

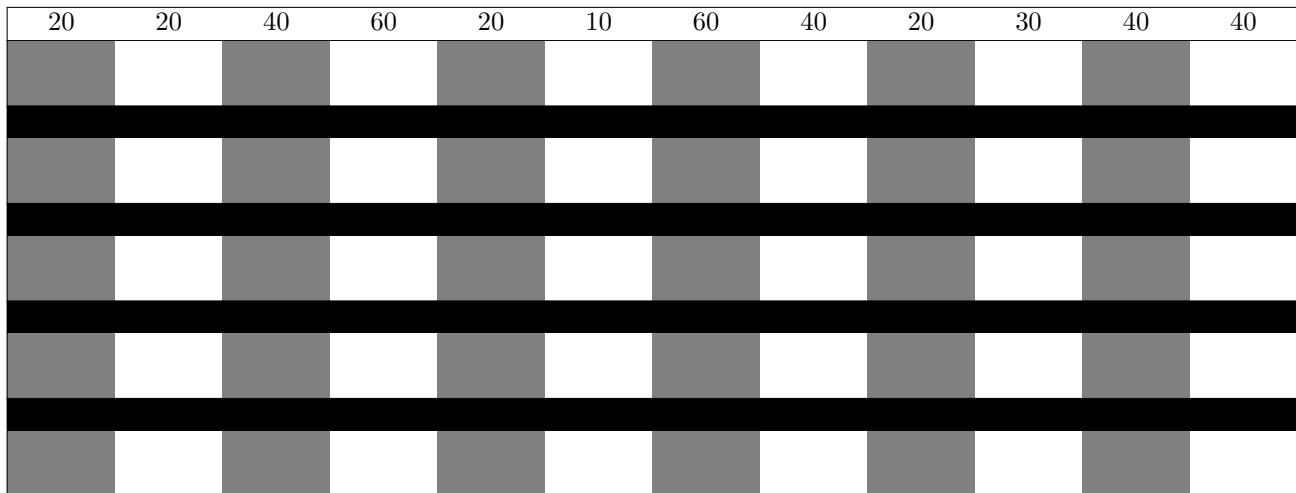
# Exam Review

## 1 Memory Partitioning

The following diagram shows a partition of memory. If three separate requests for memory of 40M, 20M, and 10M are made, determine the spots in memory where the requests will each be placed, and indicate the beginning address of the request based on the following policies:

1. First Fit
2. Best Fit
3. Next Fit
4. Worst Fit

The grey blocks represent already filled spaces in memory.



# 2 Memory Allocation

A 1Mb block of memory is allocated using the buddy method. Show the resulting partitions in the following diagram if they recieve the following memory requests:

$$A = 70K \qquad B = 35K \qquad C = 80K \qquad D = 60K$$

The memory is requested/released as follows:

1. Request A  
2. Request B  
3. Request C  
4. Return A
5. Request D  
6. Return B  
7. Return D  
8. Return C

1024						Initial	
A		128		256		512	Request A
A		B	64	256		512	Request B
A		B	64	C	128	512	Request C
128		B	64	C	128	512	Return A
D	64	B	64	C	128	512	Request D
D	64	128		C	128	512	Return B
256				C	128	512	Return D
1024							Return C

### 3 Paging I

Given the following table, determine which physical address, if any, the following virtual address would correspond.

1103

2325

5544

VPN	Valid Bit	Reference Bit	Modify Bit	PFN
0	1	1	0	4
1	1	1	1	7
2	0	0	0	-
3	1	0	0	2
4	0	0	0	-
5	1	0	1	0

$$1103 = (1 * 1024) + 79$$

$$\implies \text{PFN: } 7$$

$$\begin{aligned}\text{Address} &= (7 * 1024) + 79 \\ &= 7247\end{aligned}$$

$$2325 = (2 * 1024) + 277$$

$$\implies \text{PFN: } -$$

$$\implies \text{PAGE FAULT}$$

$$5544 = (5 * 1024) + 424$$

$$\implies \text{PFN: } 0$$

$$\begin{aligned}\text{Address} &= (0 * 1024) + 424 \\ &= 79\end{aligned}$$

## 4 Paging II

Given the following table, assume there is a page fault at 4. Determine which page will be replaced using the following policies:

1. FIFO
2. LRU
3. Clock
4. Optimal (Assume the remaning sequence is 4, 0, 0, 0, 2, 4, 2, 1, 0, 3, 2)

VPN	Time Loaded	Reference Time	R-Bit	M-Bit
2	60	161	0	1
1	130	160	1	0
0	26	162	1	0
3	20	163	1	1

VPN	3	0	2	1	...	1	2	0	3	4
VPN In Memory	3	3	3	3	...	3	3	3	3	4
		0	0	0		0	0	0	0	0
			2	2		2	2	2	2	2
				1		1	1	1	1	1
Time	20	26	60	130	...	160	161	162	163	164
VPN	3	0	2	1	...	1	2	0	3	4
VPN In Memory	3	3	3	3	...	3	3	3	3	3
		0	0	0		0	0	0	0	0
			2	2		2	2	2	2	2
				1		1	1	1	1	4
Time	20	26	60	130	...	160	161	162	163	164
VPN	3	0	2	1	...	1	2	0	3	4
VPN In Memory	3	3	3	3	...	3	3	3	3	3
		0	0	0		0	0	0	0	0
			2	2		2	2	2	2	4*
				1		1	1	1	1	1*
Time	20	26	60	130	...	160	161	162	163	164
VPN	1	2	0	3	4	0 0 0 2 4 2 1 0 3 2 ...				
VPN In Memory	3	3	3	3	4					
	0	0	0	0	0					
	2	2	2	2	2					
	1	1	1	1	1					
Time	160	161	162	163	164	...				

## 5 Processor Scheduling

Determine the processor scheduling times of the following policies.

1. First Come First Serve
2. Round Robin ( $q = 1$ )
3. Round Robin ( $q = 4$ )
4. Shortest Process Next
5. Shortest Remaining Time
6. Highest Response Ratio Next (Response Ratio =  $\frac{\text{wait time} + \text{service time}}{\text{service time}}$ )
7. Feedback ( $q = 1$ )
8. Feedback ( $q = 2^i$ )

Process Name	Arrival Time	Service Time
1	0	3
2	1	5
3	3	2
4	9	5
5	12	5

Process Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	█	█	█																	
2				█	█	█	█	█												
3									█	█										
4											█	█	█	█	█					
5																█	█	█	█	█
1	█		█			█														
2		█		█			█		█		█									
3					█			█						█		█		█		
4										█		█		█		█		█		
5													█		█		█		█	█
1	█	█	█																	
2				█	█	█	█	█		█										
3								█	█											
4											█	█	█	█					█	
5															█	█	█	█		█
1	█	█	█																	
2						█	█	█	█	█										
3				█	█															
4											█	█	█	█	█					
5																█	█	█	█	█
1	█	█	█																	
2						█	█	█	█	█										
3				█	█															
4											█	█	█	█	█					
5																█	█	█	█	█

Process Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1																				
2																				
3																				
4																				
5																				
1																				
2																				
3																				
4																				
5																				
1																				
2																				
3																				
4																				
5																				
1																				
2																				
3																				
4																				
5																				

# 6 Multiprocessor Scheduling I

Determine the runtimes of the following processes based on the Fixed Priority ( $A \rightarrow B \rightarrow C$ ), and Earliest Deadline policies.

Process	Arrival Time	Execution Time	Ending Deadline
$A_1$	0	10	20
$A_2$	20	10	40
$A_3$	40	10	60
$A_4$	60	10	80
$A_5$	80	10	100
$B_1$	0	10	50
$B_2$	50	10	100
$C_1$	0	15	50
$C_2$	50	15	100

		10		20		30		40		50		60		70		80		90		100
FP	$A_1$	$A_1$	$B_1$	$B_1$	$A_2$	$A_2$	$C_1$	$C_1$	$A_3$	$A_3$	$B_2$	$B_2$	$A_4$	$A_4$	$C_2$	$C_2$	$A_5$	$A_5$	$C_2$	
ED	$A_1$	$A_1$	$B_1$	$B_1$	$A_2$	$A_2$	$C_1$	$C_1$	$C_1$	$A_3$	$A_3$	$B_2$	$B_2$	$A_4$	$A_4$	$C_2$	$C_2$	$C_2$	$A_4$	$A_4$

# 7 Multiprocessor Scheduling II

Display the result of the following periodic tasks under the Earliest Starting Deadline, First Come First Serve, and Earliest Starting Deadline with Unenforced Idle Times policies.

Process	Arrival Time	Execution Time	Starting Deadline
A	10	20	100
B	20	20	20
C	40	20	60
D	50	20	80
E	60	20	70

	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100	105	110	115	120
ESD		A	A	A	A		C	C	C	C	E	E	E	E	D	D	D	D						
ESDU					B	B	B	B	C	C	C	C	E	E	E	E	D	D	D	D	A	A	A	A
FCFS			A	A	A	A		C	C	C	C	D	D	D	D									



## 8 I/O Operations

(a) Perform a FIFO, SSTF, SCAN, and C-SCAN on the following sequence of disk track requests in order to calculate the average seek length:

27, 129, 110, 186, 147, 41, 10, 64, 120

Assume that the disk starts at address 100, and the head is moving in the direction of decreasing track number.

FIFO:		SSTF:		SCAN:		C-SCAN:	
	100		100		100		100
27	73	110	10	64	36	64	36
129	102	120	10	41	23	41	23
110	19	129	9	27	14	27	14
186	76	147	18	10	17	10	17
147	39	186	39	110	100	186	176
41	106	64	122	120	10	147	39
10	31	41	23	129	9	129	18
64	54	27	14	147	18	120	9
120	56	10	17	186	39	110	10
Average:	61.8	Average:	29.1	Average:	29.6	Average:	38

(b) Perform a FIFO, SSTF, SCAN, and C-SCAN on the following sequence of disk track requests in order to calculate the average seek length:

55, 58, 39, 18, 90, 160, 150, 30, 184

Assume that the disk starts at address 100, and the head is moving in the direction of increasing track number.

FIFO:		SSTF:		SCAN:		C-SCAN:	
	100		100		100		100
55	45	90	10	150	50	150	50
58	3	58	32	160	10	160	10
39	19	55	3	184	24	184	24
18	21	39	16	90	94	18	166
90	72	30	9	58	32	30	12
160	70	18	12	55	3	39	9
150	10	150	132	39	16	55	16
30	120	160	10	30	9	58	3
184	154	184	24	18	12	90	32
Average:	58.1	Average:	27.5	Average:	27.7	Average:	35.7

## 9 UNIX Inodes

Consider a UNIX file system represented by inodes. Assume that there are 12 direct block pointers and a singly, doubly, and triply indirect pointer in each inode. Further, assume that the system block size is and the disk sector size are both 8 Kb. If the disk block pointer is 32 bits, with 8 bits to identify the physical disk and 24 bits to identify the physical block, then

1. What is the max file size supported by the system?

Since there are 12 block pointers in the inode, and each block is 8 Kb, we have  $12 * 8$  Kb, or 96 Kb, directly addressable. The indirect pointer points to an entire block of pointers. Since each block is 8 Kb, and each pointer is 32 bits, there are 1024 pointers in the block; each pointer points to an 8 Kb block. This means the single pointers take up  $1024 * 8$  Kb, or 8192 Kb. The double pointer points to a pointer that points to a block. There are 1024 of those, so we have  $1024 * 1024 * 8$  Kb, or 8,388,608 Kb. The triple pointer will yield  $1024 * 1024 * 1024 * 8$  Kb, or 8,589,934,592 Kb. This yields a total size of 8,598,331,488 Kb.

2. What is the max file system partition supported by the system?

There are 24 bits assigned to identify the physical block, so the max file system partition supported is  $2^{24} * 8$ , or 134,217,728 Kb.

3. How many disk accesses are required to access the byte in position 13,423,956?

The location in question is located in the third pointer, so it would take 4 disk accesses to access the location.