**Review for final test**

**Processes**

Process states

Process creations

Interprocess communication

Reasons

Shared-memory system

Message passing system

**Multithreading**

Benefits of multithreaded programming

Multithreading models

**Scheduling algorithms**

Scheduling Criteria

FCFS

SJF

SRTF

Priority

**Deadlocks**

Necessary Conditions for deadlock

Methods for handling deadlocks

Prevention

Avoidance

Resource allocation graph algorithm

Banker’s algorithm

Detection

Wait-for-graph

Deadlock-detection algorithm

**Process Synchronization**

Requirements of solution to the critical-section

TestAndSet() instructions

Readers-writers problem

The bounded-buffer problem

Monitors

**Main Memory**

Binding of data to physical memory

Logical address space vs physical address space

Memory allocation

Dynamic storage allocation

Strategies of selecting the free space

Paging system

The method of implementing paging system

Effective access time

Memory protection in a paging system

Different structures of the page table

Segmentation system

Memory protection in segmentation systems

Fragmentation in both systems

Advantages one system over another one

Virtual memory

Benefits of virtual memory

Demand paging

Steps in handling a page fault

Effective access time

Copy-on-write

Page replacement methods

Second chance algorithm

Allocation of frames

Allocation algorithms

Global vs local

Thrashing

Working-set model

Prepaging

Page size

Program structure

**File system interface and implementation**

File operations

Access methods

Different schemes for defining the logical structure of a directory

Consistency semantics: Unix and Session

**Mass storage structures**

Disk scheduling

Selection of disk scheduling algorithm

Bad blocks

Average I/O time : latency, rotation, transfer

**Distributed systems**

What are motivation factors for building distributed systems?

Network OS vs. Distributed OS

Describe the Remote Procedure Call model for Distributed OS

Network topology

Communication structures

Communication protocol

Problem 1. Given the following set of processes with, arrival times, burst times, priorities

|  |  |  |  |
| --- | --- | --- | --- |
| **Process** | **Arrival time** | **Burst time** | **Priority** |
| **P1** | **0** | **9** | **4** |
| **P2** | **1** | **6** | **2** |
| **P3** | **2** | **1** | **1** |
| **P4** | **3** | **5** | **3** |
| **P5** | **5** | **2** | **2** |

1. Draw four Gant charts that illustrate the execution of these processes using the following scheduling algorithms: FCFS, SJF (Shortest-Job First)(non-preemptive), non-preemptive priority, preemptive Shortest-Remaining–time first

FCFS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P1 | P2 | P3 | P4 | P5 |

0 9 15 16 21 23 23 23

SJF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P1 | P3 | P5 | P4 | P2 |

0 9 10 12 17 23 23 23

Priority

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P1 | P3 | P2 | P5 | P4 |

0 9 10 16 18 23 23 23

SRTF

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | P2 | P3 | P2 | P5 | P2 | P4 | P1 |

0 1 2 3 5 7 10 15 23

1. What is the turnaround time of each process for each of the scheduling algorithms in part 1?

FCFS: P1: 9, P2: 14, P3:14. P4:18, P5: 18

FJF: 9,22,8,14,7

Priority:9,15,8,20,13

SRTF: 23,9,1,7,2

1. What is the waiting time of each process for each of the scheduling algorithms in part 1?

SRTF: 14,5,0,7,0

1. What is the average waiting time over all processes for each of the scheduling algorithms in part 1?

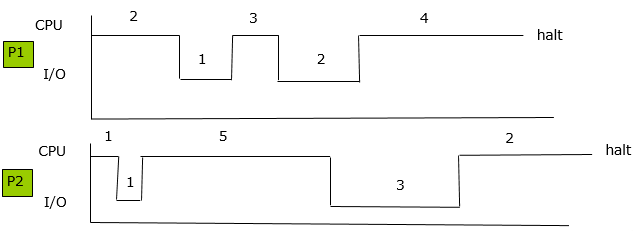
SRTF: (14+5+0+7+0)/5

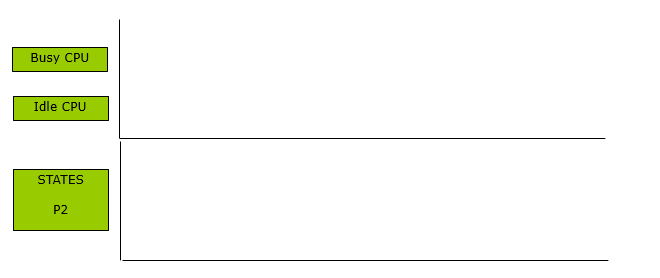
**Round Robin**

**The CPU and I/O times for 2 processes are shown below. Assume that PI gets to the ready queue just before P2 and the scheduling algorithm used by the OS is Round Robin with a time slice of 3 time units. Assume that the I/Os for the processes are different so that there is no I/O queue. Assume also that an interrupt from a completed I/O for process "X" will place process "X" in the ready queue BEHIND the process that was just interrupted.**

Using the first empty graph, describe how the CPU will be assigned to each process and for how long.

Use the second empty graph to show ALL the states that P1 goes through and the amount of time it has remained in that state until it has halted.

****

**P1**

P1 P2 P1 P1 P2 P2 P1 P1 P2

2 1 1 2 2 3 3 1 2

Problem 1. Draw the wait-for graph for the following situation:

P1 is using R1 and waiting for R2

P2 is using R2 and waiting for R4

P3 is using R5 and R3 and is waiting for R1

P4 is using R4 and is waiting for R5

1. Is there deadlock in this system?
2. If the answer is YES , then which processes are deadlocked

Problem 2. A system has: 5 processes *P*0 through *P*4; and 3 resource types: *A* (9 instances), *B* (5 instances), and *C* (7 instances). At a time T0 the state of the system is as shown below

***Allocation Max Available Need***

***A B C A B C A B C A B C***

***P*0 1 1 0 6 5 3 2 3 1 5 4 3**

***P*1 1 0 2 3 2 2 2 2 0**

***P*2 2 0 1 80 2 6 0 1**

***P*3 2 1 1 3 2 2 1 1 1**

***P*4 1 0 2 4 2 3 3 2 1**

1. Is the system is in a safe state?

**P1 P3 P4 P0 P2**

**P3 P1 P4 P0 P2**

**P3 P1 P4 P2 P0**

**………………..**

1. Can request for (2,2,1) by *P*4 be granted? If YES find the safe state. If NO explain why NOT

***Allocation Max Available Need***

***A B C A B C A B C***

***P*0 1 1 0 6 5 3 0 1 0**

***P*1 1 0 2 3 2 2**

***P*2 2 0 1 80 2**

***P*3 2 1 1 3 2 2**

***P*4 3 2 3 4 2 3**

1. Can request for (0,2,0) by *P*0 be granted? If YES find the safe state. If NO explain why NOT

***Allocation Max Available Need***

***A B C A B C A B C ABC***

***P*0 1 3 0 6 5 3 2 1 1**

***P*1 1 0 2 3 2 2**

***P*2 2 0 1 80 2**

***P*3 2 1 1 3 2 2**

***P*4 1 0 2 4 2 3**

1. Can request for (1,0,1) by *P*2 be granted? If YES find the safe. If NO explain why NOT

***Allocation Max Available Need***

***A B C A B C A B C A B C***

***P*0 1 1 0 6 5 3 1 3 0**

***P*1 1 0 2 3 2 2**

***P*2 3 0 2 80 2**

***P*3 2 1 1 3 2 2**

***P*4 1 0 2 4 2 3**

Problem 1. A producer and a consumer threads are sharing a buffer as shown below. The buffer is initialized with “blank” characters in each position. Semaphores X, Y, and Z are used to protect the buffer and are initializing properly. After the threads have been running for a while, the buffer appears as shown below and the threads are executing at the location shown by the arrows below. If the last character placed in the buffer was ‘8’ and the value of semaphore Y is 5, answer the question below.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **2** | **a** | **8** | **d** | **f** | **1** | **9** | **s** | **m** | **c** | **w** | **7** | **b** |

Producer

Repeat

do other calculation

P(Y)

P(Z)

Place item in buffer

V(Z)

Do other calculation

V(X)

Until false

Consumer

Repeat

do other calculation

P(X)

P(Z)

Read item from buffer

V(Z)

Do other calculation

V(Y)

Until false

What was the value of semaphore X when it was initialized? \_\_\_\_\_\_\_0\_\_\_\_\_\_\_\_\_

What is the value of semaphore X now? \_\_\_\_\_\_\_\_\_\_7\_\_\_\_\_\_\_

What is the value of the consumer buffer index now? \_\_\_\_\_\_\_\_9\_\_\_\_\_\_\_\_\_

 What is the value of the producer buffer index now? \_\_\_\_\_\_\_\_\_3\_\_\_\_\_\_\_\_

Which was the last character read by the consumer? \_\_\_\_\_\_\_\_\_m\_\_\_\_\_\_\_\_

Which specific character that are still in the buffer HAVE BEEN READ by the consumer

\_\_\_\_\_\_\_\_\_\_\_df19sm\_\_\_\_\_\_\_\_\_\_\_\_\_

**Note. Initialization: X (full)= 0, Y(empty)=13, Z(mutex) = 1**

**Problem 2.** Suppose that processes P1, P2, and P3 shown below running concurrently. S1 and S2 are among the statements that P1 will eventually execute, S3 and S4 are among the statements that P2 will eventually execute and S5 and S6 are among the statements that P3 will eventually execute.

You need to use semaphores to guarantee that statement S3 is executed AFTER statement S6 has been executed – also that statement S1 will be executed BEFORE statement S4 – also that statement S5 is executed BEFORE statement S2.

Show, within the structure of the processes below, how you would use semaphores to coordinate these three processes. Include semaphore names and to which values you would initialize the semaphores.

1. Which semaphores are you using and their initial values: Sem1=1, Sem2=1, Sem3=1

Process P1 Process P2 Process P3

Wait(sem1)

S1 S3 S5

Signal(sem2) Signal(sem3)

Wait(sem3) Wait(sem2)

S2 S4 S6

Signal(sem1)

1. After the semaphores have been appropriately placed, is it possible to determine which of the 6 statements above will be executed first?

If YES – which one? \_\_\_\_No\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. And which of the 6 statements above will be executed last?

If YES – which one \_\_\_\_\_\_No\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Problem 3.** Assume that the following section of main memory is used to store the page table for 3 different processes. The page-table base register values for process P1 is 1080, for P2 is 1085, and for P3 is 1090. Assume that the contests of memory below correspond to frame numbers. Also assume that frame size is 8192.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **contents** | **3** | **5** | **8** | **4** | **7** | **2** | **0** | **15** | **11** | **18** | **6** | **20** | **24** | **9** | **10** | **13** | **30** | **38** | **40** | **1** |
| **Memory addresses** | **1**  **0**  **7**  **6** | **1**  **0**  **7**  **7** | **1**  **0**  **7**  **8** | **1**  **0**  **7**  **9** | **1**  **0**  **8**  **0** | **1**  **0**  **8**  **1** | **1**  **0**  **8**  **2** | **1**  **0**  **8**  **3** | **1**  **0**  **8**  **4** | **1**  **0**  **8**  **5** | **1**  **0**  **8**  **6** | **1**  **0**  **8**  **7** | **1**  **0**  **8**  **8** | **1**  **0**  **8**  **9** | **1**  **0**  **9**  **0** | **1**  **0**  **9**  **1** | **1**  **0**  **9**  **2** | **1**  **0**  **9**  **3** | **1**  **0**  **9**  **4** | **1**  **0**  **9**  **5** |

To which physical memory address would the logical address (1, 1200) correspond to if generated by P1?

Answer:\_\_ **17584** \_\_\_\_\_\_\_\_\_\_\_\_\_

To which physical memory address would the logical address (3, 800) correspond to if generated by P2?

Answer:\_\_ **197408\_**\_\_\_\_\_\_\_\_\_\_\_\_

Assume that each process has 5 pages. To which process does the following physical address belong and which logical address corresponds to each physical address:

Physical address 57344 Process \_\_ **P1\_**\_\_\_\_\_\_Logical address \_\_\_\_**(0,0)\_**\_\_\_\_\_\_\_\_\_

Physical address 57343 Process \_\_\_\_\_ **P2**\_\_\_\_\_\_\_Logical address \_\_\_**(1,8191)**\_\_\_\_

**Problem 4.** A paging system is experiencing a page fault rate of 1 in 1 million page references. When a page fault occurs, 30% of the time an empty frame is not available and frame replacement is needed. Assume that it takes 10 milliseconds to service a page fault needing no frame replacement. When frame replacement is needed, 40% of time the frame has been modified and it takes 20 milliseconds to service the page fault, in contrast it takes 12 milliseconds when the frame to be replaced has not been modified. It takes 100 nanoseconds to reference a physical memory location and we can neglect the time that it takes to access the page table.

Calculate the effective memory access time for this system under these conditions. Clearly indicate intermediate steps.

Answer: 111.56 nanoseconds

**Problem 5.** Consider the following page reference string:

1. 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 6

How many page faults would occur for the following replacement algorithms, assuming three and four frames? Remember that all frames are initially empty, so your first unique pages will cost one fault each.

* FIFO replacement
* Optimal replacement
* LRU replacement

Answer:

FIFO: 13, 12

Optimal: 9, 7

LRU: 13, 9

**Problem 6.** An OS designer wants to have a paged memory management system with no more than 15% increase in the memory access time. Assume that all pages are in the main memory and the paging system has a translation look-aside buffer (TLB) and a page table in main memory. Given the main memory cycle time is 1200ns and the cache access time is 100 ns what does the hit ratio need to be?

Solution: EAT = 1200\* 0.15+1200 = 1380

1380 = p\* (100 +1200) +(1-p) (2\*1200+100)

1380 =p\*1300 + (1-p)(2500)

1380 = 1300p-2500p+2500

1120 =1200p

p = 1120/1200= 93.33333%

**Problem 7.**

Assume that in a paging system a process is given 2 frames. The frames are 200 words long. Assume that all of instruction for the process fit perfectly in the first frame. A 300 by 50 two-dimensional array, A, is declared by the process and the second frame is to be used when an array element is referenced. The elements of the array fit perfectly in the appropriate number of logical pages and no other data or instructions are included in these pages – only array elements. The array is stored in ROW MAJOR. Assume that it takes a word to store a value in each array position.

Consider the following loop that manipulates the array elements:

for j= 0 to 49 do

for i = 0 to 299 do

A[i][j] =100

How many total number of page faults will be generated by the statement A[i][j] = 0 in the loop above?

**Answer: \_\_50\*75=3750\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

How many total number of page faults will be generated by the statement A[i][j] = 0 in the loop above?

for i= 0 to 299 do

for j = 0 to 49 do

A[i][j] = 100

**Answer: \_\_\_75\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Problem 8.**  Clearly explain the relationship between the concept of “working set” and the concept “thrashing” in virtual memory system.

Answer:

In order for process to execute with a minimal number of page faults it must have its entire working set in main memory. A working set is a group of pages that actively referencing. As more and more processes are brought into main memory, fewer frames will be available per process. If the number for frames falls below the number of pages in the working set of a process then the number of page faults will increase. If many processes will be in this situation then the trashing will occur.

**Question 9.** Q1. Consider the following page reference string:

1 4 7 5 3 4 7 5 6 7 8 3 5 6 1

How many page faults would occur for the following replacement algorithms, assuming a) three frames, b) four frames?

All frames are initially empty, so your first unique pages will all cost one fault each.

Optimal replacement

1 4 7 5 3 4 7 5 6 7 8 3 5 6 1

LRU replacement

1 4 7 5 3 4 7 5 6 7 8 3 5 6 1

**Problem 10.** Assume that the outermost track of a disk is numbered 0 and the innermost track is numbered 600. Assume that the disk head has just read a sector on track 40 and that the disk I/O queue contains the following track requests: **50, 75, 20, 150, 85, 300.**

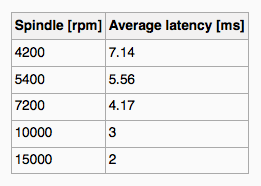
1. In what sequence will these requests be serviced if C-Scan scheduling is used. Also calculate the total distance in tracks traveled by the disk head in satisfying these requests.

**Answer: 40-50-75-85-150-300-599-20**

**599-40+20 =579**

1. In what sequence will these requests be serviced if Look scheduling is used. Assume that the head is traveling in the direction of the outermost track. Also calculate the total distance in tracks traveled by the disk head in satisfying these requests.

1. In what sequence will these requests be serviced if S-S-T-F disk scheduling is used. Also calculate the total distance in tracks traveled by the disk head in satisfying these requests.
2. What is average I/O time to transfer a 8KB block on a 7200 RPM disk with 10 ms average seek time, 2Gb/sec transfer rate with a 0.2 ms controller overhead.



Average I/O time = average access time + (amount to transfer / transfer rate) + controller overhead

**Average access time** = average seek time + average latency

**Problem 11.** What are the advantages and disadvantages of the various network topologies in terms of the following attributes?

1. **Installation cost**
2. **Communication cost and**
3. **Availability**

**Answer:**

**Problem 12.** What are the three most common routing strategies? What are advantages and disadvantages for each of them?

Answer:

**Problem 13.** What are four reasons for building distributed systems?

Answer: