0009

CS-2006: Operating Systems (CY)

Friday, 17th June, 2022

Course Instructors

Serial No:

Final Exam -Part A
Total Time: 1 Hour 15 mins
Total Marks: 80

[redacted]

Signature of Invigilator

[redacted] Roll No. [redacted] Section [redacted] Signature

DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

Instructions:

1. **This is only the Part-A of the final exam; you have maximum 75 minutes (1 hours and 15 minutes) for attempting this part. Once you return it (within 75 minutes) you will be provided with the Part-B.**
2. For MCQs, you MUST completely fill the box ☐ for the correct answer. Cutting, over writing, multiple answers or not correctly filling the box would be considered as incorrect. There is no negative marking. You MUST fill in the table given on next page, any answer written elsewhere would not be considered.
3. Attempt on question paper. Attempt all of them. Read the question carefully, understand the question, and then attempt it.
4. No additional sheet will be provided for rough work. Use the back of the last page for rough work.
5. If you need more space write on the back side of the paper and clearly mark question and part number etc.
6. After asked to commence the exam, please verify that you have Sixteen (16) different printed pages including this title page. There are a total of 2 questions.
7. Use permanent ink pens only. Any part done using soft pencil will not be marked and cannot be claimed for rechecking.
8. **CALCULATOR IS NOT ALLOWED.**

	Q-1	Q-2	Part2	Total
Marks Obtained	29	27.5	93	154.5
Total Marks	40	40	100	180

National University of Computer and Emerging Sciences

FAST School of Computing Spring-2022 Islamabad Campus

Question 1 [40 Marks]

Please completely fill the box ☐ for the correct answer like this ■ and NOT like this ☒ or this ☑. Cutting, over writing, multiple answers or not correctly filling the box would be considered as incorrect. There is no negative marking.

	A	B	C	D	E
01	☐	☐	☐	☐	■
02	■	☐	☐	☐	☐
03	☐	☐	☐	☐	■
04	☐	☐	☐	☒	■
05	☐	■	☐	☒	☐
06	☐	■	☐	☐	☐
07	☐	☐	☐	☐	■
08	☐	☐	■	☐	☐
09	☐	■	☐	☐	☒
10	☐	■	☐	☐	☐
11	☐	■	☐	☐	☐
12	☐	☐	☐	■	☐
13	☐	☐	☐	☐	■
14	■	☐	☐	☐	☐
15	☐	☐	■	☐	☐
16	☐	■	☐	☐	☐
17	■	☐	☐	☐	☐
18	☐	☐	☐	■	☐
19	☐	☐	■	☐	☐
20	☐	■	☐	☐	☐
21	☐	☐	■	☐	☐
22	☒	☐	■	☐	☐
23	☐	☐	■	☐	☐
24	☐	■	☐	☐	☐
25	☐	☐	■	☐	☐

	A	B	C	D	E
26	■	☐	☐	☐	☐
27	☐	■	☐	☐	☐
28	☐	☐	☒	☐	■
29	☐	☐	■	☐	☐
30	☐	☐	☐	■	☐
31	☐	■	☐	☒	☐
32	☐	☐	☒	■	☐
33	☐	☐	☐	☐	■
34	■	☐	☐	☐	☐
35	■	☐	☐	☒	☐
36	■	☐	☐	☒	☐
37	☒	■	☐	☐	☐
38	☐	☐	☐	☐	■
39	☐	■	☐	☐	☒
40	■	☐	☐	☐	☐

29

National University of Computer and Emerging Sciences

Question 1 [40 Marks]

1. What can be disadvantage of disabling the interrupts
   a. System may crash
   b. Unwise to give such power to user process
   c. The computer will not be able to service useful interrupts
   d. Both b & c
   e. a , b & c
2. A(n)_____ is the unit of work in a system.
   a. Process
   b. operating system
   c. timer
   d. mode
   e. a and b
3. Disabling interrupts for achieving synchronization can work for
   a. Distributed systems
   b. Multi-processing systems
   c. Any Unix based system
   d. Both a & c
   e. None of the above
4. A _____can be used to prevent a user program from never returning control to the operating system.
   a. portal
   b. program counter
   c. firewall
   d. timer
   e. None of Above
5. Which of the following is true about TSL
   a. Only one process can access memory during TSL
   b. TSL is uninterruptible
   c. Interrupts are disabled
   d. Both a & b
   e. None of the above
6. Which technique can prevent a process from wasting CPU cycles
   a. Busy waiting
   b. Sleep & wake up
   c. TSL
   d. Both a & b
   e. Both b & c
7. Which of the following true about mutual exclusion using machine instructions
   a. Applicable for systems with multiple CPUS
   b. Deadlock can occur
   c. Easy to verify
   d. Both a & c

e. All the above

8. Priority inversion problem can be solved through
   a. Priority double inversion
   b. No inversion
   c. Priority inheritance
   d. No priority
   e. Both a & d
9. Which of the following approaches for handling synchronization issue may result in deadlock?
   a. Lock variable
   b. Strict alternation
   c. Peterson solution
   d. Both a & b
   e. None of the above
10. If one thread opens a file with read privileges then:
   a. other threads in another process can also read from that file
   b. other threads in the same process can also read from that file
   c. any other thread cannot read from that file
   d. All of the above
11. The time required to create a new thread in an existing process is:
   a. greater than the time required to create a new process
   b. less than the time required to create a new process
   c. equal to the time required to create a new process
   d. sometimes b, sometimes c
   e. sometimes a, sometimes c
12. Which one of the following is not shared by threads?
   a. program counter
   b. stack
   c. code
   d. both a & b
   e. None of the above
13. Thread synchronization is required because:
   a. all threads of a process share the same address space
   b. all threads of a process share the same global variables
   c. all threads of a process can share the same files
   d. both a & b
   e. a, b, & c
14. Multithreading an interactive program will increase responsiveness to the user by :
   a. continuing to run even if a part of it is blocked
   b. waiting for one part to finish before the other begins
   c. asking the user to decide the order of multithreading
   d. both a & b
   e. None of these
15. If the kernel is single threaded, then any user level thread performing a blocking system call will
   a. cause the entire process to run along with the other threads
   b. cause the thread to block with the other threads running

c. cause the entire process to block even if the other threads are available to run
d. both a & c
e. None of the above

16. Because the kernel thread management is done by the Operating System itself :
    a. kernel threads are faster to create than user threads
    b. kernel threads are slower to create than user threads
    c. kernel threads are easier to manage as well as create then user threads
    d. both b & c
    e. both a & c
17. If a kernel thread performs a blocking system call, _____.
    a. the kernel can schedule another thread in the application for execution.
    b. the kernel cannot schedule another thread in the same application for execution.
    c. the kernel must schedule another thread of a different application for execution.
    d. the kernel must schedule another thread of the same application on a different processor.
    e. Both b & d
18. Which of the following is FALSE ?
    a. Context switch time is longer for kernel level threads than for user level threads
    b. User level threads do not need any hardware support
    c. Related kernel level threads can be scheduled on different processors in a multiprocessor system
    d. Blocking one kernel level thread blocks all other related threads
    e. Both a & b
19. If all processes I/O bound, the ready queue will almost always be _____, and the Short term Scheduler will have a _____ to do.
    a. full, little
    b. full, lot
    c. empty, little
    d. empty, lot
    e. none of the above
20. A problem encountered in multitasking when a process is perpetually denied necessary resources is called:
    a. Deadlock
    b. Starvation
    c. Inversion
    d. Aging
    e. Both a & b
21. The number of resources requested by a process :
    a. must always be less than the total number of resources available in the system
    b. must always be equal to the total number of resources available in the system
    c. must not exceed the total number of resources available in the system
    d. must exceed the total number of resources available in the system
    e. both c & d
22. For Mutual exclusion to prevail in the system :
    a. at least one resource must be held in a non sharable mode
    b. the processor must be a uniprocessor rather than a multiprocessor
    c. there must be at least one resource in a sharable mode
    d. both a & c

e. All of these

23. Which one of the following is the address generated by CPU?
    a. Physical address
    b. Absolute address
    c. Logical address
    d. None of the mentioned
    e. All of Above
24. The address of a page table in memory is pointed by:
    a. Stack pointer
    b. Page table base register
    c. Page register
    d. Program counter
    e. All of above
25. What is compaction?
    a. A technique for overcoming internal fragmentation
    b. A paging technique
    c. A technique for overcoming external fragmentation
    d. A technique for overcoming fatal error
    e. All of above
26. Demand paged memory allocation
    a. Allows the virtual address space to be independent of the physical memory
    b. Allows the virtual address space to be a multiple of the physical memory size
    c. Allows deadlock to be detected in paging schemes
    d. None
    e. All of Above
27. In memory management, a technique called as paging, physical memory is broken into fixed-sized blocks called _____.
    a. Pages
    b. Frames
    c. Blocks
    d. Segments
    e. None of Above
28. The degree of Multiprogramming is controlled by
    a. CPU Scheduler
    b. Context Switching
    c. Long-term Scheduler
    d. Medium term Scheduler
    e. All of Above
29. Semaphore can be used for solving _____.
    a. Wait & signal
    b. Deadlock
    c. Synchronization
    d. Priority
    e. All of Above

30. Which of the following statements about critical sections is false?
    a. Only one thread at a time can execute the instructions in its critical section for a particular resource.
    b. If one thread is already in its critical section, another thread must wait for the executing thread to exit its critical section before continuing.
    c. Once a thread has exited its critical section, a waiting thread may enter its critical section.
    d. All threads must wait whenever any critical section is occupied.
    e. All of the above

31. The Banker's algorithm is used:
    a. To prevent deadlocks
    b. To detect deadlocks
    c. To rectify a deadlocked state
    d. None of the above
    e. All of the above

32. The operating system maintains a data structure called _____ that keeps track of how many frames have been allocated, how many are there, and how many are available.
    a. Memory Manager
    b. Process Address Space
    c. Page Table
    d. None of the above
    e. All of the above

33. The mechanism that bring a page into memory only when it is needed is called?
    a. Fragmentation
    b. Pre Paging
    c. Page in Need is a Page Indeed
    d. Inverted offset replacement
    e. None of the above

34. Which of the following statement is TRUE in relation to Paging?
    a. It removes external fragmentation
    b. It removes internal fragmentation
    c. All of the above
    d. None of the above

35. If there are 6 user processes sharing memory, how many page tables are there in the complete memory being manage by the OS?
    a. Obviously six
    b. More than six, as some processes have more than one page-tables
    c. Process contain threads and thus more than 6
    d. None of the above

36. If there are 6 user processes sharing memory, how many page tables registers are needed?
    a. Obviously six, one per each process
    b. Can be any number, OS has own registers as well
    c. All of the above

d. None of the above

37. Which of the following statement is TRUE in relation to MMU and TLB?
    a. MMU uses TLB for address translation
    b. TLB is accessed first and then MMU
    c. None of the above

38. Page fault occurs when
    a. The page is corrupted by application software.
    b. The page is in memory.
    c. One tries to divide a number by 0.
    d. Page table is corrupted
    e. None of the above

39. The mechanism that bring a page into memory only when it is needed is called
    a. Segmentation
    b. After demand Paging
    c. Page Replacement Scheme
    d. Inverted offset replacement
    e. None of the above

40. What happens if page size is too large?
    a. This would lead to more internal fragmentation
    b. This will result in more external fragmentation
    c. The need for multi-level page tables would be more evident
    d. None of the above

National University of Computer and Emerging Sciences

Question 2 [40 Marks]

Problem 1 - Assume the following code is compiled and run on a modern Linux machine:

```
int main(int argc, char *argv[]){
  printf("a");
  fork();
  printf("b");
  return 0;
}
```

a. Assuming fork() succeeds and printf() prints its outputs immediately (no buffering occurs), what are possible outputs of this program? **[2 Marks]**

In case of no buffering.
1) abb

In case of buffering.
1) abb
2) ~~bab~~

b. Assuming fork() might fail (by returning an error code and not creating a new process) and printf() prints its outputs immediately (no buffering occurs), what are possible outputs of the same program as above? **[2 Marks]**

ab

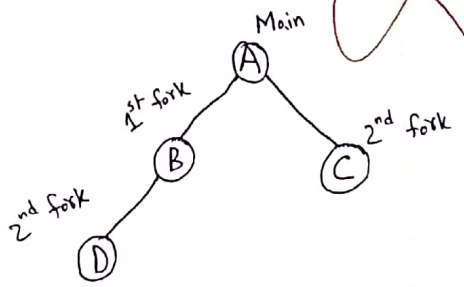
Problem 2 – Look at the code below and draw the process tree. Incorrect tree will lead to Zero Marks **[5 Marks]**

```
int i = 0;
main()
{
    printf("%d\n", i);
    fork();
    i = i + 1;
    printf("%d\n", i);
    fork();
    i = i + 1;
    printf("%d\n", i);
}
```

1st

2nd

Zero



Problem 3- Write a program to communicate between parent and child using pipes where child writes into the pipe while the parent reads from it. Given that you should handle the case below:
"Parent tried to read from the pipe before the child writes to it. When this happens, the parent is said to block, or sleep. until data arrives to be read". **[9 Marks]**

The resultant output will be something similar to the following:
PARENT: reading from pipe
CHILD: writing to the pipe
CHILD: exiting
PARENT: read "test"

```
int main() {
    int fd[2];
    if (pipe(fd) == -1) {
        printf("Pipe not created!\n Exiting!!\n\n");
        exit(0);
    }
    pid_t pid = fork();
    if (pid < 0) {
        printf("fork failed\nexiting!!\n\n");
        exit(0);
    }
    if (pid == 0) { //child
        close(fd[0]);
        while(1) {
            char buffer[500];
            printf("Enter messege: ");
            scanf("%[^\n]%*c", buffer);
            printf("Writing %s to pipe\n CHILD\n\n", buffer);
            write(fd[1], buffer, sizeof(buffer));
            if (strcmp(buffer, "exit") == 0) {
                printf("Exiting\n CHILD\n\n");
                exit(0);
            }
        }
    else {
        close(fd[1]);
        while(1) {
            char buffer[500];
            printf("Reading from Pipe\n Parent\n\n");
            read(fd[0], buffer, sizeof(buffer)); // blocking statement
            printf("Messege Received: %s", buffer);
            if (strcmp(buffer, "exit") == 0) {
                printf("Exiting\n Parent\n\n"); exit(0); }
        }
    }
}
```

Problem 4 – Look at the code below

```
void mythread(void *arg) {
    return;
}

thread_t p1, p2;
thread_create(&p1, mythread, NULL);
thread_create(&p2, mythread, NULL);
thread_join(p1);
thread_join(p2);
```

When this code runs, how many total threads can there be (maximum) at a given moment intime? **[2 Marks]**

Supposing if p1 and p2 ~~have~~ haven't been scheduled yet.
There can be 3 threads maximum.
1 main thread and 2 threads that are created during execution.

2

Problem 5- Look at the code below:

```
void mythread(void*arg)
    {
    int result = 0;
    result = result + 200;
    printf("result %d\n", result);
}

thread_t p1, p2;
thread_create(&p1,mythreadNULL);
thread_create(&p2,mythread,NULL;
thread_join(p1);
thread_join(p2);
```

When this code runs, and resultis printed, what value will be printed? **[2 Marks]**

Supposing p1 is executed when "thread_join(p1)" statement is executed and p2 is scheduled after execution of p1.

p1: result 200
p2: result 200

2

roblem 6 - The pseudo-code below creates 4 threads that each run threadfunc on a shared variable.

```
std::mutex mutex;

void threadfunc(void* x) {
   for (int i = 0; i != 5; ++i) {
       mutex.lock();
       *x += 1;
       mutex.unlock();
   }
}

int main() {
   pthread_t tid[4];
   unsigned n = 0;
   for (int i = 0; i != 4; ++i) {
        thread_create(&tid, Null, threadfunc, &n);
   }
   printf("%u\n", n);

   // process exits, all threads terminate
   exit(0);
}
```

a. Why does this code sometimes fail to produce the expected result of 20? **[2 Marks]**

Because print statement in main may execute before complete execution of all threads.

b. Fix the above code to produce expected result 20 **[3 Marks]**

Problem 7 - Complete the prototype of pthread_create function. **[5 marks]**

pthread_create(pthread_t * threaded ,
pthread_attr_t * attr ,
Void * (* start_routine) (void *) ,
Void * arg);

4.5

Problem 8 - Add the missing line in TSL so that it can work as expected. **[1 Mark]**

```
boolean TestAndSet (boolean *target)
{
    bool rV = *target
    *target = TRUE;
    return rv:
}
```

01

Problem 9 -Use the exchange instruction to avoid critical section **[1 mark]**

```
// Lock is a shared variable and set to False
// key is local variable for each pi
key:=1
while(key)
    exchange(lock,key);
Critical_section();
Lock =0;
Remainder_section();
```

01

roblem 10 – You are given three semaphores above. Use them at correct positions to solve the producer consumer problem below. **[6 marks]**

Semaphore A(BUFFER_SIZE)
Semaphore B(0)

Semaphore C(1)

```
nt BUFFER_SIZE = 100;
nt count = 0;
void producer(void) {
        int item;
        while(TRUE) {

                produce_item(&item);
                wait(A);
                wait(C);

                enter_item(item);
                signal(C);
                signal(B);

        }
}
void consumer(void)
{       int item;
        while(TRUE) {

                wait(B);
                wait(C);
                remove_item(&item);
                signal(C);
                signal(A);

                consume_item(&item);
        }
}
```

06

National University of Computer and Emerging Sciences

FAST School of Computing Spring-2022 Islamabad Campus

CS-2006: Operating Systems (CY)

Serial No:

Final Exam – Part B
Total Time: 105 Minutes
Total Marks: 100

Friday, 17th June, 2022

Course Instructors

Signature of Invigilator

Mr. Shoaib Mehboob

Fahad Waheed	20I-0651	T	
Student Name	Roll No.	Section	Signature

DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

Instructions:

1. Attempt on question paper. Attempt all of them. Read the question carefully, understand the question, and then attempt it.
2. No additional sheet will be provided for rough work. Use the back of the last page for rough work.
3. If you need more space write on the back side of the paper and clearly mark question and part number etc.
4. After asked to commence the exam, please verify that you have **eighteen** different printed pages including this title page. There are a total of 7 questions.
5. Use permanent ink pens only. Any part done using soft pencil will not be marked and cannot be claimed for rechecking.
6. **CALCULATOR IS NOT ALLOWED.**
7. **DO NOT DETACH THE LAST ROUGH PAGE.**

	Q-1	Q-2	Q-3	Q-4	Q-5	Q-6	Q-7	Total
Marks Obtained	25	8	12	18	13	8	9	93
Total Marks	25	10	12	18	13	10	12	100

National University of Computer and Emerging Sciences

FAST School of Computing **Spring-2022** **Islamabad Campus**

Question 1 [25 Marks]

Five processes A, B, C, D and E arrive as per their given arrival time. order at the same time wi the following CPU bursts. The process is immediately eligible for scheduling i.e. if a process arrive at 1, it can be scheduled at 1.

Process	CPU Burst	Arrival Time
A	7	5
B	2	4
C	3	2
D	6	3
E	4	6

Fill in the entries of the following table with **total** turnaround time, **total** waiting time, **tot** response time, and throughput for each indicated scheduling policy. Ignore context switching c any other overhead. **Rough work depicting the calculations (i.e. correct Gantt chart matchin your final answer) is mandatory. Filling the table without correct rough work will lead t zero marks in the question.** [2 mark – each correct entry in the table]

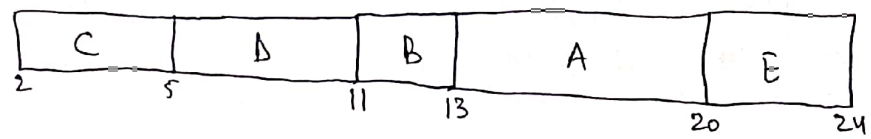
	Total turnaround time	Total waiting time	Total response time
First Come First Serve	53	31	31
Shortest Job First	44	22	22
Shortest Remaining Time First	44	22	22
Round Robin (quantum = 2)	61	39	12

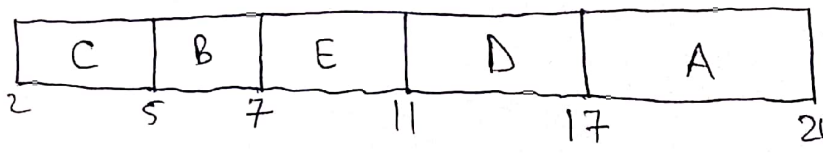
FCFS **Rough work for this part: (You may use the next page too)** SJF

Process	A.T	CPU	CT	WT RT	TAT	W.T R.T	T.A.T
C	2	3	5	0	3	0	3
D	3	6	11	2	8	8	14
B	4	2	13	7	9	1	3
A	5	7	20	8	15	12	19
E	6	4	24	14	18	1	5

24

FCFS



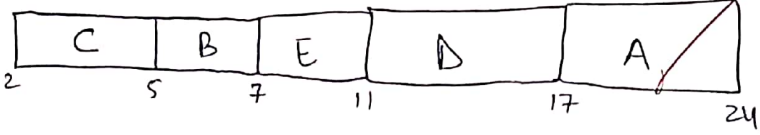


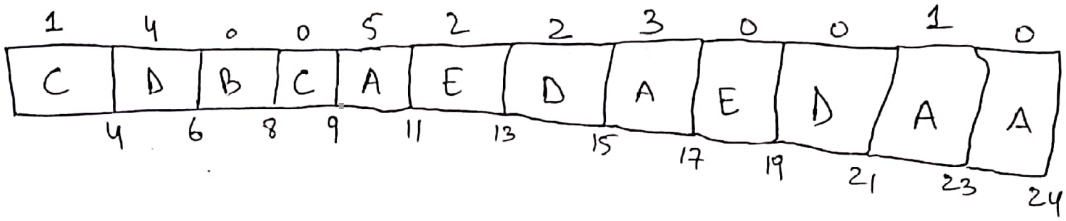
Round Robin

Process	A.T	CPU	R.T	W.T	T.A.T
C	2	3	0	4	7
D	3	6	1	12	18
B	4	2	2	2	4
A	5	7	4	12	19
E	6	4	5	9	13

SRTF

Similar to SJF as no-preemption occured





d. Based on the above scores, which of the policy is the best one in your opinion? **[1 mark]**

For these kind of processes SJF/SRTF is best although due to arrival time of these processes, there are no-premption in SRTF.

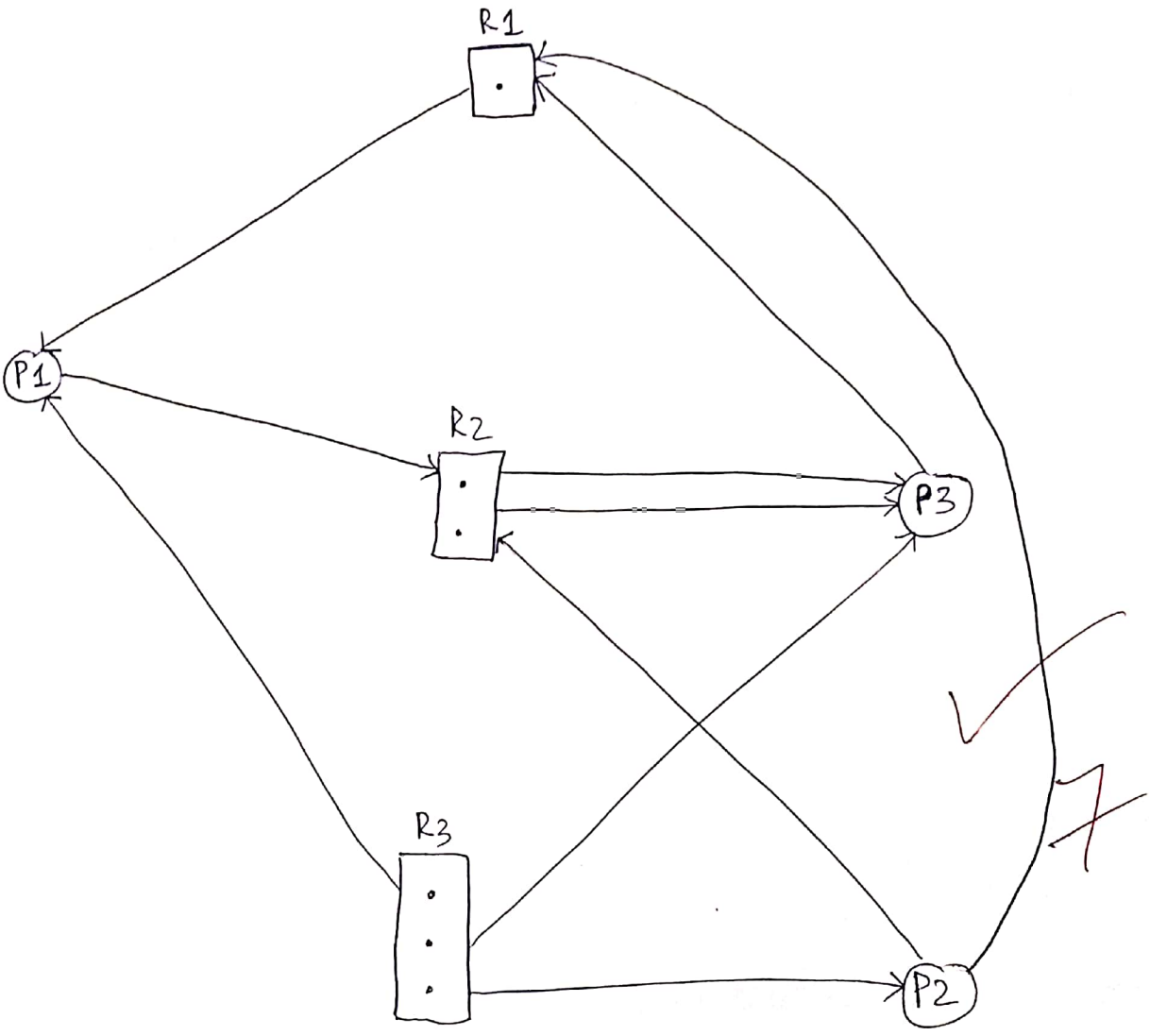
1

National University of Computer and Emerging Sciences

Question 2 [10 Marks]

A system has three processes (P1, P2, P3) and three reusable resources (R1, R2, R3). There is one instance of R1, two instances of R2 and three instances of R3. P1 holds an R1 and an R3 and is requesting an R2. P2 holds an R3 and is requesting an R1 and an R2. P3 holds two R2 and an R3 and is requesting an R1.

a. Draw the resource allocation graph for this system. Use the style of diagram from the lecture notes. **[7 marks]**



b. Is this system deadlocked? If so, state which processes are involved. [3 marks]

Yes, the system is under deadlock.
All the three process are involved as it is impossible to execute one of them under current state.

1

Question 3 [12 Marks]

Suppose we have the following resources:

- 5 processes P0 through P4
- 3 resource types A (6 instances), B (9 instances) and C (5 instances).

Below Snapshot is taken at time T0

	Max			Allocation		
	A	B	C	A	B	C
P0	6	7	3	1	1	1
P1	2	2	2	1	1	2
P2	2	6	3	0	3	0
P3	2	2	2	2	1	1
P4	4	6	3	1	1	1

a. Calculate the available vector. **[2 mark]**

Allocated instances of A = 1 + 1 + 0 + 2 + 1 = 5
" B = 1 + 1 + 3 + 1 + 1 = 7
" C = 1 + 2 + 0 + 1 + 1 = 5

2

Available Vector

$$\begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}$$

b. Calculate the Need matrix. **[2 marks]**

Need

	A	B	C
P0	5	6	2
P1	1	1	0
P2	2	3	3
P3	0	1	1
P4	3	5	2

2

c. Suppose at time T0, now Process P2 requests (1,2,0) from its Max need. Should this request be granted? Why or why not? **[2 marks]**

No, this request should be denied because we have only (0,2,1) available at T0 if we allocate these to P2 then no request for other processes will be accepted due availability of no resources. Hence, there is a high chance of deadlock if request (1,2,0) of P2 is accepted.

2

d. Is the following sequence "P1 -> P3 -> P2 -> P4 -> P0" a safe sequence. Show your detailed computation. Only saying Yes/No (without proper calculations will result in 0 marks for this part). **[6 marks]**

Available: A = 1, B = 2, C = 0

	Max			Need			Allocation						
	A	B	C	A	B	C	A	B	C	A	B	C	
P0	6	7	3	5	6	2	1	1	1	6	9	5	(5)
P1	2	2	2	1	1	0	1	1	2	2	3	2	(1)
P2	2	6	3	2	3	3	0	3	0	4	7	3	(3)
P3	2	2	2	0	1	1	2	1	1	4	4	3	(2)
P4	4	6	3	3	5	2	1	1	1	5	8	4	(4)

Hence, given sequence is a safe sequence.
So, answer is Yes.

6

National University of Computer and Emerging Sciences

FAST School of Computing Spring-2022 Islamabad Campus

Question 4 [18 Marks]

Consider the following string of memory references in terms of requested page number: **[18 marks]**

1, 2, 3, 4, 5, 3, 4, 1, 6, 7, 8, 7, 8, 9, 7, 8, 9, 5, 4, 5, 4, 2.

Assume the total number of physical frames in the main memory is greater than the number of pages requested. For the above string, how many minimum/compulsory page faults will be needed (for any page replacement algorithm). **[1 mark]**

As number of frames > number of pages

Minimum/Compulsory page faults = unique page numbers = 9 (Nine)

1/1

Now assume that the total number of frames in main memory are 5. Fill in the tables below with using the FIFO, OPTIMAL and LRU (Least Recently Used) page replacement algorithms for the given memory by filling in the table with page numbers. The top row is the string of memory references, and each row contains the page numbers that reside in each physical frame F1 – F5 after each memory reference. **For readability purposes, please only fill in the table entries that have changed (i.e. when any page fault occurs) and leave the unchanged entries blank. If several pages meet the replacement criteria, replace the one with the smallest frame number. You can perform the needed rough work in the space after the given tables. Marks will be awarded for correct entries in the tables below. [17 marks]**

FIFO

	1	2	3	4	5	3	4	1	6	7	8	7	8	9	7	8	9	5	4	5	4	2
F1	1								6											5		
F2		2								7												2
F3			3								8											
F4				4										9								
F5					5														4			

Rough work for FIFO

6

OPTIMAL

	1	2	3	4	5	3	4	1	6	7	8	7	8	9	7	8	9	5	4	5	4	2
F1	1								6		8											2
F2		2												9								
F3			3							7												
F4				4																		
F5					5																	

Rough work for OPTIMAL

LRU

	1	2	3	4	5	3	4	1	6	7	8	7	8	9	7	8	9	5	4	5	4	2
F1	1																	5				
F2		2							6										4			
F3			3								8											
F4				4						7				9								2
F5					5																	

Rough work for LRU

Question 5 [13 Marks]

Given fixed memory partitions of 100K, 500K, 200K, 300K, and 600K (in order), how would each of the first-fit, next-fit, best-fit, and worst-fit algorithms place processes of 212K, 417K, 112K, and 426K (in order)?

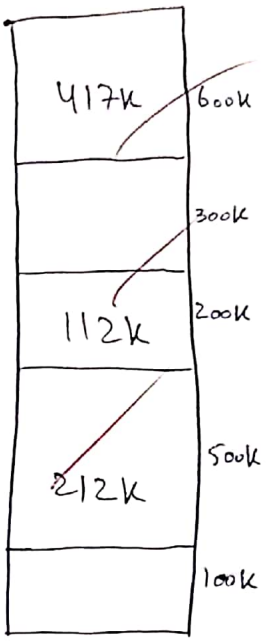
First-fit: **[3 marks]**

212K, 417K, 112K, 426K

Now; 426K will have wait or

212K/417K process should be swapped with 426K process (based on priority).

3

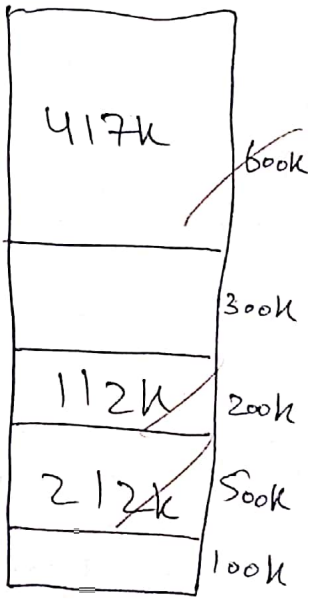


Next-fit: **[3 marks]**

212K, 417K, 112K, 426K

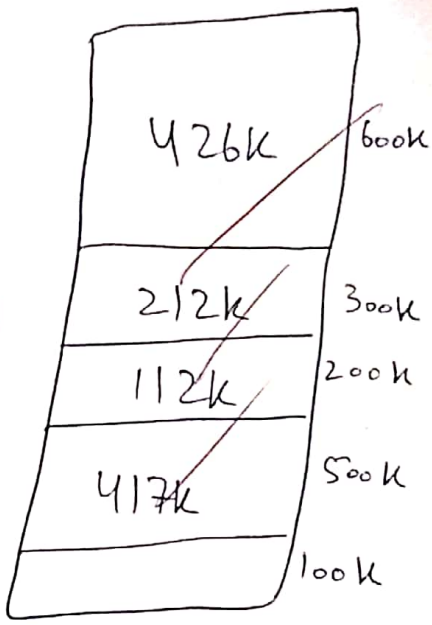
Now; again 426K will have to wait or should be swapped with 417K/212K.

3



Best-fit: **[3 marks]**

212K, 417K, 112K, 426K

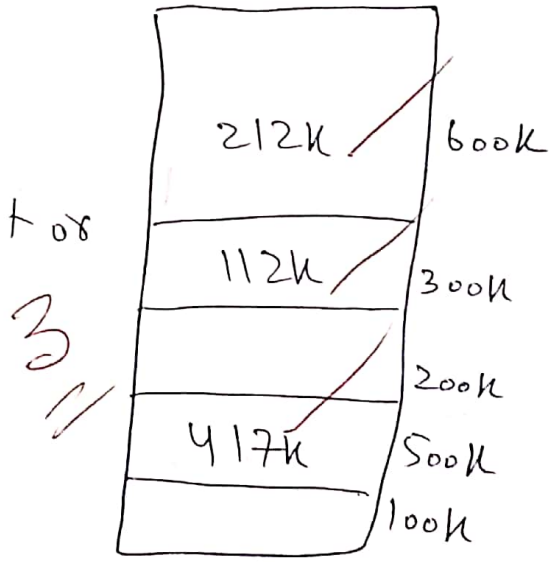


3

Worst-fit: **[3 marks]**

212K, 417K, 112K, 426K

Now; again 426K will have to wait or swapped with 417K/212K.



3

Which algorithm makes the most efficient use of memory for the above given data? **[1 mark]**

Best-fit as all the processes are successfully loaded into the memory.

1

Question 6 [10 Marks]

Consider a demand-paged virtual memory system with a 24-bit addressing scheme, 8KB page size, and 8 MB of main memory. Assume the access time for main memory is 100 ns and the page fault service time is 10ms. Assume the page table is stored in main memory.

a. If no TLB is used, how long does it take for a user program to load a byte from main memory into a CPU register? Assume the desired page is already in main memory. **[4 marks]**

= 2 * accesstime of main memory
= 2 * 100ns
= 200ns

∴ 1 for page table
∴ 1 for actual byte

4

b. Suppose we add a TLB to the system to speed up logical-to-physical address translation. With a TLB access time of 10 ns and a TLB hit rate of 0.9, what is the effective memory access time? Assume we only do a main-memory page table lookup after a TLB miss, and that the desired pages are always in main memory. **[4 marks]**

In case of hit:
= 10ns + 100ns
= 110ns

In case of miss:
= 10ns + 100ns + 100ns
= 210ns

2

Rest ?

c. With a TLB, what is the time required to load the page into memory if there is a page fault. [2 marks]

= 10ns + 100ns + 10ns

= 10ns + 100ns + 100000000ns

= 100000120ns

2

Question 7 [12 Marks]

Neuron Inc. hires you to design the memory management system for a new cell phone with 32-bit logical address space where page size is allocated 8 Kbits. Given Size of physical memory is 1Gbits. Suppose that you decide to use a single-level page table.
Answer the following questions: (All answers must be accompanied with correct and complete calculations)

a. How many bits would represent page number in logical address space? **[2 marks]**

$m = 32$ $n = 13$ $\because 2^{10} = 1KB$ $2^3 = 8$

$m - n = 32 - 13$

Hence; 19 bits (left) represent page numbers in Logical Address Space. ✓

2

b. How many (bits) would represent offset within a page in logical address space? **[2 marks]**

$\because 2^{10} = 1KB$ $2^3 = 8$

Right 13 bits ✓

2

c. How many total pages would be created in logical address space? **[2 marks]**

$2^{19} = 524288$ pages ✓

2

d. How many total frames would be created in physical address space. [2 marks]

$2^{30} = 1\,GiB$

$30 - 13 = 17$

$2^{17} = 131072$ frames

2

f. How much maximum memory is needed to store the page table of each process? [4 Marks]

Assuming size of 1 entry = 3 bytes

Table size = # of pages * size of 1 entry

$= 524288 * 3$

$= 1572864$ Bytes

1