1. Consider the following workload:

Process	Arrival Time	Processing Time
P1	0	12
P2	2	6
P3	8	18
P4	10	4
P5	12	4

a. Show the execution pattern for the above request using all the six different scheduling policies:

- FCFS : 111111111111222222333333333333333333344445555
- Round Robin (q = 2): 112211223344551122334455113311331133333333333
- Round Robin (q = 4)111122223333444455551111223333111133333333333
- SPN: 111111111111444455552222223333333333333333333
- SRT: 112222221144445555111111113333333333333333333
- HRRN: 111111111111222222444455553333333333333333333
- Feedback (q = 1): 112212123344553453451231231313131313333333333
- Feedback (q = 2^i): 111222111133345445522233334511111133333333333

Note: Use the example on slides # 21 as a reference for the solution (Ch 09 - Scheduling).

b. Recreate the table, similar to the one on slide # 22 in Ch 09 - Scheduling lesson, for comparing all the schedluing poiclies used. NOTE: check the bottom of the page for the screenshot of the excel sheet

2. Consider a memory-management system based on paging. Given the process has the following page table, answer the questions that follow.

Page#	Frame#
0	000101
1	001110
2	010101
3	011100

(i) Determine the physical address layout if the logical address for an instruction is - `0000010111011110`.

first 2 bits are for Page #, in this case is page # 0, other 14 bits are for offset (00010111011110 or 1502 in decimal).
frame # 000101 is in Page 0, which is 5 in decimal.
we are at frame number 5, each frame # is 6 bits so there are 64 frames in memory, each frame is 14 bits or has 16384 memory addresses

5 * 16384 = 81920 + the offset of 1502 = 83422, which is the binary memory address of 10100010111011110

(ii) Determine the physical address layout if the logical address for an instruction is - `0000110101100011`.

Frame # 5, offset is 00110101100011 (3427). 5 * 16384 + 3427 = 85347 which is the physical memory address 10100110101100011

3. Consider a memory-management system based on segmentation. Given the process has the following segment table, answer the questions that follow. The segment # in the logical address is the first 4 bits.

Segment#	Length	Base
0	000101000100	0000011010001111
1	000001100001	0100000000000001
2	000110001100	0000101010001001

(i) Determine the physical address layout if the logical address for an instruction is - 0000000001011110.

segment # is 0, so the offset is 000001011110 (94). Base is at 0000011010001111 (1679).
1679 + 94 = 1773, which is address 11011101101

(ii) Determine the physical address layout if the logical address for an instruction is - 0010000000011011.

segment # is 2 since 0010 is the first 4 bits, offset is 000000011011 (27). the base is 0000101010001001 (2697)
2697 + 27 = 2724 or address 101010100100

4.Consider a simple segmentation system that has the following segment table:

Starting address	Length bytes
64	128
250	1024
1,508	408
2000	500

For each of the following logical addresses, determine the physical address **and** indicate if a segment fault occurs. **Note:** The physical address can be given in decimal format, and need not be converted in binary!

Logical address	Physical Address	Seg Fault?
0, 12	76	No
2, 648	2156	Yes
3, 776	2776	Yes
1, 98	348	No
3, 240	2240	No

5. Consider the following page-reference string: a, b, d, c, b, e, d, b, d, b, a, c, b, c, a, c, f,a, f, d. Assume that there are 3 frames available and that they are all initially empty. Show the resulting page replacement policies for all the 4 different algorithms learned in class, indicate the total number of page faults in each policy. **For the CLOCK policy, indicate the pointer and * clearly in the solution.**

NOTE: check the bottom of the page for the screenshot of the excel sheet

	a	b	d	c	b	e	d	b	d	a	c	b	c	a	c	f	a	f	d
OPT																			

	a	b	d	c	b	e	d	b	d	a	c	b	c	a	c	f	a	f	d
LRU																			

	a	b	d	c	b	e	d	b	d	a	c	b	c	a	c	f	a	f	d
FIFO																			

	a	b	d	c	b	e	d	b	d	a	c	b	c	a	c	f	a	f	d
CLOCK																			

6. What is fragmentation and what causes fragmentation? What are the types of fragmentation?

Essentially all fragmentation is is wasted space in memory. An example is if we have memory locations 0-10, and we have one program from memory locations 0-3 and another program from 5-10, there is fragmentation since memory location 4 is unused. Internal fragmentation is within frames or the memory block itself, which is found when memory management is based on paging. A cause of internal fragmentation is you bring over a block larger than what's needed/ what will be used, which is unavoidable since blocks are fixed-sized. External fragmentation is outside of the blocks, which is caused by segmentation systems. Memory between blocks become unused since segments vary in size and, for example, if memory for a small program that takes up not a lot of space is deallocated from memory, it would be difficult to make use of that memory.

7. Page faults can be reduced by using a TLB. What is a TLB and how can it help in reducing page faults?

TLB is Transitional Lookaside Buffer, which is basically a cache used to store the recently used virtual to physical address translation, which uses the idea of temporal locality (memory just accessed may be used again soon). this reduces page faults since it causes less lookups, so if I just recently replaced "A" on the page table but "A" was also recently accessed, there is no page fault since there was no need to access the page table if "A" is still cached in TLB.

for question 1b						
Process	1	2	3	4	5	
Arrival Time	0	2	8	10	12	
Service Time	12	6	18	4	4	mean
			FCFS			
Finish Time	12	18	36	40	44	
Turnaround	12	16	28	30	32	23.6
Tr/Ts	1	2.6666667	1.5555556	7.5	8	4.1444444
			RR (q = 2)			
Finish Time	34	18	44	22	24	
Turnaround	34	16	36	12	12	22
Tr/Ts	2.8333333	2.6666667	2	3	3	2.7
			RR (q = 4)			
Finish Time	34	26	44	16	20	
Turnaround	34	24	36	6	8	21.6
Tr/Ts	2.8333333	4	2	1.5	2	2.4666667
			SPN			
Finish Time	12	26	44	16	20	
Turnaround	12	24	36	6	8	17.2
Tr/Ts	1	4	2	1.5	2	2.1
			SRT			
Finish Time	26	8	44	14	18	
Turnaround	14	2	26	10	14	13.2
Tr/Ts	1.1666667	0.3333333	1.4444444	2.5	3.5	1.7888889
			HRRN			
Finish Time	12	18	44	22	26	
Turnaround	0	12	26	18	22	15.6
Tr/Ts	0	2	1.4444444	4.5	5.5	2.6888889
			Feedback (q = 1)			
Finish Time	37	25	44	19	20	
Turnaround	25	19	26	15	16	20.2
Tr/Ts	2.0833333	3.1666667	1.4444444	3.75	4	2.8888889
			Feedback (q = 2^i)			
Finish Time	33	22	44	27	28	
Turnaround	21	16	26	23	24	22
Tr/Ts	1.75	2.6666667	1.4444444	5.75	6	3.5222222

Question 5																				
	a	b	d	c	b	e	d	b	d	a	c	b	c	a	c	f	a	f	d
	a	a	a	c	c	e	e	e	e	a	a	a	a	a	a	a	a	a	d
OPT		b	b	b	b	b	b	b	b	b	b	b	b	b	b	f	f	f	f
9 faults			d	d	d	d	d	d	d	d	c	c	c	c	c	c	c	c	c
	F	F	F	F		F				F	F					F			F
	a	b	d	c	b	e	d	b	d	a	c	b	c	a	c	f	a	f	d
	a	a	a	c	c	c	d	d	d	d	d	b	b	b	b	f	f	f	f
LRU		b	b	b	b	b	b	b	b	b	c	c	c	c	c	c	c	c	d
11 faults			d	d	d	e	e	e	e	a	a	a	a	a	a	a	a	a	a
	F	F	F	F		F	F			F	F	F				F			F
	a	b	d	c	b	e	d	b	d	a	c	b	c	a	c	f	a	f	d
	a	a	a	c	c	c	c	c	d	d	d	b	b	b	b	b	b	b	b
FIFO		b	b	b	b	e	e	e	e	a	a	a	a	a	a	f	f	f	f
12 faults			d	d	d	d	d	b	b	b	c	c	c	c	c	a	a	a	d
	F	F	F	F		F		F	F	F	F	F				F			F
	a	b	d	c	b	e	d	b	d	a	c	b	c	a	c	f	a	f	d
	a*	a*	a*	c*	c*	c*	c*	c*	d*	d	d	b*	b*	b*	b*	b	b	b	d*
CLOCK		b*	b*	b	b	e*	e*	e*	e	a*	a	a	a	a	a	f*	f	f	f
13 faults			d*	d	d	d	d	b*	b	b	c*	c	c	c	c	c	a*	a*	a
red is pointer	F	F	F	F		F		F	F	F	F	F				F	F		F