher i is hungry */
    test(i);                            /* try to acquire 2 forks */
    up(&mutex);                         /* exit critical region */
    down(&s[i]);                        /* block if forks were not acquired */
}

void put_forks(i)                       /* i: philosopher number, from 0 to N-1 */
{
    down(&mutex);                       /* enter critical region */
    state[i] = THINKING;                /* philosopher has finished eating */
    test(LEFT);                         /* see if left neighbor can now eat */
    test(RIGHT);                        /* see if right neighbor can now eat */
    up(&mutex);                         /* exit critical region */
}

void test(i) /* i: philosopher number, from 0 to N-1 */
{
    if (state[i] == HUNGRY && state[LEFT] != EATING && state[RIGHT] != EATING) {
        state[i] = EATING;
        up(&s[i]);
    }
}
```

Figure for Q 2(c)

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY, DHAKA

L-3/T-2 B. Sc. Engineering Examinations 2018-2019

Sub: **CSE 313** (Operating Systems)

Full Marks: 180 Section Marks: 90 Time: 2 Hours (Sections A + B)

USE SEPARATE SCRIPTS FOR EACH SECTION

The figures in the margin indicate full marks.

SECTION – A

There are **FOUR** questions in this section. Answer any **THREE**.

1. (a) A computer provides each process with 65,536 bytes of address space divided into pages of 4096 bytes each. A particular program has a text size of 32,768 bytes, a data size of 16,386 bytes, and a stack size of 15,870 bytes. Will this program fit in the machine's address space? Suppose that instead of 4096 bytes, the page size were 512 bytes, would it then fit? Each page must contain either text, data, or stack, not a mixture of two or three of them. **(10)**

(b) Write an argument for and an argument against small page size. Derive the formula for determining the optimal page size in a system in terms of average process size (s) and each page table entry (e). **(4+6)**

(c) A small computer on a smart card has four page frames. At the first clock tick, the R (reference) bits are 0111 (page 0 is 0, the rest are 1). At subsequent clock ticks, the values are 0010, 1010, 1100, 1011, 1010, 1101, and 0001. If the aging algorithm is used with an 8-bit counter, give the values of the four counters after the last tick. If a page has to be replaced now, which page will be selected. **(8+2)**

2. (a) Describe the architecture and write down the advantages and disadvantages of RAID level 2. **(10)**

(b) Suppose disk requests come in to the disk driver for cylinders 8, 30, 20, 19, 40, 16, and 50, in that order. A seek takes 6 msec per cylinder. What is the total seek time needed if the Elevator algorithm is used for disk arm scheduling? Given, the arm is initially at cylinder 20 and moving upward. **(10)**

(c) Explain how a system can simulate multiple virtual clocks with a single physical clock. Use the following example for your explanation. The current time is 3000 and there are pending clock requests for time 3006, 5014, 5019, 5025, and 5039. **(10)**

3. (a) Describe the four conditions that are necessary for a resource deadlock to occur. Give an example to show that these conditions are not sufficient for a resource deadlock to occur. **(6+4)**

(b) Write down the four strategies that are used for dealing with deadlocks. Describe some ways of recovering from deadlocks. **(4+6)**

The more I study science the more I believe in God. – Albert Einstein

(c) Consider the following snapshot of a system. There are three processes, P1, P2 and P3 competing for three resources R1, R2, and R3. **(8+2)**

Total Resource Instances		
R1	R2	R3
3	2	5

Available		
R1	R2	R3
1	1	1

Process ID	Allocated		
	R1	R2	R3
P1	0	0	2
P2	1	0	2
P3	1	1	0

Process ID	Maximum Required		
	R1	R2	R3
P1	3	1	3
P2	1	2	3
P3	2	2	1

Now, answer the following questions using the banker's algorithm:

(i) Illustrate that the system is in a safe state by demonstrating an order in which the threads may complete.

(ii) If a request from process P1 arrives for (1, 1, 1), can the request be granted immediately?

4. (a) What is the problem of using single level page tables for very large virtual address spaces? How can this problem be eliminated using multilevel page tables? Explain with a suitable example. **(10)**

(b) What is the difference between 'cycle stealing' and 'burst mode' in the context of DMA operation? Describe three techniques that an operating system can use to reduce power consumption in hard disks. **(4+6)**

(c) Why is Banker's algorithm for deadlock avoidance not useful in practice? Describe how the circular wait condition can be eliminated in a system to prevent deadlocks. **(5+5)**

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY, DHAKA

L-3/T-2 B. Sc. Engineering Examination (January 2020 Term)

Subject: **CSE 313** (Operating System)

Full Marks: 180 Section Marks: 90 Time: 2 Hours (Sections A + B)

USE SEPARATE SCRIPTS FOR EACH SECTION

The figures in the margin indicate full marks.

SECTION – B

There are **FOUR** questions in this section. Answer any **THREE**.

5. (a) What are the steps of booting a computer? You can assume a BIOS system (NOT a UEFI system). **(12)**

   (b) What are the three cases when a CPU can operate in kernel mode? Briefly describe all three cases. **(3x6=18)**

6. (a) What are the four scenarios when a process may get created? Briefly explain all of them in one/two lines. **(4x3=12)**

   (b) Draw the process state diagram. **(9)**

   (c) What are the differences between a user thread and a kernel thread? **(9)**

7. (a) Explain the IPC primitive known as a barrier. Give an example scenario where a barrier can be useful. **(12)**

   (b) Explain the guaranteed scheduling algorithm. In particular, explain the goal of the algorithm, the metric used to determine which process will run next, the pros and cons. **(4+8+3+3=18)**

8. (a) What is a journaling filesystem? Give an example. **(12)**

   (b) How does UNIX-V7 handle very large files? Explain with proper figures. **(18)**

L-3/T-2/CSE **Date : 31/03/2019**

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY, DHAKA

L-3/T-2 B. Sc. Engineering Examinations 2017-2018

Sub : **CSE 313** (Operating System)

Full Marks : 210 Time : 3 Hours

The figures in the margin indicate full marks.

USE SEPARATE SCRIPTS FOR EACH SECTION

SECTION – A

There are **FOUR** questions in this section. Answer any **THREE**.

1. (a) Briefly describe first fit and next fit memory management algorithms. Which one performs better in practice? Consider a swapping system in which memory consists of the following hole sizes in memory order: 10 MB, 4 MB, 20 MB, 18 MB, 7 MB, 9 MB, 12 MB, and 15 MB. Which hole is taken for successive segment requests of (i) 12 MB, (ii) 10 MB, and (iii) 9 MB for best fit and worst fit? **(10)**

   (b) Suppose that a machine has 38-bit virtual addresses and 32-bit physical addresses. Now answer the following in this context. **(10)**

   i. What is the main advantage of a multilevel page table over a single-level one?

   ii. With a two-level page table, 16-KB pages, and 4-byte entries, how many bits should be allocated for the top-level page table field and how many for the next level page table field?

   (c) What are the advantages of segmentation over paging? Describe with necessary diagrams how MULTICS incorporates segmentation with paging. **(10)**

   (d) The average process size is s bytes, the page size is p bytes, and each page entry is e bytes. Deduce the equation for optimal page size. **(5)**

2. (a) Suppose that the WSClock page replacement algorithm uses at τ of 2 ticks, and the system state is the following: **(10)**

Page#	Time stamp	V	R	M
0	9	1	1	0
1	9	1	1	1
2	7	1	0	0
3	4	0	0	0
4	6	1	0	1

   Here, the three flag bits V, R, and M stand for Valid, Referenced, and Modified, respectively.

   (i) If a clock interrupt occurs at tick 10, show the contents of the new table entries with explanation.

   (ii) Suppose that instead of a clock interrupt, a page fault occurs at tick 10 due to a read request to page 3. Show the contents of the new table entries with explanation.

Contd. P/2

CSE 313

Contd ... Q. No. 2

(b) A computer has four-page frames. The time of loading, time of last access, and the R and M bits for each page are as shown below (the times are in clock ticks): **(10)**

Page	Time of loading	Last ref.	R	M
0	230	265	0	1
1	140	270	0	0
2	110	285	1	1
3	126	280	1	0

i. Which page will NRU algorithm replace?

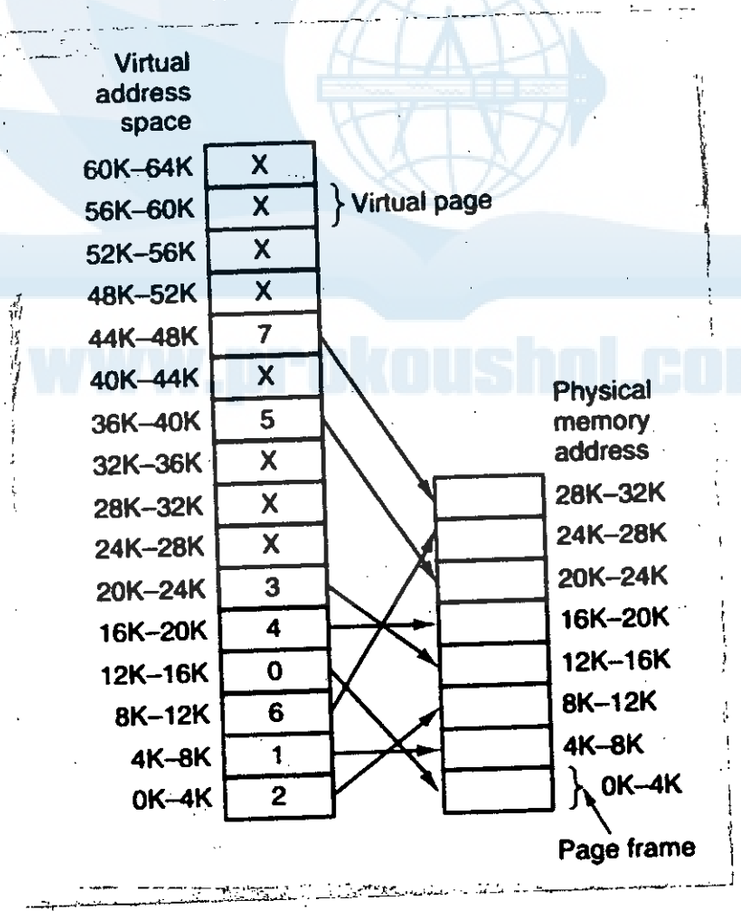
ii. Which page will FIFO algorithm replace?

iii. Which page will LRU algorithm replace?

iv. Which page will second chance algorithm replace?

(c) "Page fault cost varies" - is this statement true or false? Give appropriate reasoning of your answer. Briefly describe copy on write technique for the fork system call. **(10)**

(d) Using the page table below, give the physical address corresponding to each of the following virtual addresses: (i) 20, (ii) 4100, and (iii) 8300 **(5)**



CSE 313

3. (a) Briefly describe resource allocation graph with examples. Write down the algorithm that uses resource allocation graph to detect deadlock with one resource of each type. **(10)**

(b) Write down the algorithm for deadlock detection with multiple resources of each type with describing each of the matrices (C, R, E, A) the algorithm requires. **(10)**

Consider the following state of a system with four processes, P1, P2, P3, ad P4, and five types of resources, RS1, RS2, RS3,RS4, and RS5:

C =

0	1	1	1	2
0	1	0	1	0
0	0	0	0	1
2	1	0	0	0

R=

1	1	0	2	1
0	1	0	2	1
0	2	0	3	1
0	2	1	1	0

E=

2	4	1	4	4

A=

0	1	0	2	1

Using the above deadlock detection algorithm, find out whether there is a deadlock in the system and identify the processes that are deadlocked.

(c) Briefly describe the four conditions for resource deadlock. By attacking which condition, you can achieve the most feasible deadlock prevention technique? How can you prove the correctness of this technique? **(10)**

(d) Two processes, A and B, each need three records, 1, 2, and 3, in a database. If A asks for them in the order 1, 2, 3, and B asks for them in the same order, deadlock is not possible. However, if B asks for them in the order 3, 2, 1, then deadlock is possible. What fraction of all the combinations is guaranteed to be deadlock free? **(5)**

4. (a) Suppose you are writing a program to print your first name using the printer. Describe how you can achieve that using, (i) Programmed I/O, (ii) Interrupt driven I/O, and (iii) I/O using DMA with small code segments identifying key features. **(10)**

(b) Disk requests come in to the disk driver for cylinders 10, 22, 20, 2, 40, 6, and 38, in that order. A seek takes 6 ms per cylinder. How much seek time is needed for (i) First-come, first served, (ii) Shortest Seek First, (iii) Elevator algorithm (initially moving upward). In all cases, the arm is initially at cylinder 20. **(10)**

(c) What is meant by stable storage? Describe three operations to accomplish stable storage. Prove with example that stable writes can survive CPU crashes. **(10)**

(d) What is a cache coherence protocol? What happens (based on this protocol) if a CPU attempts to write a word that is in one or more remote caches? **(5)**

SECTION – B

There are **FOUR** questions in this section. Answer any **THREE**.

5. (a) What is **abstraction?** Briefly explain using an example how an operating system helps an application programmer by providing abstractions. **(5)**

(b) Briefly explain **dual mode operation.** How can a user program access services provided by an operating system? **(5+5=10)**

Contd. P/4

CSE 313

Contd ... Q. No. 5

(c) The following processes arrive at a system according to the table below. Assume that the process switching time is 0 second. Calculate **average turnaround** and **response time** for (i) FCFS Scheduling and (ii) Round Robin Scheduling with quantum of 2 seconds. **(10+10=20)**

Process	Arrival Time (sec)	CPU Burst Time (sec)
A	0	3
B	1.8	6
C	3.2	4
D	5.6	5
E	7.9	2

6. (a) Briefly describe the states of a typical process with a diagram. Mention how transitions occur between the states. **(10)**

   (b) Mention the advantages and disadvantages of implementing threads in the Kernel. **(4)**

   (c) How can you differentiate a program from a process? How a program becomes a process? **(6)**

   (d) What makes context switching a costly operation? How threads are helpful over processes in this regard? **(10)**

   (e) "Each user level thread needs its own stack" - do you agree with the statement? Why or why not? **(5)**

7. (a) How can you use semaphore to ensure (i) **mutual exclusion.** (ii) **access to limited resource,** and (iii) **synchronization?** Briefly explain each case with appropriate pseudo-code(s). **(15)**

   (b) When multiple co-operating processes communicate, special attention is required so that the system does not get stuck because of deadlocks. Providing proper synchronization mechanism to ensure liveliness and controlling access to shared resources are also some factors which makes the job quite difficult. **(20)**

   To pass this exam, you might have studied some of the classical IPC problems and their solution approaches. However here we are asking you to **design a new problem**. Think of a non-classical IPC problem. You don't need to solve the problem. Rather, **clearly mention the IPC issues in this problem.**

8. (a) What is a **race condition**? Why **cooperating processes** are necessary? **(5)**

   (b) What is **disk fragmentation**? Briefly explain how implementing files using contiguous allocation leads to this problem. **(10)**

   (c) Briefly describe a strategy to check consistency of blocks in a file system. **(10)**

   (d) Show the structure of an **i-node** in **UNIX V7** file system. **(10)**

L-3/T-2/CSE **Date : 12/03/2018**

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY, DHAKA

L-3/T-2 B. Sc. Engineering Examinations 2016-2017

Sub : **CSE 313** (Operating System)

Full Marks: 210 Time : 3 Hours

USE SEPARATE SCRIPTS FOR EACH SECTION

The figures in the margin indicate full marks.

SECTION – A

There are **FOUR** questions in this section. Answer any **THREE** questions.

1. (a) Draw the block diagram of the Unix System kernel and show the interaction among different subsystems. **(10)**

   (b) Why is it necessary to have two different stacks: user stack and kernel stack? Explain with an example. **(8)**

   (c) How is buffer list maintained in Unix system? Draw necessary diagram. What can happen to a process, if a buffer it is requesting is marked "delayed write"? **(7)**

   (d) Consider two events for which processes are at sleep states: (i) "Any buffer becomes free", (ii) "a specific buffer becomes free". If the specific buffer becomes free after some time, how is it handled by the operating system? Explain briefly. **(10)**

2. (a) Consider a case when a process P is waiting for a buffer #99 to become free (currently process Q is using it). Is it possible (due to some race condition) that suddenly process P discovers buffer #99 is not in the buffer cache list? Explain with necessary diagram. **(12)**

   (b) If somebody just changes the file access permissions (using chmod Command), what changes will be seen in file access times? Explain briefly. **(8)**

   (c) When a shortcut is made for a file, how many new inode(s) is used for this purpose? Give an example with respect to the changes in disk inode(s). **(7)**

   (d) "Superblock free inode list contains all the free inodes available in the disk" – do you agree with the statement? Why or why not? Explain. **(8)**

3. (a) Consider a modified table of contents (ToC) for inode having fourteen entries where the last entry is quadruple indirect pointer to data block, whereas other entries remain the same as before. If each data block is of 2K bytes and 32- bits are required for data block addressing, find out the maximum file size. **(10)**

   (b) Considering a similar structure as explained in Question # 3(a), locate the byte offset 4,50,000 in a file. Assume block numbers to explain and draw necessary diagram(s). **(8)**

   (c) What is the role of remembered inode while freeing an inode? Give examples. **(7)**

   (d) Write down the steps that need to be followed in wakeup system call. **(10)**

Contd P/2

CSE 313

4. (a) While writing to a file, is it possible that kernel allocates three new disk blocks (to just write one byte of data)? Explain the scenario with diagram. **(8)**

(b) Draw the process state transition diagram for Unix System. **(7)**

(c) What is the purpose of Register triple while managing regions? Show how it changes when kernel changes the size of a stack region by 2K bytes (consider each page to be 1K bytes). **(10)**

(d) Hoe does Kernel use u area in the Unix System? What are major information kept there? Explain briefly. If there are processes (P, Q, and R) running currently, how many u area are handled by kernel simultaneously? **(10)**

SECTION-B

There are **NINE** questions in this section. Answer any **SEVEN** questions.

5. (a) What is multiprogramming? Suppose you have a single CPU system. Will implementing multiprogramming improve the efficiency of your system? Justify your answer. **(6)**

(b) Compare the advantages and disadvantages of implementing threads in user space and kernel space. **(9)**

6. (a) What is meant by preemptive and non- preemptive scheduling algorithms? Give examples. **(4)**

(b) What is meant by **throughput** and **turnaround** time in process scheduling? Give examples. **(5)**

(c) Consider the following four processes: **(6)**

Process	Duration	Arrival time
P1	10	0
P2	4	5
P3	16	2
P4	6	4

Draw Gantt charts and then calculate average turnaround time and throughput for the processes using **First Come First Serve** scheduling algorithm.

7. (a) What is two phase locking? **(3)**

(b) Why multiple threads are used in a process instead of using multiple processes for the same task? **(5)**

(c) What is meant by the term **quantum** in context of process scheduling? What are the disadvantages if the quantum for scheduling is too large or too small? **(7)**

Contd P/3

CSE 313

8. (a) What is a multilevel page table? Explain with necessary figures. **(5)**

(b) Describe a scenario when you would want to use multilevel page tables instead of single level page table. **(4)**

(c) Do you think using multilevel page tables may reduce page faults in a system? Justify your answer. **(6)**

9. (a) Suppose you have four processes with length 48, 64, 120, and 28 bytes, respectively. The memory available for user space is 300 bytes. Discuss a way by which you can run all the four processes parallelly in your system minimizing number of memory accesses. **(5)**

(b) What is meant by **protection** and **relocation**? Can both protection and relocation be ensured using Base and Limit registers only? Explain your answer. **(10)**

10. (a) Suppose you are going to design Memory Management Unit for a system having 8GB memory with 64-bit virtual address space. The system should be robust enough to allow any number of processes of arbitrary size to run parallelly. Now, briefly discuss the memory management technique you will use for ensuring protection and relocation in your system. Give arguments justifying your choice. **(8)**

(b) Enumerate the four strategies used for dealing with deadlocks. Which one of them is the most popular and why? **(7)**

11. (a) What is meant by **deadlock** in operating system? Briefly discuss how deadlock can be prevented. **(15)**

12. (a) What is meant by safe and unsafe states in the context of process scheduling? **(3)**

(b) Suppose a system consists of three different processes A, B, and C. There are in total 5 printers and 6 scanners available to the system which the processes can use in a non-preemptable way. Process A is currently using 2 scanners and needs 3 more printers and 3 scanners in order to complete. Process B is currently using 3 printers and 3 scanners and needs 2 more printers and 3 more scanners in order to complete. Finally, process C is currently using a printer and needs one more printer and a scanner in order to complete. From this information, determine whether the processes can be in deadlock or not. **(12)**

13. (a) What is meant by memory mapped I/O and port mapped I/O? Discuss the advantages and disadvantages of each of them. **(10)**

(b) Write a short note on **Precise and Imprecise Interrupts**. **(5)**