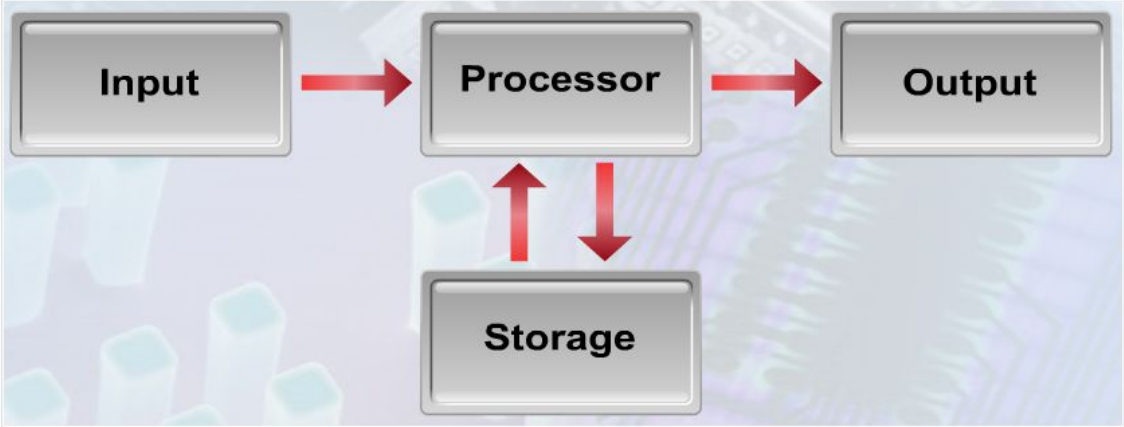
BACS1024 - RDS2(S1)G3 - Tutorial

Tutorial 1 - 16.6.2020

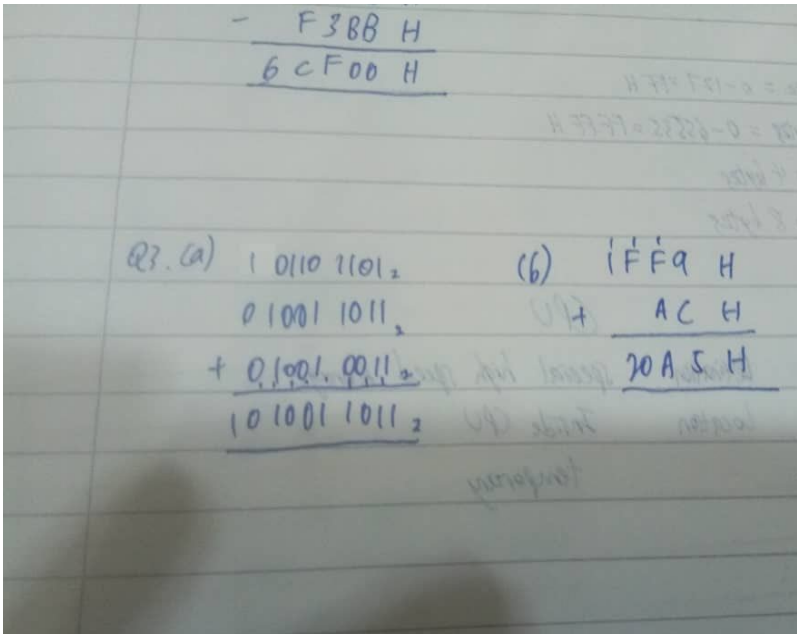
Q1	**i) Hardware** elements: Physical mechanisms that process data by executing instruction, storing and moving data. **Example**: Keyboard **ii)Software** elements:. Software is a set of instructions, data or programs used to operate computers and execute specific tasks. Software can be categorized as system software and application software **Example of system software**: Windows, Mac, Linux **Example of application software:** Microsoft Word, Excel, PowerPoint etc. **iii)Communication** elements: Hardware & Software that facilitate sharing, locally & remotely data accesses.Example: modem **iv) Data** elements: Fundamental representation of facts and observations. **Example**: user's data. User uses computer keyboard (hardware) to input password (data) to the computer system through Google Chrome (software) and send to request login to email account via modem (communication device).
Q2	**Volatile** Memory is used to store computer programs and data that CPU needs in real time but this data is not permanently stored which means it will be erased once the computer is switched off. And Volatile memory is also faster than non-volatile memory. Examples of RAM and Cache memory are volatile memory. While **non-volatile** memory is static which means that the data that is stored in the memory is permanent and remains in the computer even if the computer is switched off but it is slower than the volatile memory. Examples of non-volatile memory are HDD and Rom.
Q3	The user interacts directly with hardware for the human *input* and *output* such as displays, e.g. through a graphical user interface. The user interacts with the computer over this software interface using the given input and output (*I/O*) hardware. For example, the User will see the output of the computer which is the user interface from the monitor screen whereas the user will input the data that needs to be entered into the computer with the keyboard or selecting something in the computer by clicking the mouse button.
Q4	**System Software :** It is responsible for managing files, load and executing the programs on a computer. Low level languages are used.

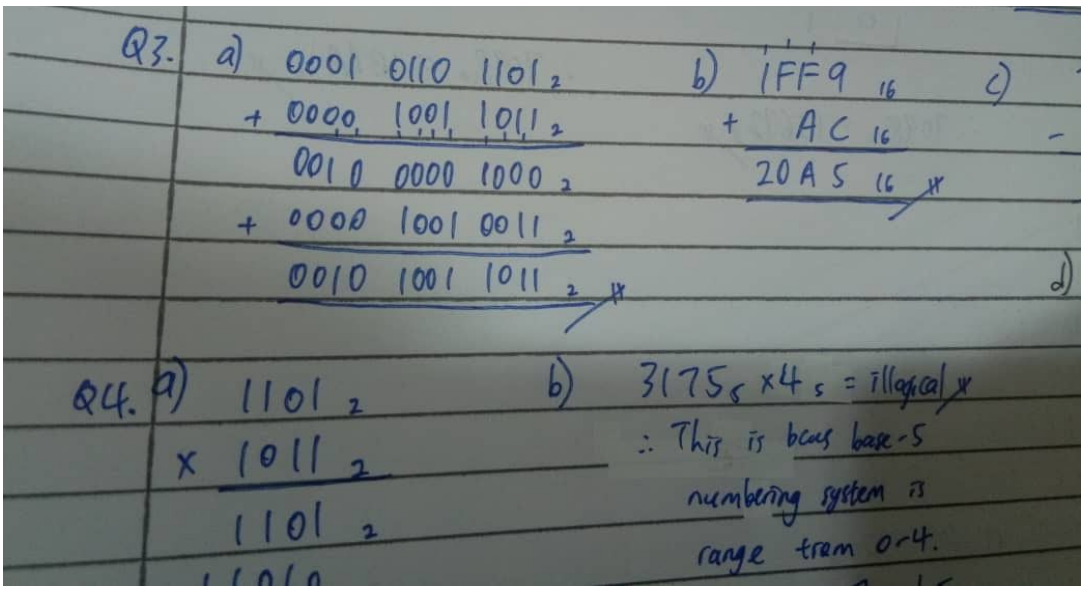
	If without system software, the system cannot run. For example, operating system like macOs, Linux and Microsoft Windows **Application Software :** It is responsible to perform functions, tasks, activities for the benefits of the users. High level languages are used. If without application software, the system still can run. For example, Word processing software, Spreadsheets Software and Photoshop. Both of them consist of the step by step instruction to tell the hardware what to do and how to do (a.k.a = also known as) program.
Q5	Agree. Because bus topology is inexpensive and easy to install. Nodes can be attached or detached from the bus without distributing the network. Besides, the bus transmits data in both directions.
Q6	 Information Processing Cycle (IPC) provides an important basic tool for system operation. The sequence of events in processing information, which includes input, processing, storage and output. These processes work together and repeat over and over. For example: Input = enter 5 , 10 (enter the data into the system) Example of the input device = keyboard Process = the system will add up two of the input number (addition) - (performing operation on the data) Output = and it will be display 15 which 5 + 10 - (presenting the results) Example of output device = monitor Storage = the system will store the data for future use

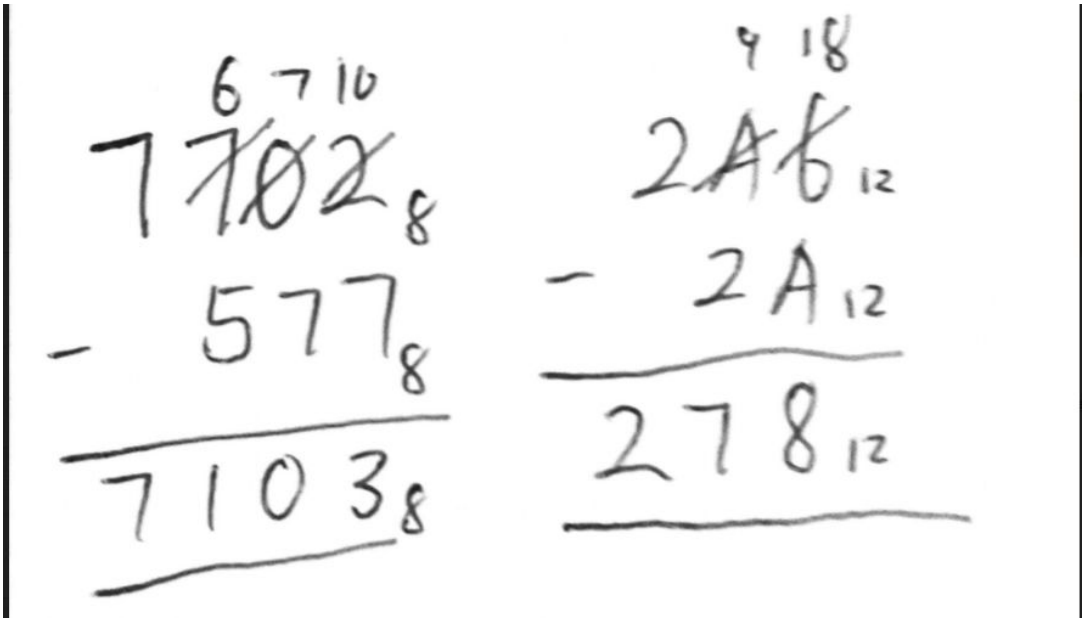
Q7	I would like to choose client/server as the most appropriate network architecture to implement a network system in my hostel with 8 hostel mates because number of the network clients are allowed to send files or data to server at the same time and they may request to the server who manage the data to access those resources at the same time within multiple users. Meanwhile, this will help them to increase their efficiency on sending packets of file by not waiting for others to queue instead of a peer-to-peer network need to wait for the status is idle and then the next person is allowed to share files toward each other. Recommended Peer to peer. Justification: So that every member could have equal processing power and not depend on others. The power off of one member's PC will not affect others. Every member could send / receive data without restriction

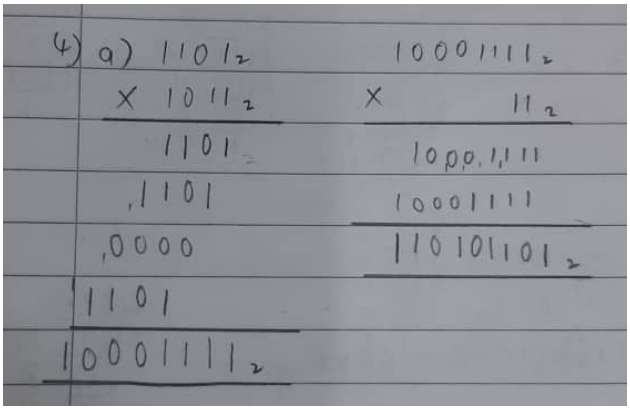
Tutorial 2 - 23.6.2020

Q1a&b	Lee Jun Xian **a) Convert $3D7_{16}$ to binary, octal and decimal respectively** 1 a) $3D7_{16}$ = 11 1101 0111$_2$ = 1727$_8$ = 983_{10} → 2+2+1m (missing working for Base 10) WOrking = 1m Final answer = 1m **b) Convert 1100010100100001_2 to octal, decimal and hexadecimal respectively** b) 1100 0101 0010 0001$_2$ = C521$_{16}$ = 142441$_8$ = 50465_{10} → 2+2+1m (missing working for Base 10)
Q1c&d	Leong Yit Wee c) **Convert 7098_{10} to binary, octal and hexadecimal respectively** 1c) 7098(10) = 1 1011 1011 1010 B

	 → 1m (missing working) = 15672(8) → 2m = 1BBA H → 2m **d) Convert 13612$_8$ to binary, decimal and hexadecimal respectively** 1d)13612(8) = 1 0111 1000 1010 B → 2m = 6026(10) → 1m (missing working) = 178A H → 2m
Q1e&Q 2a	Lim Chia Chung **1 e) Convert 210102$_3$ to decimal** $210102_3 = 578_{10}$ → 1m (missing working) **2 a) 1011$_2$ + 1111$_2$** $101\ 1_2$ $+\ 111_1 1_2$ $= (1)1010_2$ → 2m
Q2b&c	**715$_8$ – 57$_8$** → 2m

	2(b) $715_8 - 57_8 = 636_8$ ✱ $^{6}7\,1\,5_8^{13}$ $-\ 57_8$ 636_8 **C521H x 3DH** → 2m 2(c) C521H X 3DH = $2EF8DD_{16}$ ✱ $^{1\,4}\,^{1}C521$ 16|126 1-10 16|160 10-0 X 3D A02AD 16|66 4-2 16|36 2-4 24F63 2EF8DD
Q3a&b	LIM MING JUN **a.) 101101101_2 + 10011011_2 + 10010011_2** **b.) $1FF9_{16}$ + AC_{16}** 

Q3c&d	Lim Yih Feng **c.) $7702_8 - 577_8$** **d.) $2A6_{12} - 2A_{12}$**  7702 - 577 7103 2A6 - 2A 27 8
Q4	Ong T'nsam **a) $1101_2 \times 1011_2 \times 11_2$** **b) $3175_5 \times 4_5$** a) 110101101

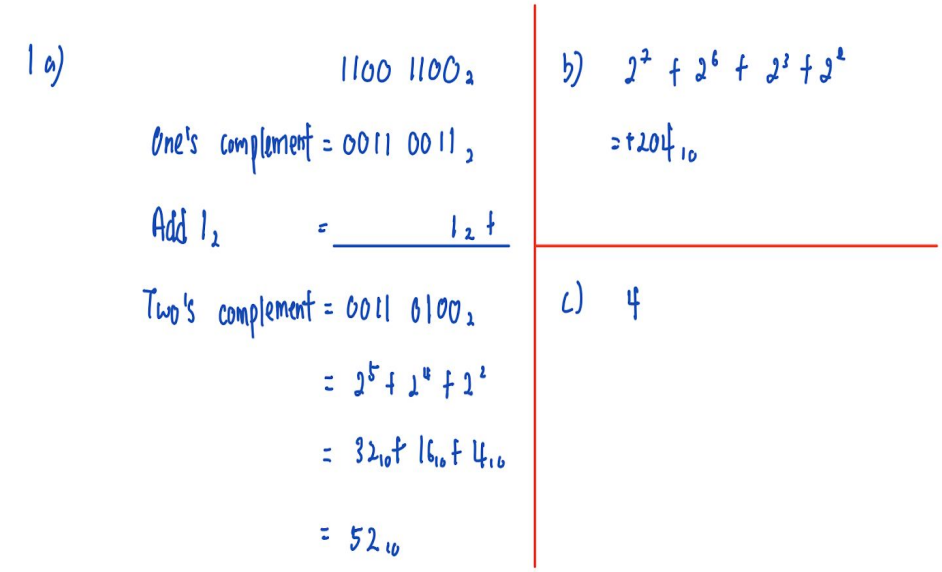


b) Illogical because the data range for base 5 is 0,1,2,3,4, so the number should not be more than or equal to 5.
-31**75**(number 7 more than 5; number 5 equals to 5)

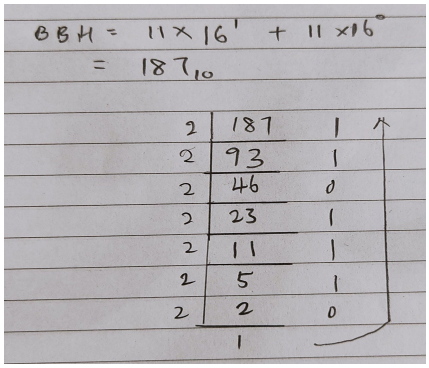
Tutorial 3 - To discuss at 30.6.2020

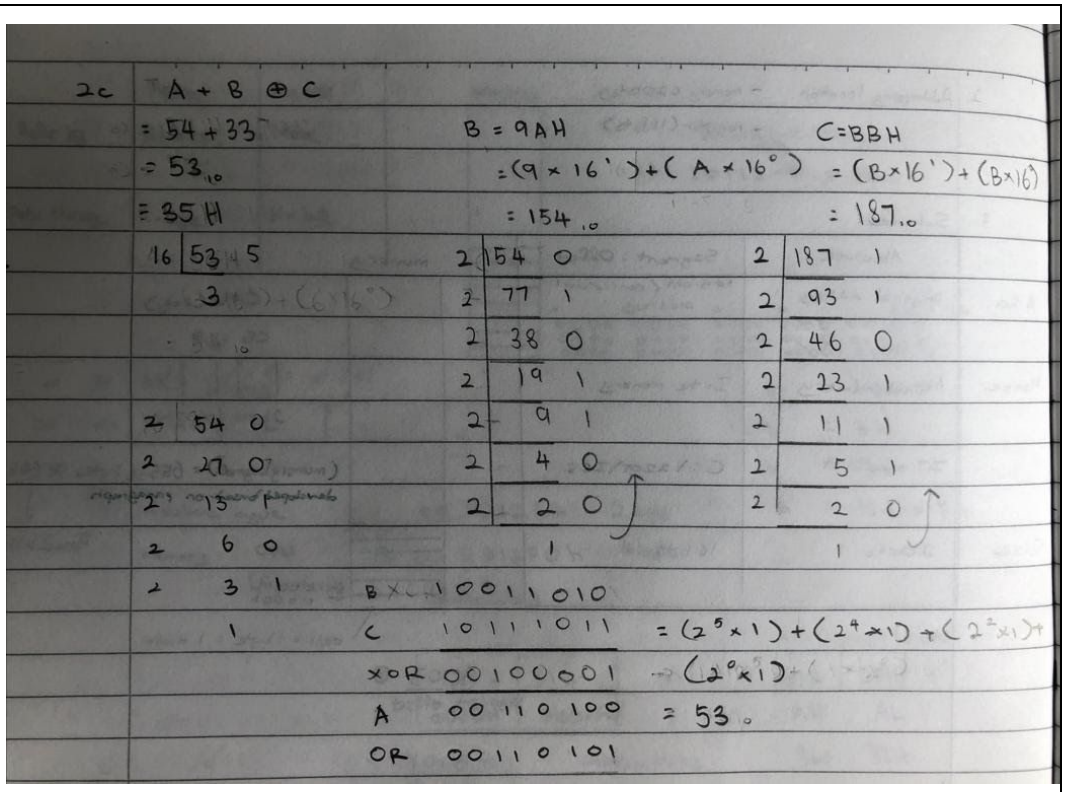
- = AND, + = OR, (+) = XOR, ‘ = NOT

1a



1100 1100b = -(2^7) +(2^6)+(2^3)+(2^2)
= -128 +64+8+4
= -52d (signed decimal value)
1100 1100b = (2^7) +(2^6)+(2^3)+(2^2)
= 128 +64+8+4
= +204d (unsigned decimal value)
ASCII = 1100 1100b
= C C h

	= CC H
2a	A = 36H , B = 9AH A to binary = 0011 0110_2 B to binary = 1001 1010_2 0011 0110_2 AND 1001 1010_2 0001 0010_2 A AND B = 0001 0010_2 = 12H
2b	B = 9A H , C = BB H convert C to binary, C = 1011 1011_2 C' = 0100 0100_2 convert C back to Hexadecimal C' = 44 H B + C' = 9A H + 44 H = DE H ←wrong B = 9AH = 1001 1010B C= BBH = 1011 1011B C' = 0100 0100B B+C'= B OR C'= 1001 1010B 0100 0100B OR 1101 1110B → DEH 
2c	( ) → NOT → XOR → AND → OR



CORRECT ANSWER AS BELOW

B = 9AH = 1001 1010B

C= BBH = 1011 1011B XOR

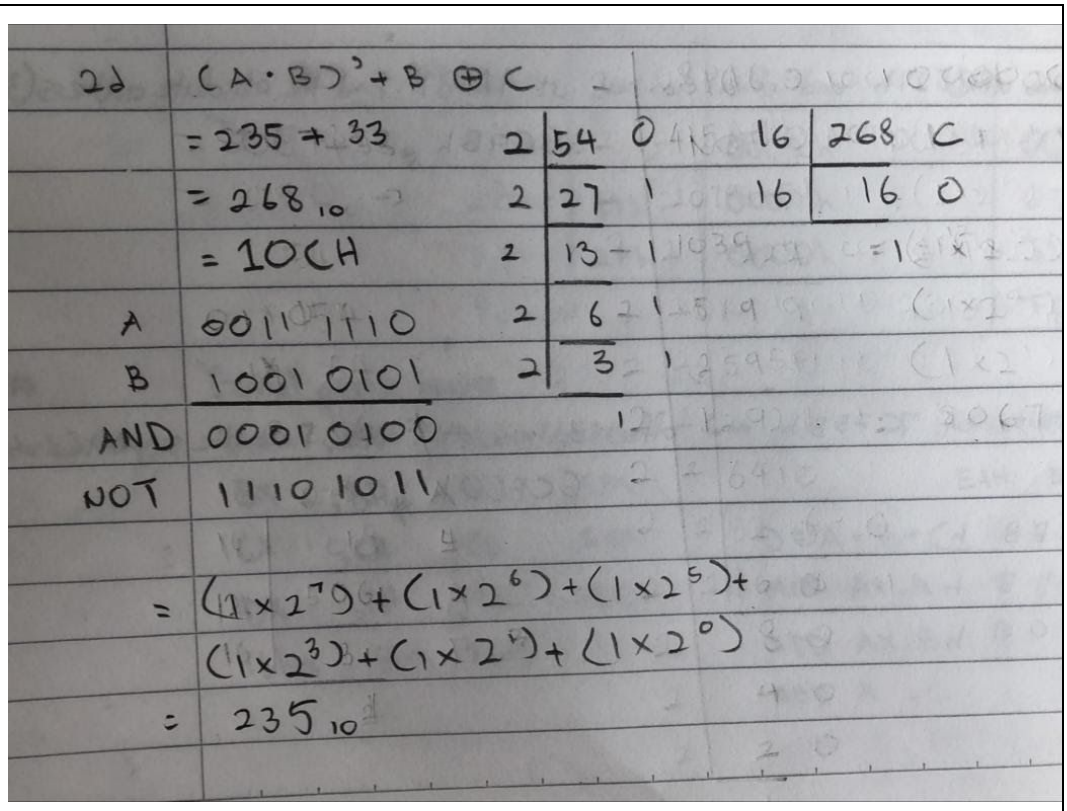
B XOR C = 0010 0001B

A = 0011 0110B OR

A OR **B XOR C=** 0011 0111B

= 3 7H

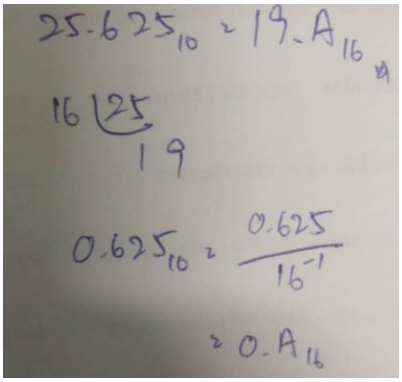
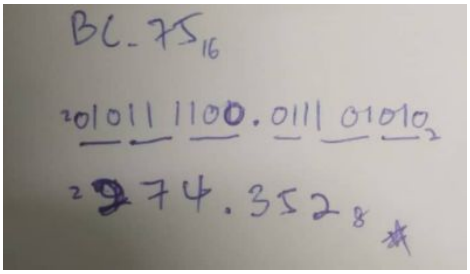
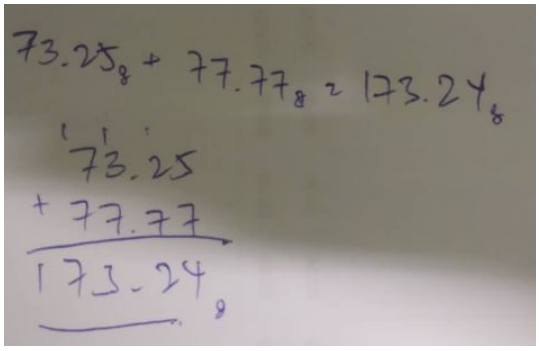
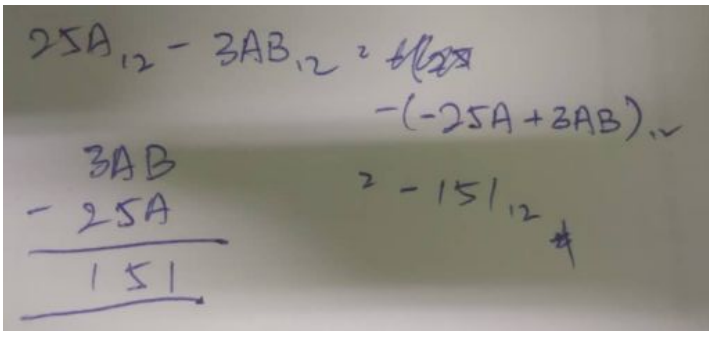
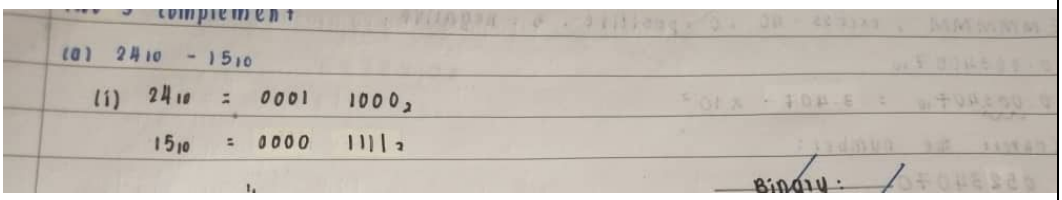
2d

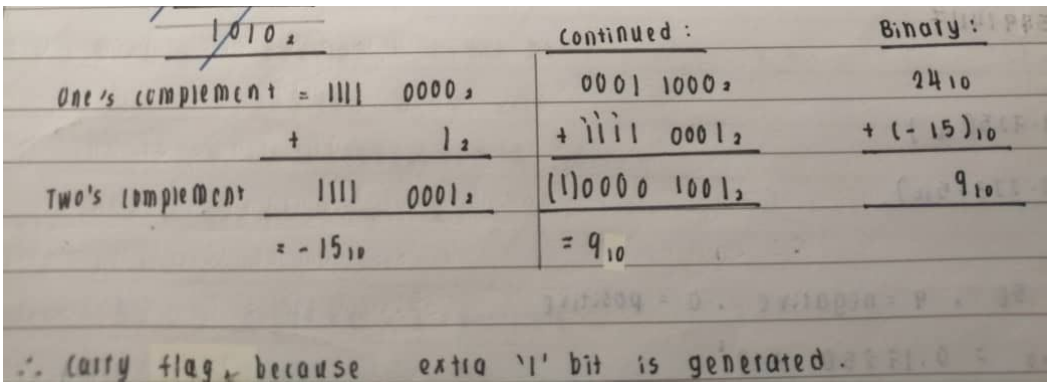


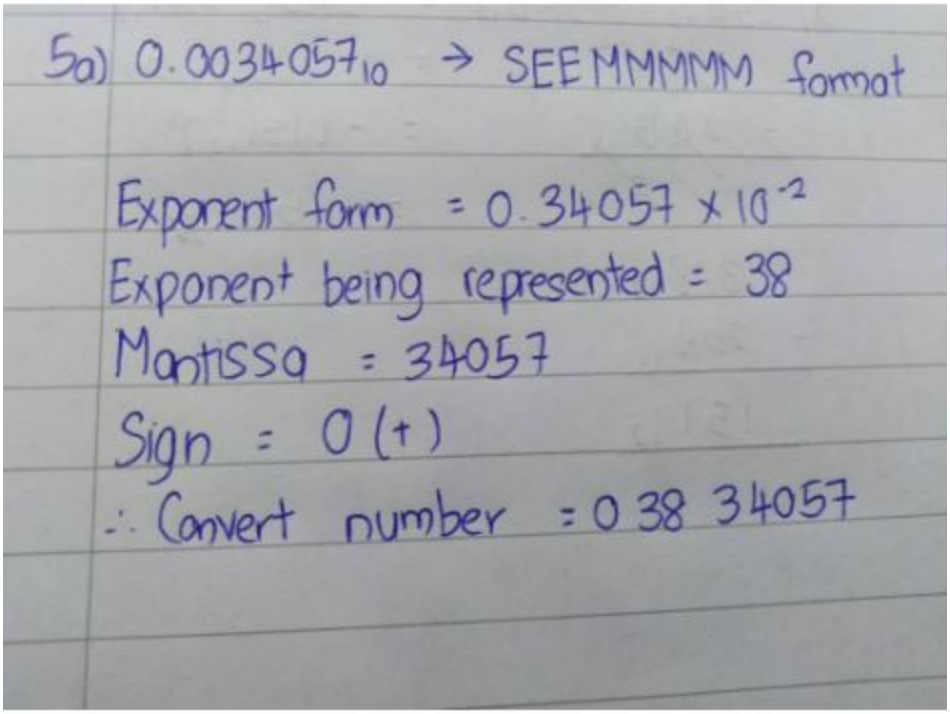
A = 36H = 0011 0110 B
B = 9AH = 1001 1010 B AND
(A AND B) = 0001 0010B
NOT (A AND B) = 1110 1101B

B = 9AH = 1001 1010 B
C = BBH = 1011 1011 B XOR
B XOR C = 0010 0001

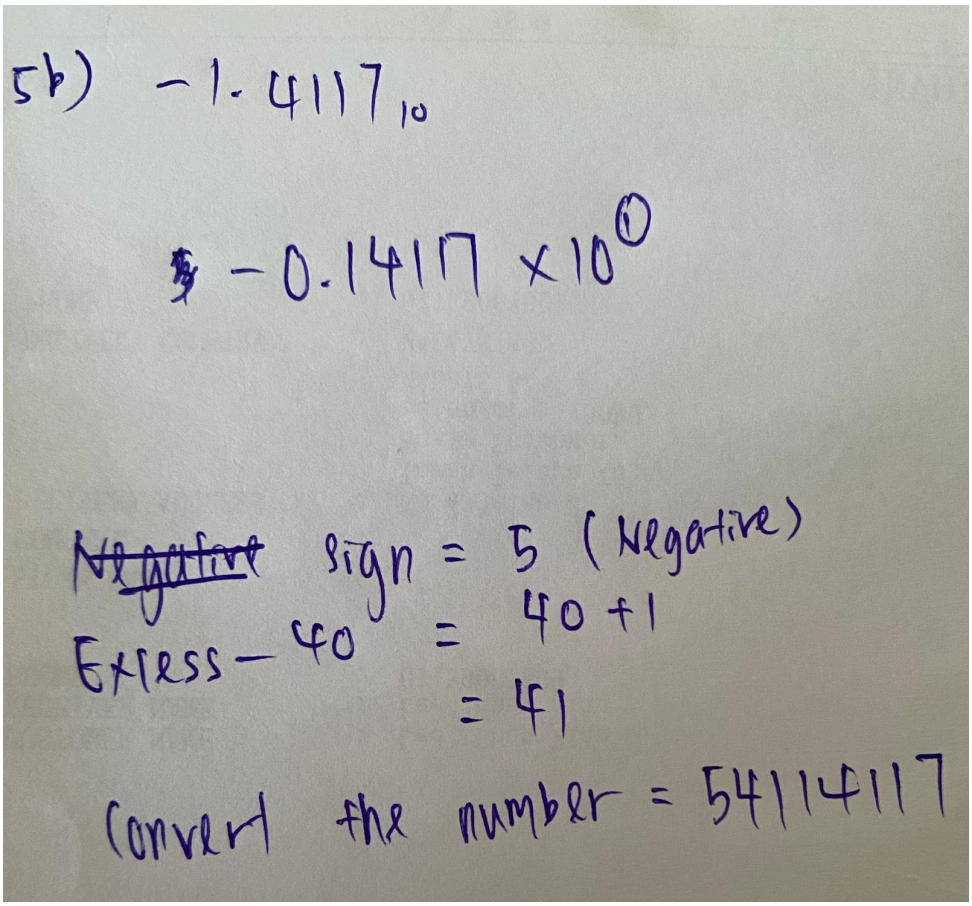
NOT (A AND B) = 1110 1101B
B XOR C = 0010 0001B OR
NOT (A AND B) OR B XOR C= 1100 1101B
= CCH

3a	 =
3b	
3c	
3d	
4a	

	 (ii) 24_{10} - 15_{10} = 9_{10} 0000 1001B = (2^3) + (2^0) = 9d (iii)Valid. Because the computation result is within the range (-127 to +128 = 8 bits total 1 byte = FFh = 255d). Both decimal & binary return the same value. (iv)carry occurs because an extra '1' bit is generated. Overflow does not occur.
4b	(i) $70_{10} - 30_{10} = 1 \times 2^5 + 1 \times 2^3 = 40_{10}$ $70_{10} = 100\ 0110_2$ $30_{10} = 1\ 1110_2$ 2 | 70 2 | 35 — 0 2 | 17 — 1 2 | 8 — 1 2 | 4 — 0 2 | 2 — 0 1 — 0 2 | 30 2 | 15 — 0 2 | 7 — 1 2 | 3 — 1 1 — 1 0001 1110 1110 0001 + 1 1110 0010 1110 0001 + 1 1110 0010 0100 0110 + 1110 0010 (1)0010 1000 (ii) $70_{10} - 30_{10} = 40_{10}$

	0010 1000b = 2^5 + 2^3 = 40 (iii) Valid. Because the computation result is within the range (-127 to +128 = 8 bits total 1 byte = FFh = 255d). Both decimal & binary return the same value. (iv) No overflow occur but carry occur. Detected when extra '1' bit generated.
5a (To continue at Week 4)	

5b



✓

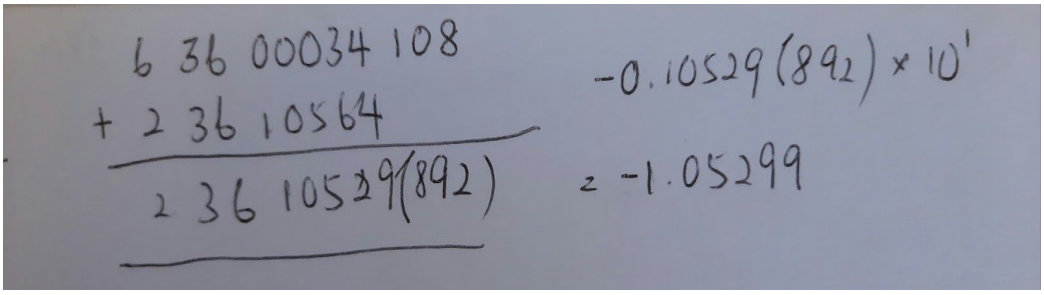
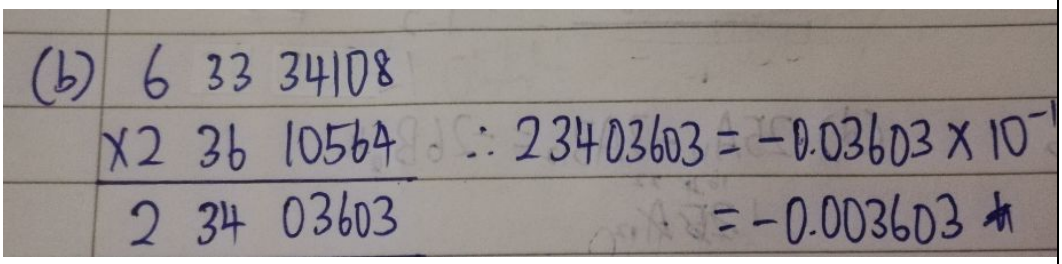
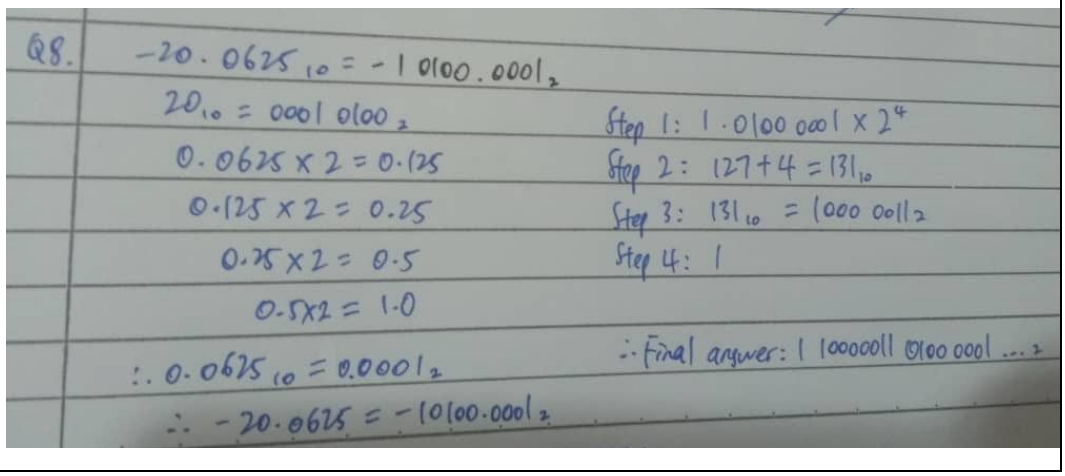
6a

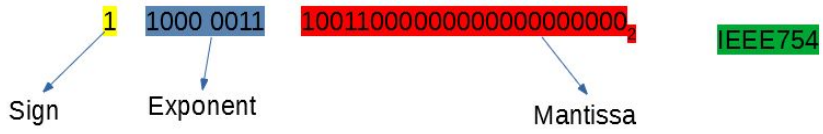
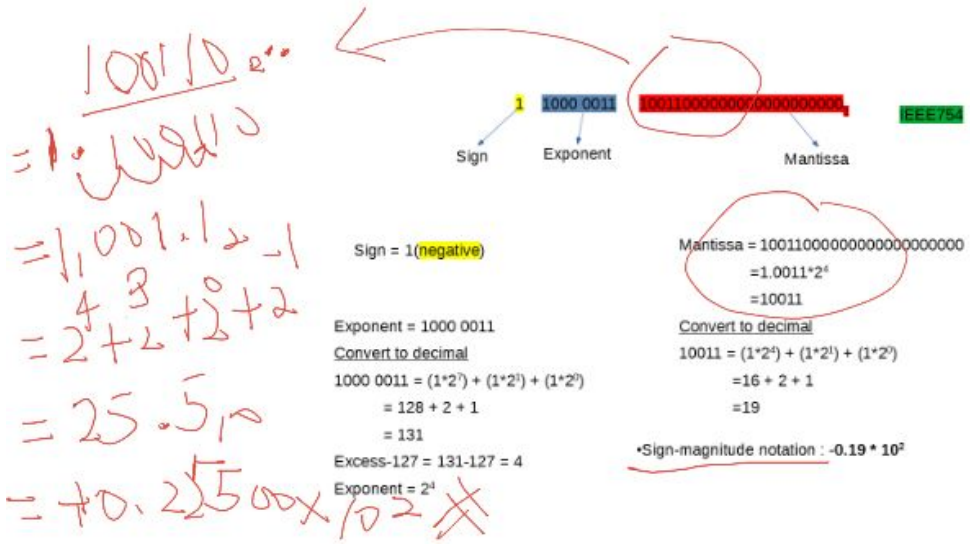
	+1.7250	-0.22375
Exponent	1.7250×10^{0} $= 0.17250 \times 10^{1}$ EE = 50 +1 = 51 ✓	0.22375×10^{0} EE = 50 + 0 = 50 ✓
Sign	+ : S = 0 ✓	- : S = 9 ✓
Mantissa	MMMMM = 17250	MMMMM = 22375

<table>
<tr><td></td><td>
<table>
<tr><td></td><td>✓</td><td>✓</td></tr>
<tr><td>SEEMMMM</td><td>05117250
✓</td><td>95022375
✓</td></tr>
</table>
</td></tr>
<tr><td>6b</td><td>0 51 17250
- 9 50 22375
---
0 51 198475 = 1.7250 - (-0.22375) = 1.94875
---

0 51 17250
- 9 50 22375
Step 1: Adjust exponent: 0 51 17250
- 9 51 022375
Step 2: Subtraction : + 51 17250
- - 51 022375
+ 51 19487(5)
Step 3: SEEMMMMM: 0 51 19488
Step 4: Sign-magnitude / scientific = + 0.19488 x 10^1</td></tr>
<tr><td>6c</td><td>0 51 17250 excess = 51 + 50 - 50 = 51
X 9 50 22375
---
9 51 38597

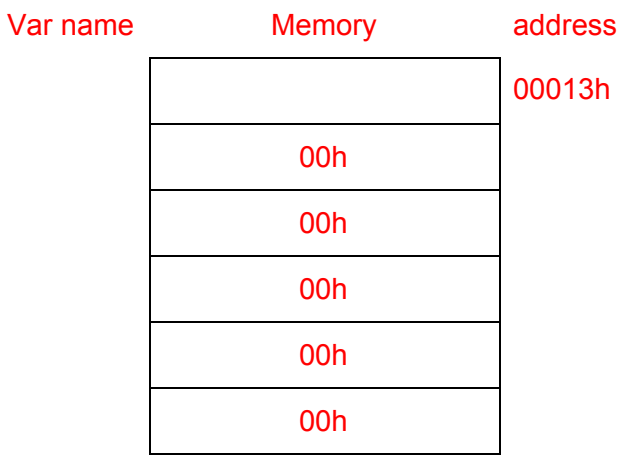
Step 1: excess = 51 + 50 - 50 = 51
Step 2: 0.17250 * (-0.22375) = (-0.038597) * 10^1
= - (o.38597 x 10^-1) * 10^1
Step 3: sign-magnitude notation = - 0.38597 x 10^0
Step 4: SEEMMMMM = 9 50 38597</td></tr>
</table>

7a	 Step 1: Adjust exponent: 6 33 34108 + 2 36 10564 Step 2: Addition : 6 36 00034108 + 2 36 10564000 - 2 36 10529(892) Step 3: SEEMMMMM: 2 36 10530 Step 4: Sign-magnitude / scientific = - 0.10530 x 10^1
7b	 Step 1: Adjust exponent: 33 + 36 -35 = 34 = 10^-1 Step 2: Multiply : 0.34108 * -0.10564 = - 0.03603 Step 3: Sign-magnitude / scientific = -0.36030 x 10^-1 x 10^1 = - 0.36030 x 10^-2 Step 4: SEEMMMMM: 2 33 36030
8	

	✓
9	 Sign = 1(negative) Exponent = 1000 0011 Convert to decimal 1000 0011 = $(1*2^7) + (1*2^1) + (1*2^0)$ = 128 + 2 + 1 = 131 Excess-127 = 131-127 = 4 Exponent = 2^4 Mantissa = 10011000000000000000000 $=1.0011*2^4$ =10011 Convert to decimal 10011 = $(1*2^4) + (1*2^1) + (1*2^0)$ =16 + 2 + 1 =19 •Sign-magnitude notation : $-0.19 * 10^2$ ⇒ Error in Mantissa Correction 

Tutorial 4

1	T'nsam 16bytes(128bits) ✓
2	Kai Yuan RAM - Holds data up to some GB - Holds the operands or instruction that CPU is currently processing - Primary Storage Register - Holds a small amount of data around 32-bits to 64-bits. - Holds The data/instr/address/status that the currently used in program execution in CPU - Special High Speed Storage ✓
3 **(a diagram of a consecutive block of memory is required. Pls update your answer)**	(a) VAR1 db "A" (b) VAR2 dw 2018 (c) VAR3 DD "BACS",00001024h (d) VAR4 DQ 983



		00h	
		03h	
	VAR4	D7h	0000Bh
		00h	
		00h	
		10h	
		24h	
		“B”	
		“A”	
		“C”	
	VAR3	“S”	00003h
		07h	
	VAR2	E2h	00001h
	VAR1	“A”	00000h

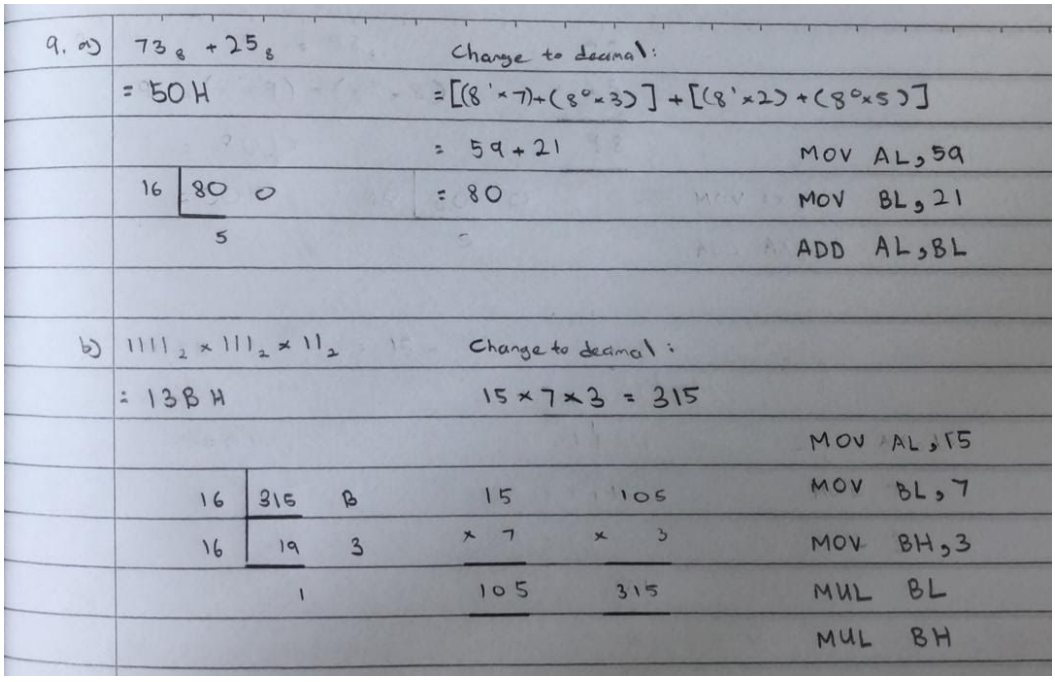
4 (Week 5) **(Select only 1 pair. Pls select the right pair)**	**Keyword**: Instruction CS = 2788_{16} , IP = 1705_{16} 2788 H x 10 H = 27880 H + 1705 H 28F85 H ✓
5	25A34H (absolute address: 20 bits) - DAC4H (offset address: 16bits) 17F70H 17F70H/10 = 17F7H (segment address: 16 bits) ✓

<table>
<tr><td>6</td><td>Kah Wei

<table>
<tr><td>Flag</td><td>Carry</td><td>Overflow</td></tr>
<tr><td>Definition</td><td>Is a flag register used to indicate when an arithmetic carry or borrow has been generated out of the most significant arithmetic logic unit (ALU) bit position
= extra bit in front of MSB</td><td>Is a flag register used to indicate when an arithmetic overflow has occurred in an operation, indicating that the signed two's-complement result would not fit in the number of bits used for the operation.</td></tr>
<tr><td>Detect in signed or unsigned numbers?</td><td>Unsigned</td><td>Signed</td></tr>
<tr><td>How to Detect?</td><td>There is an extra bit at the leftmost position of the answer</td><td>There is an negative/positive bit occurs in the case of addition of two positive bit or two negative bits
1) Both operands have same sign
2) Result at opposite sign</td></tr>
<tr><td>Example</td><td>255 + 8 = 263 which has the answer of:
1111 1111 b
+ 0000 1000b
= (1) 0000 0111b</td><td>127+127 is 254, but using 8-bit arithmetics which is 0111 1111 + 0111 1111 and the result would be 1111 1110 binary, which is negative in two's complement, and thus negative.

0111 1111b
+ 0111 1111b
1111 1110b</td></tr>
</table>

  </td></tr>
<tr><td>7</td><td>A. Parity Flag (PF)</td></tr>
</table>

	B. Counter Register (CX) C. Stack PointerRegister (SP) D. Base Register (BX) ✓
8	a) **CS register**: Hold the start address of code segment b) **Code segment**: Hold the machine instruction c) **Instruction Pointer Register**: Contains the offset address of the next instruction that is to be executed ✓
9	Jia Loong  73_8 + 25_8

CS:IP address	Code segment	
0000:0002	instruction	IP register
0000:0001	instruction	
0000:0000	instruction	CS register

1
120_8
= 0 0101 0 000b
= 0050h

AX = 0050h (normal byte sequence)

AH = 00h	AL = 50h

AX = 013Bh

AH = 01h	AL = 3Bh

Memory = data segment	SI = offset address
01h	0003h
3Bh	0002h
00h	0000h
50h	0000h

Tutorial 5 (Week 6)

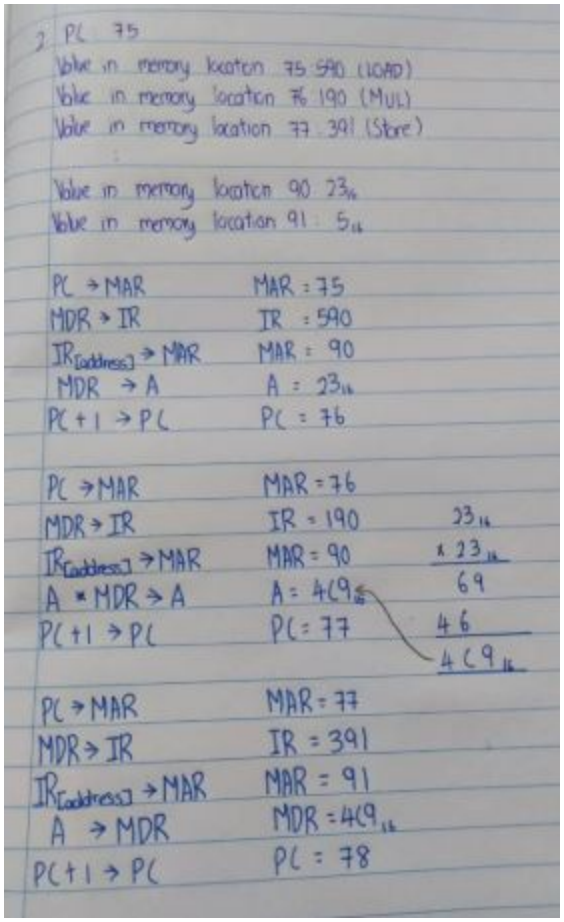
1	Yee Hui (corrected)

1. Instruction 20 :	
PC → MAR	MAR = 20
MDR → IR	IR = 560
IR [Address] → MAR	MAR = 60
MDR → A	A = 422
PC + 1 → PC	PC = 20 + 1 = 21
Instruction 21 :	
PC → MAR	MAR = 21
MDR → IR	IR = 161
IR [Address] → MAR	MAR = 61
A + MDR → A	A = 422d + 008d = 430d
PC + 1 → PC	PC = 21 + 1 = 22
Instruction 22 :	
PC → MAR	MAR = 21
MDR → IR	IR = 360
IR [Address] → MAR	MAR = 60
A → MDR	MDR = 430 d
PC + 1 → PC	PC = 22 + 1 = 23

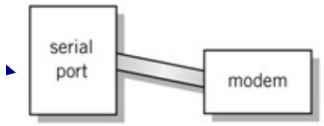
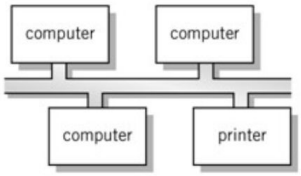
In debug program, all nums (by default) are HEX.
In Assembly language program / other like LMC, all nums (by default) are DEC.

For instruction 21 & 22
A = 422 + 008 = 430

In instruction 22.

<table>
<tr><td></td><td>A → MDR ; MDR = 430</td></tr>
<tr><td>2</td><td>Xin Yi
 

 </td></tr>
<tr><td>3</td><td>Jun Xian
<table>
<tr><td>Bus Architectures</td><td>Point-to-Point Bus</td><td>Multipoint Bus</td></tr>
<tr><td>Connection Between send & receiver</td><td>Directly connects two nodes together.
1-to-1 relationship</td><td>Carries signals to several destinations.
1-to-many relationship</td></tr>
<tr><td>Types of Buses</td><td>Data bus
Control bus</td><td>Data bus
Control bus
address bus</td></tr>
<tr><td>Data Sent</td><td>Unicast / direct</td><td>Broadcast</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>Diagram</td><td> 
</td><td> 
</td></tr>
</table>
✓</td></tr>
<tr><td>4</td><td>Yit Wee

4a)
-Cache memory is a small amount of high speed memory between CPU & main memory
-It is invisible to programmer and cannot be directly addressed
-Cache memory keeps a reproduction of data of memory

How cache memory could be applied in your daily life? ---- 5m
WHen a lecturer (CPU) requires info of a student, the lecturer will look for class rep (cache). If the class rep could answer to the lecturer, the class rep will directly revert to lecturer (cache hit happens). If the class rep does not know the info of his / her classmates, then the class rep seeks for info from the classmate (memory) then reverts to lecturer (cache miss happens).

we act as CPU, when we running, we will need water(cache) to hydrate. if we have enough water, it will revert to energy for us(cache hits happen). if no, we will get more water(memory)

4b)
First step : every memory request goes to the cache controller,which checks the request against each tag.
Second step : if there is a hit, the cache location is used instead of memory
Third step : if there is a miss, a miss requires the cache controller select a line for replacement from memory
Fourth step : after which , the new line in cache is as treated before (cache miss → cache is full) ✓

4c) Data not found in cache. Processor loads data from memory and copies into cache. This results in extra delay. Accessing memory takes time. ✓</td></tr>
<tr><td>5</td><td>Jun Rong
- Memory interleaving is a technique that divides the memory into several</td></tr>
</table>

parts, making it possible to access more than one location at a time

- Each part has its own MAR and MDR and is independently accessible.
- The memory cannot be accessed simultaneously if the locations are in the same block but it can be accessed if the memory locations are located at different blocks.

✓

How would memory interleaving support the operation of a business? --5m

E.g.: An apple is cut into several pieces. It is to illustrate the partition of memory. Each memory partition / piece of apple is placed on a unique plate. Each plate of apple will be sent to the respective customer at the same time. In other words, more customers could enjoy apples at the same time.

Idea:

1) Apples (partition) and serves a few customers (limited pieces to each customer) at the same time.
2) Divide the job to several staff (waiters, cook, cashier, captain, etc) and serve the customer (to handle order, billing, cooking, payment) at the same time

6

Yih Feng

DYNAMIC PARTITION SCHEME

- Available memory are kept in contiguous blocks
- Jobs are given only as much memory as they request when loaded
- Improves memory use over fixes partitions
- Performance deteriorates as new jobs enter the system
- Fragments of free memory are created between blocks of allocated
- memory (external fragmentation).

i. First-fit: Allocate the first partition that is big enough.
ii. Best-fit: Allocate the smallest partition that is big enough

First Fit	Best Fit	Worst fit
Faster to implement	**Slower** to implement because the entire free list table needs to be searched before allocation can be made.	
Algorithm is **simple**.	Algorithm is **complex** because it needs to find smallest block of memory into which the job can fit.	
Memory list organized according to **memory locations, low-order**	Memory list organized according to **memory size, smallest to largest**.	Memory list organized according to **memory size, largest to smallest**.
May not be making **efficient** use of memory space.	Produces the smallest leftover partition. Make **most efficient** use of memory.	Produces the largest leftover partition. Make **worst use** of memory

7a

Kai Yuan

First Fit

Partition	Partition Size	Job	Job Size	Internal Fragmentation
P1	370 KB	J1	75KB	370 - 75 = 295KB
P2	256 KB	J2	125KB	256 - 125 = 131KB
P3	120 KB			
P4	50 KB			
P5	400 KB	J3	398KB	400 - 398 = 2KB
Total Available	1196KB	Total	598KB	428KB

Best Fit

Partition	Partition Size	Job	Job Size	Internal Fragmentation
P1	370 KB	J4	225KB	370 - 225 = 145KB
P2	256 KB	J2	125KB	256 - 125 = 131KB
P3	120 KB	J1	75KB	120 - 75 = 45KB
P4	50 KB			
P5	400 KB	J3	398KB	400 - 398 = 2KB
Total Available		Total	823 KB	323KB

Worst Fit

Partition	Partition Size	Job	Job Size	Internal Fragmentation
P1	370 KB	J2	125KB	370 - 125 = 245KB
P2	256 KB	J4	225KB	256 - 225 = 31KB
P3	120 KB			
P4	50 KB			
P5	400 KB	J1	75KB	400 - 75 = 325KB
Total Available		Total	1125KB	601KB

First fit: J4 has to wait.
Worst fit: J3 has to wait

7b	T'nsam

	Best-fit algorithm makes the **best use** of memory space because it produces the s**mallest leftover (323KB)** partition and it makes the most efficient use of memory. The memory list is organized according to memory size, smallest to largest. ✓
Extra question	Given the following partitions and jobs. Arrange the jobs into memory partition

partition	P_size (KB)		Job	Job size (kb)
PA	500		J1	150
PB	100		J2	350
PC	200		J3	50
PD	50		J4	75

Worst fit

partition	P_size (KB)	Job	J_size	Internal fragmentation = unused memory space
PA	500	J1	150	350
PB	100	J4	75	25
PC	200	J3	50	250
PD	50			
J2 has to wait			Total	**625**

Best fit

partition	P_size (KB)	Job	J_size	Internal fragmentation = unused memory space/ WASTED

PA	500	J2	350	150
PB	100	J4	75	25
PC	200	J1	150	50
PD	50	J3	50	0
			Total	**225**

First fit

partition	P_size (KB)	Job	J_size	Internal fragmentation = unused memory space
PA	500	J1	150	350
PB	100	J3	50	50
PC	200	J4	75	125
PD	50			
J2 has to wait			Total	**525**

Memory

	PA
	PB
	PC

Extra question	Given Partitions: PA, PB, PC, PD, PE with partition size of 100KB, 300KB, 400KB,

50KB, 500KB
Jobs: J1, J2, J3, J4 with job size of 330kb, 120kb, 350kb,40kb
Show the jobs are allocated in the memory partition when the following memory allocation algorithms are applied respectively.
i) First fit algorithm - 3m
ii) Best fit algorithm - 3m
iii) Worst fit algorithm - 3m
Then, comment on your answer. - 5m
The most efficient algorithm is best- fit ---1m
The total internal fragmentation generated by best fit is 410kb-- 1m
According to best fit, all jobs fit to the memory partition. --- 1m
Explain IF. : IF is unused memory space. One memory partition only allows one access. Once a job occupied a partition, the rest of the memory space in that partition is unused & wasted - ---1m
Explain: lower IF is the best. FF generated 460kb IF and WF generated 710kb IF ---1m

WORST FIT

P	P_SIZE(KB)	J	J_SIZE(KB)	IF
PA	100			
PB	300	J4	40	260
PC	400	J2	120	280
PD	50			
PE	500	J1	330	170
J3 have to wait			TOTAL	710

BEST FIT

P	P_SIZE(KB)	J	J_SIZE(KB)	IF
PA	100			
PB	300	J2	120	180
PC	400	J1	330	70
PD	50	J4	40	10
PE	500	J3	350	150
			TOTAL	410

FIRST FIT

P	P_SIZE(KB)	J	J_SIZE(KB)	IF
PA	100	J4	40	60
PB	300	J2	120	180
PC	400	J1	330	70
PD	50			
PE	500	J3	350	150
			TOTAL	460

Tutorial 6 (Week 7)

1

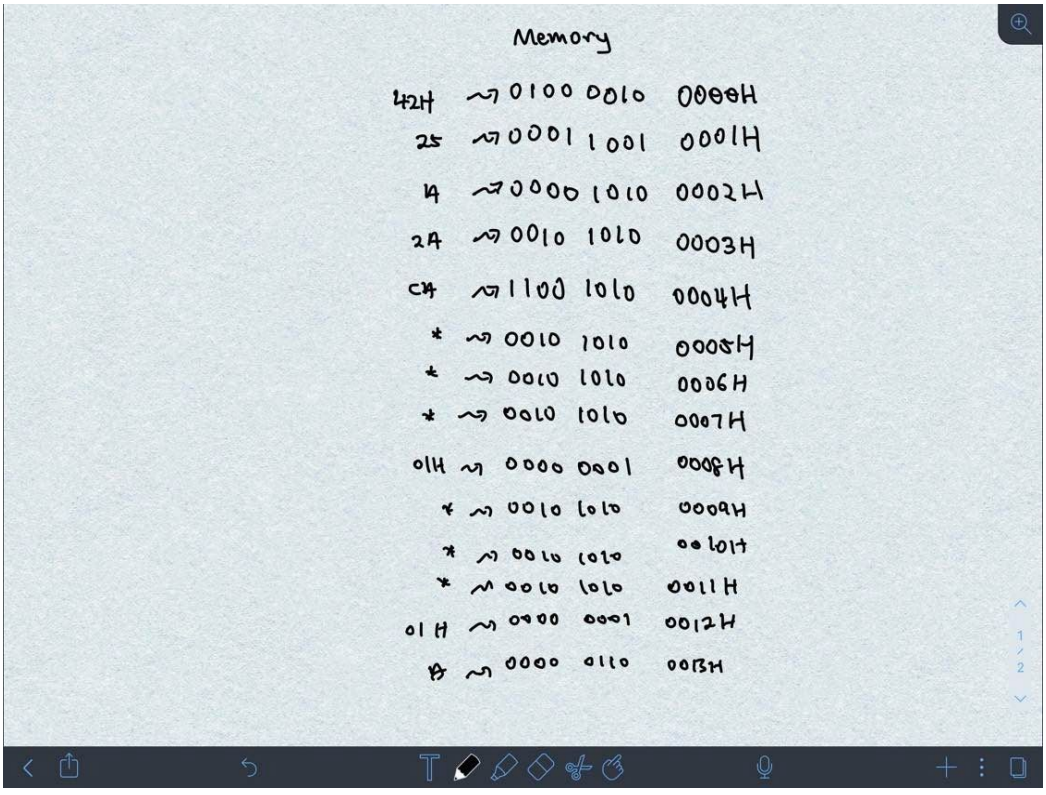
a) val1 DB 4DH ✓

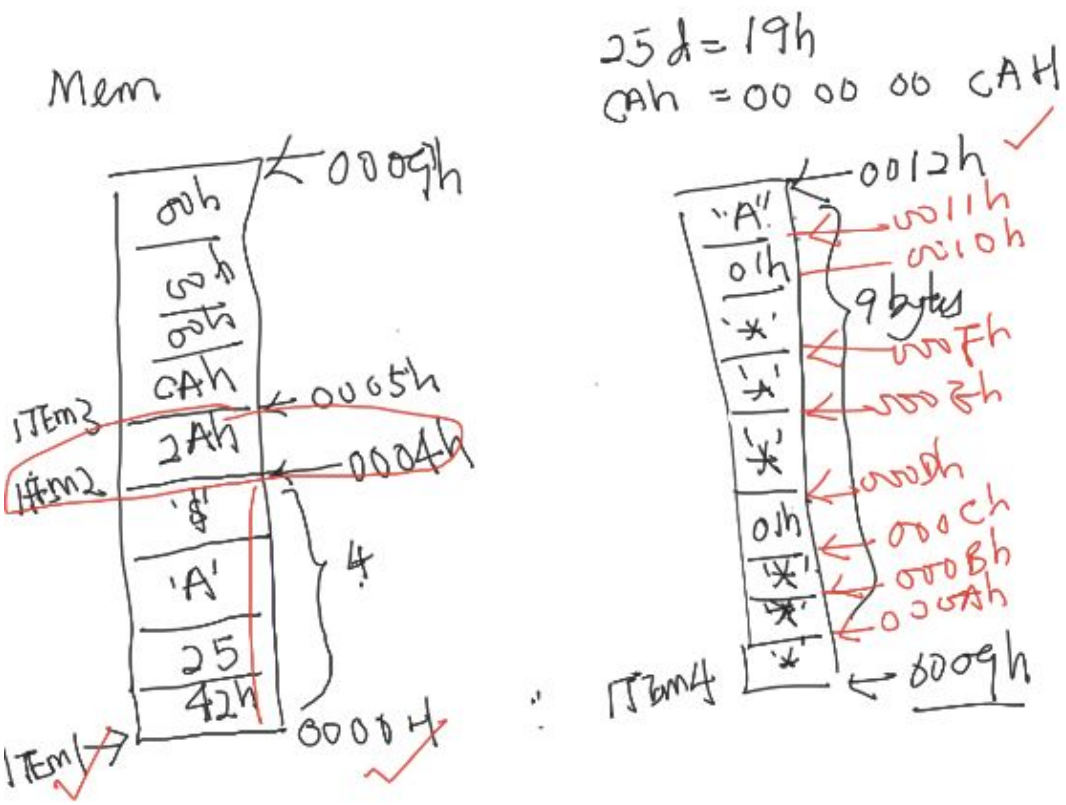
b) val2 DW FFFFH ✓

c) val3 DD 0000FFFFH ✓

2

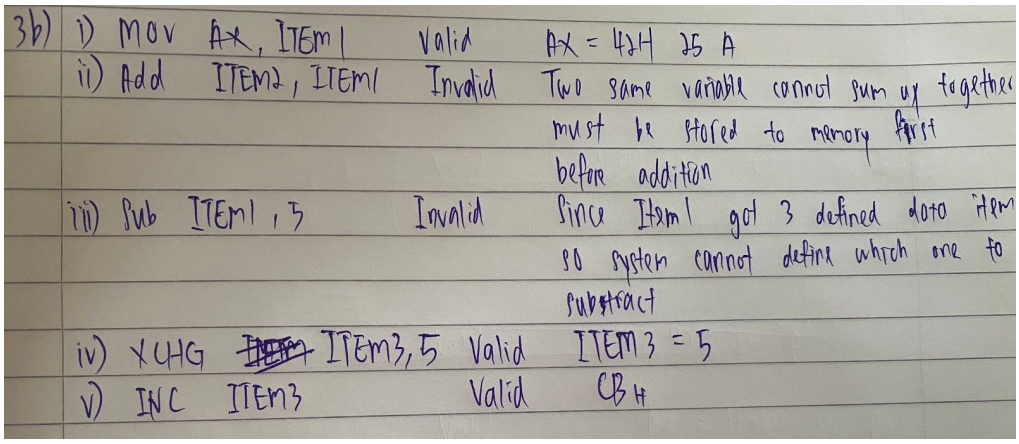
Data Item	Valid / Invalid	Justification / Result
ITEM1 DB "A", "B"	Valid ✓	Memory ITEM1[0] = "A" ITEM1[1] = 'B'
2ITEM DB 2AH	Invalid ✓	Variable name cannot start

<table>
<tr><td></td><td>
<table>
<tr><td></td><td></td><td>with a number. ✓</td></tr>
<tr><td>$ITEM DB “$”</td><td>valid ✓</td><td>$ITEM = “$” ✓</td></tr>
<tr><td>NAME DB “TEST”</td><td>Valid ✓</td><td>MEMORY
NAME[0] = “T”
NAME[1] = “E”
NAME[2] = “S”
NAME[3] = “T”</td></tr>
<tr><td>INC DB “2015”, “$”</td><td>Invalid ✓</td><td>INC is a keyword for
instruction ✓</td></tr>
</table>
</td></tr>
<tr><td>3a</td><td>Kah Wei
 
</td></tr>
</table>



3b

Chun Xian



(i)	Valid but not recommended	Because different in size. AX =0042h
(ii)	invalid	Memory cannot perform any operation. MOV v1,v2 ; v1 & v2 store in memory To perform operation, data shall be fetched into CPU (made use of register.)

	(iii)	valid	ITEM1 = IEM1[0] SUB ITEM1,5 ; ITEM1[0] = ITEM1[0] -5 = 3Dh
	(iv)	INVALID	Cannot swap constant (fixed value)
	(v)	valid	ITEM3 = ITEM3 + 1 = 000000CAH +1 = 000000CBh

4

Pei Xuan

INSTRUCTION	WORKING	OF	SF	ZF	CF
i) MOV AX, 1095H	AH: 0001 0000 / AL: 1010 0101	NV 0	PL 0	NZ 0	NC 0
ii) ADD AH, 2AH	2AH = 0010 1010 B AH = **0**001 0000 B + **0**010 1010 B () **0**011 1010 B	NV 0	PL 0	NZ 0	NC 0
iii) SUB AL, 95H	95H = 1001 0101 B AL =1010 0101 B - 1001 0101 B 0000 0000 B	NV 0	PL 0	ZR 1	NC 0
iv) MOV BL, 5	BH: **0**000 0000 / BL: **0**000 0101	**NV 0**	PL 0	NZ 0	**NC 0**
v) MUL BL	AX = 0011101000000000 B BL = X 0000 0101 B	NV 0	PL 0	NZ 0	CY 1

		0011101000000000 0011101000000000 (01)**0**010001000000000 B				

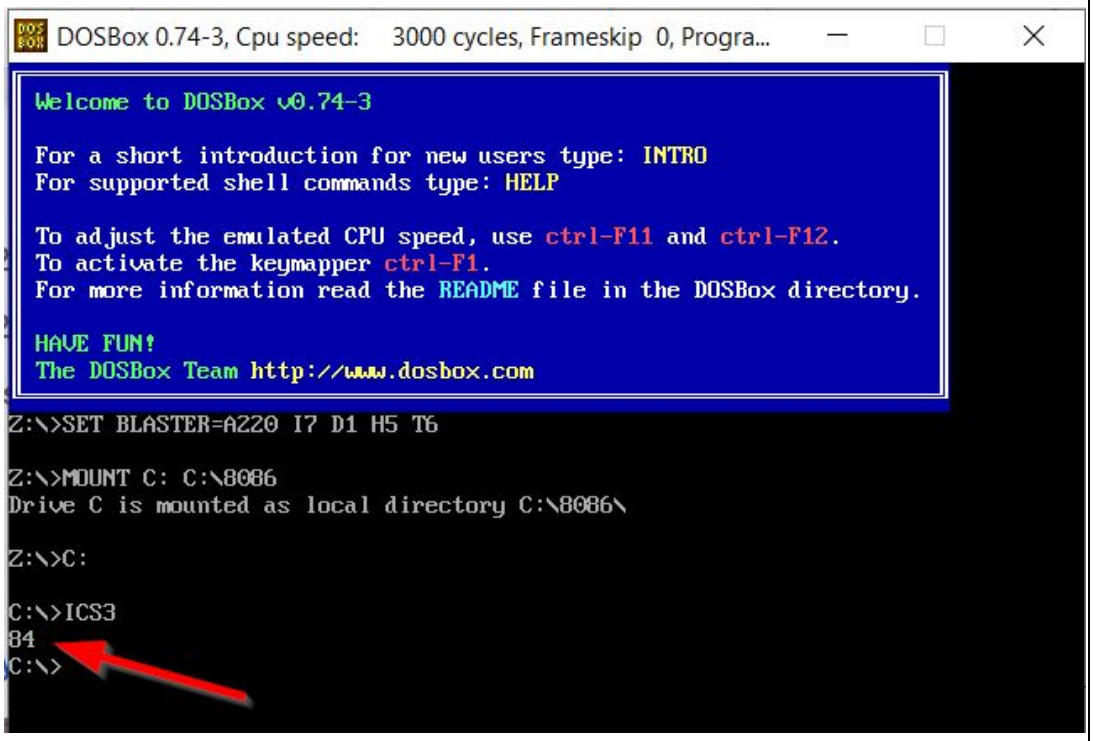
Tutorial 7: Assembly Language Fundamental – Part II

1.	Mun Jun (1) <line1> .MODAL SMALL > MODEL SMALL (2) <line5> .MAIN PROC > MAIN PROC (3) <line7> MOV DX, AX > MOV DS, AX - move to DS (Data Segment) (4) <before line 9>MOV CX, 3 - missing loop amount (5) <line9> PRINT > PRINT: (6) <line11> MOV AL, 01H > MOV DL, 01H - display output need use DL/DH (7) <line12> INT 21 > INT 21H - without H, its useless (8) <line13> LOP PRINT > LOOP PRINT (9) <line15> MOV AX, 4Co0H > MOV AX, 4C00H/ MOV AH, 4CH (10) <line18> MAIN END > MAIN ENDP

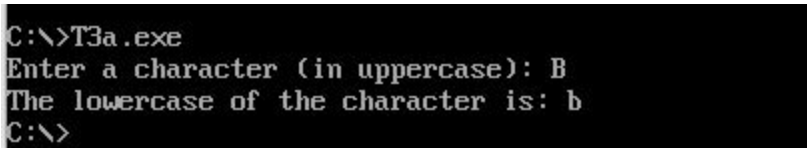
2a	Jun Wai
	mov al, ITEM1[0]
	add al, ITEM1[1]
	add al, ITEM1[2]
	add al, ITEM1[3]
	add al, ITEM1[4]
	add al, ITEM1[5]
	add al, ITEM1[6]

2bc	Jia Loong .MODEL SMALL .STACK 100 .DATA ITEM1 DB 3,6,9,12,15,18,21 TEN DB 10

```
.CODE
MAIN PROC

        MOV AX,@DATA
        MOV DS,AX

        MOV BL,ITEM1[0]
        MOV SI,1
        MOV CX,6

        ;To sum up all the values in the array
        L1:

                ADD BL,ITEM1[SI]
                INC SI


        LOOP L1


        ;To display the final answer in 2 digits
        MOV AH,0H
        MOV AL,BL

        DIV TEN

        MOV BX,AX

        MOV AH,02H
        MOV DL,BL
        ADD DL,30H
        INT 21H

        MOV AH,02H
        MOV DL,BH
        ADD DL,30H
        INT 21H


        MOV AX,4C00H
        INT 21H
```

	MAIN ENDP END MAIN 
3a	Yee Hui Given the sample output information Sample output: **Enter a character (in uppercase): A** **The lowercase of the character is: a** a) Write an assembly language program to convert the uppercase character to lowercase, by adding 20H to the user input. .MODEL SMALL .STACK 100 .DATA PROMPT_1 DB "Enter a character (in uppercase): $" PROMPT_2 DB "The lowercase of the character is: $" VALUE_1 DB ? .CODE MAIN PROC

```
 MOV AX,@DATA
 MOV DS,AX

 ;LOAD AND PRINT PROMPT_1
 MOV AH,09H
 LEA DX,PROMPT_1
 INT 21H

 ;READ A LETTER OF PROMPT_1
 MOV AH,01
 MOV BL,AL
 INT 21H

 ;SAVE THE LETTER IN BL
 MOV BL,AL

 ;--NEWLINE
 MOV dl,10
 MOV ah,02h
 INT 21H

 ;PRINT PROMPT_2
 MOV AH,09H
 LEA DX,PROMPT_2
 INT 21H

 ;CONVERT UPPERCASE TO LOWERCASE
 ADD  BL,20H

 ;STORE LOWERCASE INTO BL
 MOV VALUE_1,BL

 ;PRINT LOWERCASE
 MOV AH,02
 MOV DL,VALUE_1
 INT 21H

 MOV AH,4CH
 INT 21H

MAIN ENDP
END MAIN
```

	 C:\>T3a.exe Enter a character (in uppercase): B The lowercase of the character is: b C:\>

3b	Xin Yi
	Modify the program, convert a lowercase letter to uppercase letter using XOR instruction.
	
	.MODEL SMALL
	.STACK 64
	.DATA
	STR1 DB "Enter a character (in lowercase): $"
	STR2 DB "The uppercase of the character is : $"
	
	NL DB 13,10,"$"
	
	KEY DB 20H
	
	CHAR DB ?
	CONVERT DB ?
	.CODE
	MAIN PROC
	MOV AX,@DATA
	MOV DS,AX
	
	MOV AH,09H
	LEA DX,STR1
	INT 21H
	
	MOV AH,01H
	INT 21H
	MOV CHAR,AL ;STORE THE INPUT CHAR TO VARIABLE
	
	; ADD CHAR,20H ;USE IT FOR QUESTION 3A
	MOV BL,CHAR
	; MOV CONVERT,BL ;USE IT FOR QUESTION 3A
	XOR BL,KEY ;KEY IS 20H
	MOV CONVERT,BL
	; STEP TO CONVERT
	; C = 43H
	; 0100 0011 = 43H
	;XOR 0010 0000 = 20H
	; 0110 0011 = 63H = c
	
	MOV AH,09H

	LEA DX,NL
	INT 21H
	
	MOV AH,09H
	LEA DX,STR2
	INT 21H
	
	MOV AH,02H
	MOV DL,CONVERT
	INT 21H
	
	MOV AH,4CH
	INT 21H
	MAIN ENDP
	END MAIN

4	Chun Xian
	mov si,0
	mov cx,256
	
	l1:
	
	mov bx,si
	mov bh,0
	
	mov ah,02h
	mov dl,bl
	int 21h
	inc si
	
	loop l1
	
	 

Tutorial 8: I/O Facilities - Week 11

1.

Jun Xian

Differentiate between two key I/O handling techniques namely direct memory access and programmed I/O.

Programmed I/O: Each data item transfer is initiated by an instruction in the program. Each instruction produces a single input / output.

Direct Memory Access: each data item is transferred directly from the I/O devices to the memory and vice versa.

I/O handling techniques	Programmed I/O	Interrupt I/O	Direct Memory ACcess (DMA)
For devices	Slow	All	Fast
Sample devices involved	keyboard	Any	Pen drive, digital camera
Data transfer rate	Byte basis	Bit basis	Block basis
CPU involvement	Full	CPU only receive INT	CPU initiates the start of transfer & receive INT for status update
Advantage / pros	Full control	CPU get latest update	Support multitasking

References: https://www.geeksforgeeks.org/io-interface-interrupt-dma-mode/

2a.

Kah Wei

Considering the interrupt that occurs at the **completion** of video transfer from digital camera to memory.
a.) “Who” is interrupting “whom”?

	(1) Instruction (from CPU) CPU ←→ I/O Module ←→ I/O device (lecturer) (class rep) (student) (3) INT (to CPU) | ^ | | Memory ←- --- (2) data (G.Classroom) I/O device (digital camera) is interrupting the CPU, through the I/O module. Only sender of INT is I/O module Only receiver of INT is CPU
2b.	Yit Wee b.) Why is the interrupt used in this case? The interrupt is used to signify and signal that the block of video transfer (since digital camera are inherently block devices) is either 'ready' (ready to be transfer) or 'written' (has been transferred to the memory), and is used to signify, basically, that the operation has been **'done'**. INT could be used as a notifier / indicator for status update, for: 1) The start / **end of event** (E.g.: Before permanently deleting a file, **to notify the completion of file / video transfer**.) 2) The abnormal event event (E.g.: Division by zero) 3) The external event (E.g.: Plug in of pen drive) 4) Multiprocessing / time sharing. (E.g.: Once the printer has done the PrintJob1, an INT is sent to the CPU. so that the CPU can start / send instructions to start printing PrintJob2.) References: https://www.justanswer.com/computer-hardware/5z94e-consider-interrupt-occurs-completion-disk.html
2c.	Chia Chung c.) Describe the steps that take place after the interrupt occurs. In reference to the interrupt that occurs at the completion of a disk transfer, the disk controller interrupts the CPU to announce the transfer is complete. The interrupt signals the data is in memory and ready to use and tells the CPU to continue processing its previous instructions. Without interrupt capability, the computer would need to rely on polling to control the flow of data and to determine when the data transfer was complete. DMA is designed to speed up data transfer from external devices to memory. Polling would cause the computer to

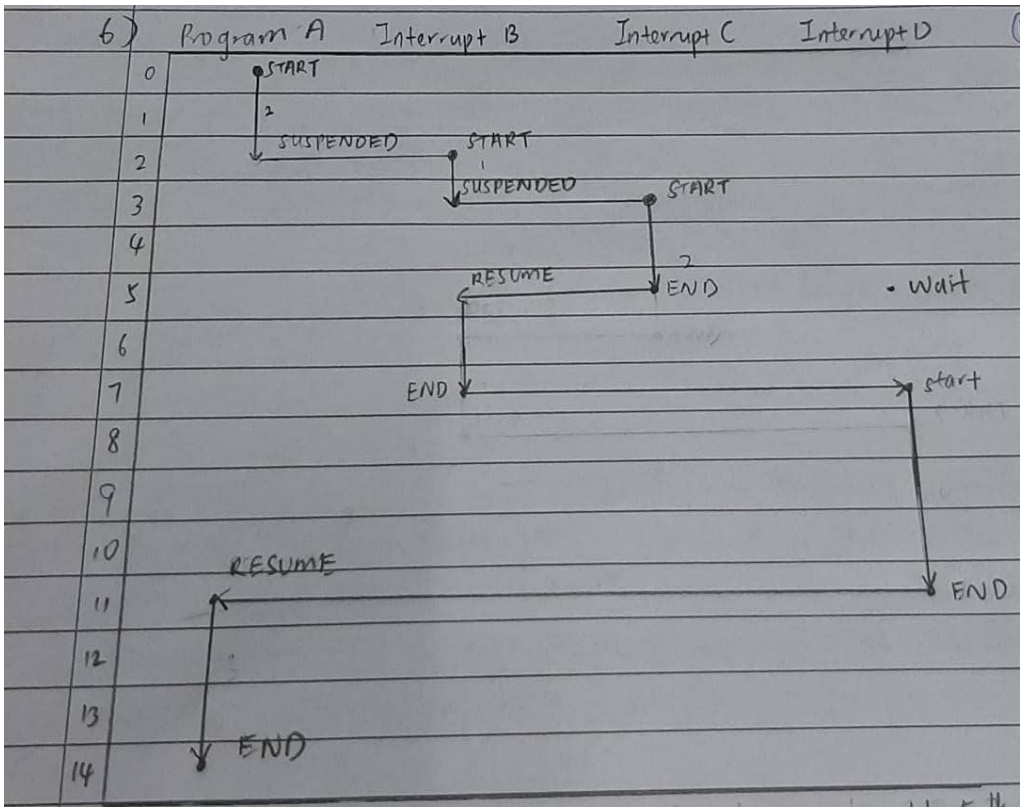
	transfer data at a very slow rate to assure that the computer did not exceed the ability of the device. After the interrupt occurs at the end of the data transfer, control is returned to the program that initiated the request or notifies the operating system that the program can be resumed. The CPU then completes the current instruction, restores the registers saved in the stack area (or an area known as a process control block), restores the program counter, and then resumes the original program exactly where it left off. **Example (real life example) & elaboration** We are having class now (current process) → Your phone rang (interrupt received) → temp **suspend** / pause the current class (suspend current process) → remember where you stopped / paused (all current status are recorded into memory, Process COntrol Block (**PCB**) → answer the phone (**handle interrupt**) → Restore / remember / recall the pause point before paused, then resume class (**resume** current / interrupted process process.) References: https://www.coursehero.com/file/p6iuqlc/Describe-the-steps-that-take-place-after-the-interrupt-occurs-Englander-293-In/#:~:text=The%20CPU%20then%20completes%20the,exactly%20where%20it%20left%20off.
3	Jun Rong What is polling used for? What are the disadvantages of polling? What is a better way to perform the same job? Polling is the process where the computer or controlling device waits for an external device to check for its readiness or state, often with low-level hardware. For example, when a printer is connected via a parallel port, the computer waits until the printer has received the next character. These processes can be as minute as only reading one bit. **Disadvantage**: - If there are too many devices to check, the time required to poll them can exceed the time available to service the I/O device. - Data loss may occur as the CPU checks the register according to the clock such as every single second. However, if the data arrives at 1.5s, then it will miss the data, resulting in data loss Interrupt is better as it is able to serve multiple devices within a short period of time. Real-life example: Suppose you are waiting for your friend. There are 2 ways to know if your friend has arrived or not. First is you wait at the door and when your friend arrives, you get to know. Another way is that you do not wait at the door, instead, you continue

	with your own work until your friend rings the doorbell. The first way is polling approach while the second way is the interrupt approach. To identify the sender or INT, CPU uses 2 methods: 1) Polling interrupt--> Interrupt was sent without sender's details. CPu needs to check with the I/O module whether there are the one which sent the INT. That's why no solution / no interrupt handling routine could be identified. So, CPU cannot handle interrupt immediately. Problems: (a) Takes time (Took many devices to check) (b) data may be corrupted / lost. Solution: Vectored interrupt 2) Vectored interrupt → INT sent together with sender's address Advantage: CPU can handle INT soonest
4	Ming Jun Explain why programmed I/O does not work very well when the I/O device is a hard disk or a graphics display? This is because most PIO architectures are based on the basic load/store bus architecture model. The CPU issues an operation via a PIO, it goes to the device, the device does something and returns a result. Depending on the specifics of the architecture, the CPU (or the core) may be blocked while it waits for the device to respond. This is inefficient because it is fully synchronous. Refer to the table in Q1. https://www.quora.com/What-is-the-explanation-for-the-reasons-why-programmed-IO-does-not-work-very-well-when-the-IO-device-is-a-hard-disk-or-a-graphics-display
5 (Week 13)	Yih Feng CPU interface:Performs CPU interfacing tasks ● Accept I//O commands from the CPU ● Eg : MOV AH, 01H(input), MOV AH, 02H (OUTPUT) ● Sending interrupts and status information to CPU ● EG : INT 21H, INT 10H Device interface:supplier control of the device ● Manage the operation of devices ● E.g.: manage printing on the printer
6	T'nsam

Given the following information, draw timeline to depict how CPU handle multiple interrupts.

Process	Arrival Time (ns)	CPU / Processing Time (ns)	Priority
Program A	0	5	4
Interrupt B	2	3	2
Interrupt C	3	2	1
Interrupt D	5	4	3

Which process spent the longest waiting time? Comment on your answer.



Process	A	B	C	D
Finish time(FT)	14	7	5	11
Arrival time(AT)	0	2	3	5
CPU Time(CT)	5	3	2	4
Waiting time = FT-AT-CT	9	2	0	2

Process A has the longest waiting time (9s).

7

Kai Yuan

Yes.

This is because without interrupts the CPU would not be able to know of direct

	memory access status or completion of action. The CPU cannot start a new / next process. E.g.: After the print job 1 is completed, if the INTERRUPT is not sent to the CPU, CPU cannot initiate the start of print job 2. As the result, the printer is waiting for the print job 2 while the print job 2 is waiting for the printer

Tutorial 9: Operating Systems

1	Define operating system. Jun Wai Operating system is: ● A program that allows computer users to communicate with the computer such as the hardware and software. ● Allow users to perform hardware and software installation based on the users' needs ● Computer users are able to manage their computer memory, files and so on.
1	Give TWO (2) examples of operating systems for desktop computer and TWO (2) examples of operating systems for mobile devices. Mun Jun operating systems for desktop computer - Windows, MacOS operating systems for mobile devices - Android, iOS
2	List and explain key functions of an operating system. Give example(s) to support your answer. Yee Hui **Memory management** - The operating system needs to perform the task of allocation and deallocation of memory space to programs in need of these resources. - For example, it will keep tracks of primary memory, which bytes of memory are used by which program.
2	List and explain key functions of an operating system. Give example(s) to support your answer. Pei Xuan **Device management** - Device management is responsible for managing all the hardware devices of the computer system. - The responsibility of the operating system is to keep track of the status of all the devices

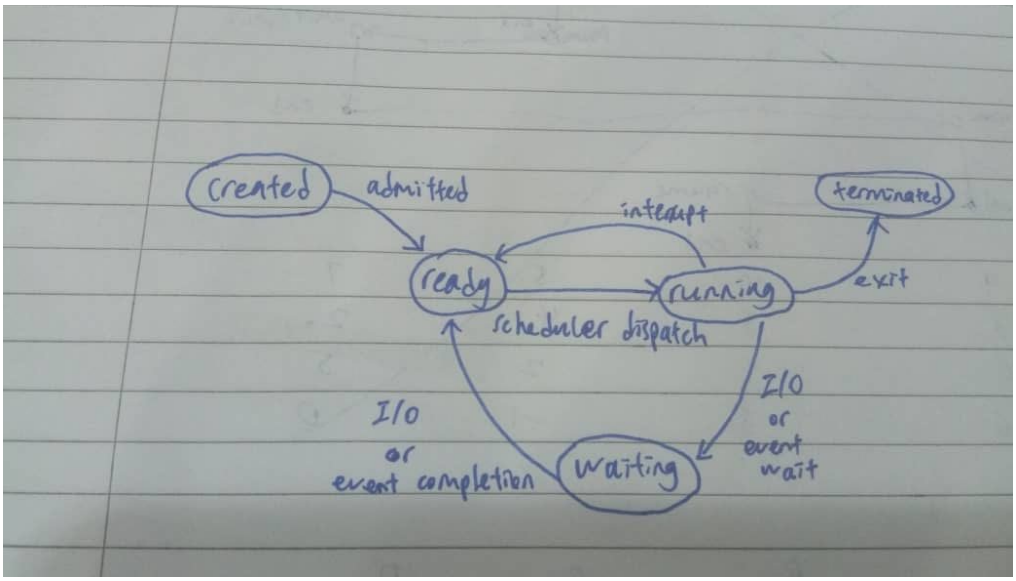
	in the computer system. Program responsible for this task is known as the I/O controller. - It decides which process gets the device when and for how much time. - It also allocates and deallocates devices in an efficient way. - For example, the various device controllers in a computer system may be a disk controller, printer controller, tape-drive controller and memory controller.

2	List and explain key function of an operating system. Give example(s) to support your answer.
	
	Xin Yi
	File Management
	- A file management is organized into directories for efficient or easy navigation and usage. These directories may contain other directories and other files.
	- OS carries out the following file management activities.
	  - It keeps track of each file through directories that contain the file's name, location in secondary storage and other important information.
	  - Allocate the resources (opening file) .For example, it will allocate the files by activating the appropriate secondary storage device and loading the file into the main memory while also updating the records of who is using what file.
	  - Deallocating the files when their use in finished and are not needed, and also communicating to others about it's availability which are waiting for it .For example, it deallocates the file by updating the file tables and rewriting the updated file into the secondary storage, then communicating with other processes and notifying them about it's availability.
	
	- For example, all file managers allow the user to view, edit, copy, and delete the files on their computer storage devices.

3a	Chun Xian
	A clerk is assigned to print the bills and mail these bills to the customers on a monthly basis.
	
	Batch OS
	- The task can be work continuously without the interaction with the user
	- It performed based on throughput
	- It need to group together based on regular basis
	  - for example the clerk need to wait and combine the bills at the end of month first only can continue the work
	- It involve minimum user involvement
	  - If the clerk want to print out all the bills to the customer, the clerk need to select the bills first and click on print button once time and wait the machine work
	- Therefore, batch operating system is most suitable to fit in this scenario
	
	Correction
	- The clerk needs to combine all the bills first based on the monthly basis and only can start the job.
	- When the clerk going to print or mail the bills to the customer, he/she just need to select all the bills first in the folder, and click on the print button once time without any clicking many times to the print button and just need to wait the machine work, therefore it involve

	minimum user involvement - And the task can be worked continuously without any interaction with the clerk.
3b	Jun Xian A multi-function laser printer which offers the features such as print, scan, fax, email, smart-card reader, connection to digital camera and so on. Embedded OS Multi-function laser printer might have a computer embedded inside the printer to have all the functions and features mentioned above. The main function of a printer is printing, but by using Embedded OS additional features can be added for examples, scanning, faxing, email, smart-card reader, digital camera connections and so on. Our campus also uses a multifunction laser printer for convenience purposes.
3c	Kah Wei = batch OS, interactive OS, real time OS, embedded OS or **hybrid OS**? Interactive OS An interactive operating system is one that allows the user to directly interact with the operating system whilst one or more programs are running. There will be an user interface in place to allow this to happen Same OS can be implemented on multiple platforms. advantage : it flexible E.g.: Hybrid car (Honda Jazz) = petrol / electricity
3d	Yit Wee Embedded OS Fuzzy Logic System is supported by Embedded OS because it is very flexible and contains multi functions. With Embedded OS various programs / options are provided to the user to select. For examples The automatic rice cooker helps to ensure the rice is properly cooked with different cooking options (e.g.: Steam, cook, bake, boil, etc)..
3e	Chia Chung Interactive OS ● An Interactive OS allows the user to directly interact with the operating system while one or more programs are running. ● Interactive OS is able to provide feedback to the user immediately. ● Interactive OS are those systems which take the input form the user. E.g. from a human being then produces an output. ● The input can be any form of gestures like pressing the button or selecting something by typing on the keyboard. ● E.g.: CUstomer / visitor clicks for the search criteria (e.g.: category: Food and beverage) to search for restaurants available. The search is done based on user input / interaction.

Tutorial 10: Processor Management

1.	Jun Rong - CPU Scheduling Information (Process priority and pointers to scheduling queues) - Memory management information (page table or segment table) - I/O status information (list of I/O devices) - Registers (Accumulator, base, registers and general registers) PCB stores: - Process ID (Process name) - Process status (hold, ready, run, wait, finish) - Process state (registers, memory, etc) - Accountability (summary, host)
2.	Ming Jun  Process cycle: Create to ready: Process is called and put into ready queue Ready to run: The process from ready queue is send to CPU to execution Run to wait: The current executing process is waiting for I/O Wait to ready: The process resume to ready queue Run to finish: The process is completed Real life example: Create to ready: Assignment question is uploaded and scheduled to be showed to student Ready to run: When the time is up, the assignment question is displayed. Run to wait: Student may ask question / submit answer Wait to ready: If the student asking the question, the lecturer explains the answer. Once the explanation is clear, the student is ready to do the assignment. Run to finish: After the student submitted the assignment, it is considered as done.
3.	Yih Feng

<table>
<tr><td></td><td>Context switch = environment changing
How it works
- Saves a job's processing information in its PCB
- Current job can be swapped out of memory
- Loads the processing information from the PCB of another new coming job into a register

Advantages
- Better control over validity of the data / all info will be recorded in PCB
- Software can be more selective / multitasking

Disadvantages
- Requires considerable processor time = delay CPU processing
- Does no useful work while switching</td></tr>
<tr><td>4. (Week 14)</td><td>T'nsam
3 guidelines to good process scheduling policies
-Minimize the response time by quickly turning around interactive requests.
-Minimize turnaround time by moving entire jobs in/out system quickly.
-Minimize waiting time by moving jobs out of READY queue as quickly as possible.

*turnaround time = FT-AT
*Waiting time = FT-AT-CT</td></tr>
<tr><td>5.</td><td>Tan Kai Yuan
Preemptive Scheduling is the scheduling which takes place when a process switches from running state to ready state or from waiting state to ready state, the processes can be scheduled. (The process might be partitioned, need to stop at each process arrives and run the next process based on algorithm / external interrupt is allowed) .e.g: PS, SRT, RR

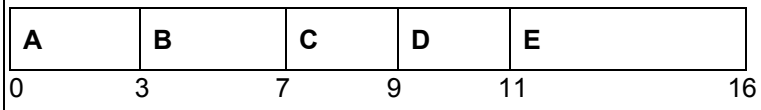
Non-Preemptive Scheduling is the scheduling which takes place when a process terminates or switches from running to waiting for state, the processes can not be scheduled. (The process will be executed as a whole) e.g.: PS, FCFS, SJN</td></tr>
<tr><td>6.</td><td>Jun Wai
<table>
<tr><th>Policy</th><th>Algorithm</th><th>Based on</th><th>Pros / advantage</th><th>Cons / disadvantage</th></tr>
<tr><td>Non-preemptive</td><td>Shortest Job First (SJN)</td><td>Shortest CPU Time
(AT + CT)</td><td>Job available at same time</td><td>Interactive system</td></tr>
<tr><td>Preemptive</td><td>Shortest remaining time (SRT)</td><td>Shortest remaining CPU cycle
(AT + RCT)</td><td>Fastest completion</td><td>Interactive system
(system is not able to respond immediately to the long processes)</td></tr>
</table></td></tr>
</table>

Preemptive	Round robin (RR)	Quantum (AT + Q + RQ)	Equally share CPU / fair	Shortest remaining (queue again, shortest remain CT process cannot be completed first)
Non-preemptive	First Come First Serve	Arrival time (AT)	Batch system (sequential. min user intervention)	Interactive system
Non-preemptive / preemptive	Priority scheduling (PS)	Priority (AT + P)	Preferential system	Low priority jobs will keep waiting
Waiting Time =	Finish Time - CPU Time - Arrival Time			
Turnaround time=	Finish Time - Arrival Time			

7ab

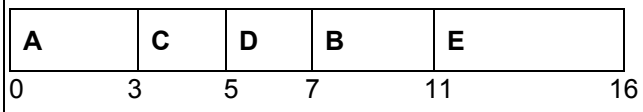
Mun Jun

A. FCFS Scheduling



Process	AT	CT	P	FT	TT	WT
A	0	3	5	3	3-0 = 3	0-0 = 0
B	2	4	2	7	7-2 = 5	3-2 = 1
C	3	2	4	9	9-3 = 6	7-3 = 4
D	5	2	1	11	11-5 = 6	9-5 = 4
E	8	5	3	16	16-8 = 8	11-8 = 3
				Total	**28**	**12**
				Average	**28/5 = 5.6**	**12/5 = 2.4**

B. Shortest Job First Scheduling (NP, AT + CT)

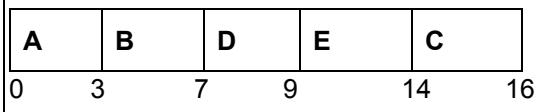


Process	AT	CT	P	FT	TT	WT
A	0	3	5	3	3-0 = 3	0-0 = 0
B	2	4	2	11	11-2 = 9	7-2 = 5
C	3	2	4	5	5-3 = 2	3-3 = 0
D	5	2	1	7	7-5 = 2	5-5 = 0
E	8	5	3	16	16-8 = 8	11-8 = 3
				Total	**24**	**8**
				Average	**24/5 = 4.8**	**8/5 = 1.6**

7cd

Yee Hui

C. Priority Scheduling (PS) (NP)



Process	AT	CT	P	FT	TT	WT
A	0	3	5	3	3	0
B	2	4	2	7	5	1
C	3	2	4	16	13	11
D	5	2	1	9	4	2
E	8	5	3	14	6	1
				Total:	**31**	**15**
				Average:	**31/5 = 6.2**	**15/5 = 3.0**

C. Priority Scheduling (PS) (P)

A	B	B	D	B	E	C	A

0 2 3 5 7 8 13 15 16

Process	AT	CT	P	FT	TT	WT
A	0	3	5	16	16	13
B	2	4 …… ……..	2	8	6	2
C	3	2	4	15	12	10
D	5	2	1	7	2	0
E	8	5	3	13	5	0
				Total:	**41**	**25**
				Average:	**41/5 = 8.2**	**25/5 = 5**

D. Shortest Remaining Time Scheduling (SRT)
H

SRT

A	A	C	D	B	B	E

0 2 3 5 7 8 11 16

Process	AT	CT	P	FT	TT	WT
A	0	3	5	3	3	0
B	2	4	2	11	9	5

C	3	2	4	5	2	0
D	5	2	1	7	2	0
E	8	5	3	16	8	3
				Total:	**24**	**8**
				Average:	**24/5 = 4.8**	**8/5 = 1.6**

e. Round Robin Scheduling (RR) (Q=3)

A	**B**	**C**	**D**	**B**	**E**	**E**

0 3 6 8 10 11 14 16

RQ: B, C, D, B,E , E

Process	**AT**	**CT**	**P**	**FT**	**TT**	**WT**
A	0	3> 0	5	3	3	0
B	2	4> 1> 0	2	11	9	5
C	3	2> 0	4	8	5	3
D	5	2> 0	1	10	5	3
E	8	5> 2> 0	3	16	8	3
		16		**Total:**	**30**	**14**
				Average:	**30/5 = 6**	**14/5 = 2.8**

8ab

Pei Xuan

Process	Arrival Time	CPU Cycle	Priority assigned
A	8	7	3
B	7	4	5 (lowest priority)
C	5	5	1 (highest priority)
D	2	3	2
E	0	6	4

Table 2

Draw a *timeline Gantt chart*, calculate the *Average Waiting Time* and *Average Turnaround Time* for each of the following algorithms.

a) First Come First Serve (FCFS)

E	D	C	B	A

0 6 9 14 18 25

Process	AT	CT	P	FT	TT	WT
A	8	7	3	25	17	10
B	7	4	5	18	11	7
C	5	5	1	14	9	4
D	2	3	2	9	7	4
E	0	6	4	6	6	0
		25		**Total Time**	**50**	**25**
				Average :	**50/5 = 10**	**25/5 = 5**

b) Shortest Job First (SJF)

E	D	B	C	A

0 6 9 13 18 25

Process	AT	CT	P	FT	TT	WT
A	8	7	3	25	17	10
B	7	4	5	13	6	2
C	5	5	1	18	13	8
D	2	3	2	9	7	4
E	0	6	4	6	6	0
Time:			**25**	**Total**	**49**	**24**
				Average :	**49/5 =9.8**	**24/5 = 4.8**

8cde

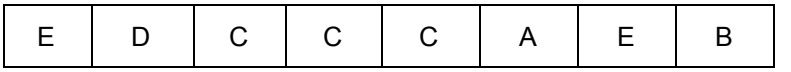
Xin Yi
https://docs.google.com/spreadsheets/d/1HToTdK_bo9lin0BtHk2X0Yeq0-a1ZzZM5IvVY0MlKqs/edit?usp=sharing
C. PS (Non-Preemptive)

E	C	D	A	B

0 6 11 14 21 25

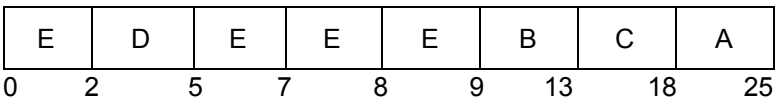
Process	AT	CT	P	FT	TT (FT-AT)	WT (FT-AT-CT)
A	8	7	3	21	13	6
B	7	4	5	25	18	14
C	5	5	1	11	6	1
D	2	3	2	14	12	9
E	0	6	4	6	6	0
Total		**25**			**55**	**30**
				Avg	**11**	**6**

C. PS (Preemptive)



Process	AT	CT	P	FT	TT (FT-AT)	WT (FT-AT-CT)
A	8	7	3	17	9	2
B	7	4	5	25	18	14
C	5	5	1	10	5	0
D	2	3	2	5	3	0
E	0	6	4	21	21	15
Total		**25**			**56**	**31**
				Avg	**11.2**	**6.2**

d. SRT (Preemptive)



Process	AT	CT	P	FT	TT (FT-AT)	WT (FT-AT-CT)
A	8	7	3	25	17	10
B	7	4	5	13	6	2
C	5	5	1	18	13	8
D	2	3	2	5	3	0
E	0	6	4	9	9	3
Total		**25**			**48**	**23**
				Avg	**9.6**	**4.6**

d. RR (Preemptive) , Q = 2

E	D	E	D	C	E	B	A	C	B	A	C	A	A

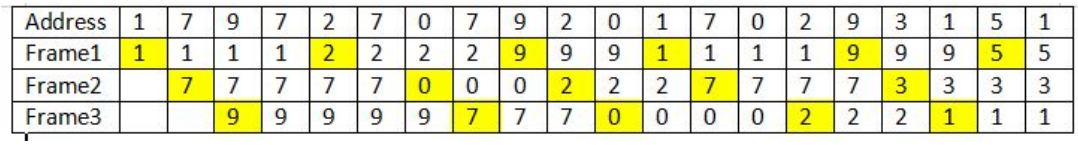
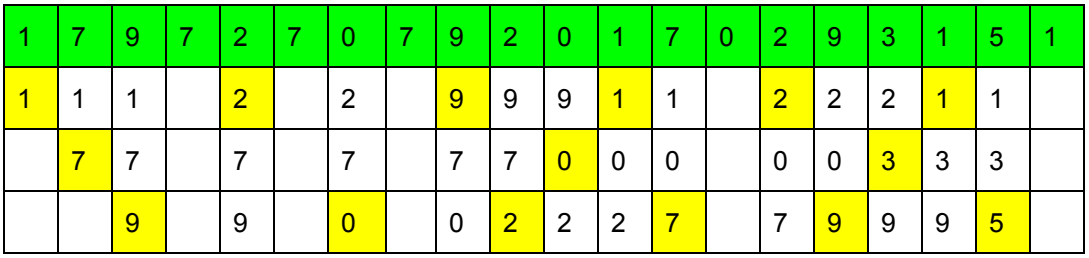
0 2 4 6 7 9 11 13 15 17 19 21 22 24 25

D -> E -> D -> C -> E -> B -> A -> C -> B -> A -> C -> A -> A

Process	AT	CT	P	FT	TT (FT-AT)	WT (FT-AT-CT)
A	8	7	3	25	17	10
B	7	4	5	19	12	8
C	5	5	1	22	17	12
D	2	3	2	7	5	2
E	0	6	4	11	11	5
Total		**25**			**62**	**37**
				Avg	**12.4**	**7.4**

Tutorial 11: Virtual Memory

1.	Jun Xian Explain demand paging memory allocation technique. List out the advantages and disadvantages of this technique.

	Demand paging keeps all pages of the frames in the secondary memory until they are required, so less I/O and memory is needed. (The required page of the program will be loaded into main memory only when it is required.) Advantages - A job is no longer constrained / limited by the size of physical memory (virtual memory) - Uses memory more efficiently (ONLY THE REQUIRED PAGES WILL BE SWAPPED INTO MEMORY) Disadvantages - Increased overhead (RESOURCES: cpu, MEMORY) caused by tables and page interrupts. - Thrashing may occur (massive swapping)., CPU will stay idle (do nothing during swapping).
2.	Kah Wei
3a	Yit Wee FIFO algorithm  No of page fault: 16 If percent = 16/20 * 100% = 80%
3b	Chia Chung LRU Algorithm  No of page fault: 15 Lf percent = 15/20x100% = 75%
4a	Jun Rong https://docs.google.com/spreadsheets/d/1PC2P4reo31rhc-8cE0lKQkThWtMGsSKj4Wsebq0ntgo/edit?usp=sharing

Address	1	7	9	7	2	7	0	7	9	2	0	1	7	0	2	9	3	1	5	1
Frame1	1	1	1	1	2	2	2	2	9	9	9	1	1	1	1	9	9	9	5	5
Frame2		7	7	7	7	7	0	0	0	2	2	2	7	7	7	7	3	3	3	3
Frame3			9	9	9	9	9	7	7	7	0	0	0	0	2	2	2	1	1	1

1	7	9	7	2	7	0	7	9	2	0	1	7	0	2	9	3	1	5	1
1	1	1		2		2		9	9	9	1	1		2	2	2	1	1	
	7	7		7		7		7	7	0	0	0		0	0	3	3	3	
		9		9		0		0	2	2	2	7		7	9	9	9	5	

FIFO																		
Address	105	580	2730	3401	1550	1800	385	12220	11700	1888	1750	1150	1112	123	1250	2760	990
Page Size	500															500	500
Page Number	0	1	5	6	3	3	0	24	23	3	3	2	2	0	2	5	1
Frame 1	0	0	0	0	3		3	3	3			2				2	2
Frame 2		1	1	1	1		0	0	0			0				5	5
Frame 3			5	5	5		5	24	24			24				24	1
Frame 4				6	6		6	6	23			23				23	23

Page fault = 11

4b

Ming Jun

LRU																		
ADDRESS	105	580	2730	3401	1550	1800	385	12220	11700	1888	1750	1150	1112	123	1250	2760	990
PAGE SIZE	500																
PAGE NUMBER	0	1	5	6	3	3	0	24	23	3	3	2	2	0	2	5	1
MEM FRAME 1	0	0	0	0	3		3	3	3			3		2		2	2
MEM FRAME 2		1	1	1	1		0	0	0			2		3		3	1
MEM FRAME 3			5	5	5		5	24	24			24		0		0	0
MEM FRAME 4				6	6		6	6	23			23		23		5	5
PAGE FAULT = PAGE MISS = PAGE NOT FOUND	12																

5

Yih Feng

Demand paging	Segmentation schemes
- Divides each incoming job into pages of **equal size** - Works well if page size = memory block size = size of disk section - **Static** memory management Advantages - Allows jobs to be allocated in non-contiguous memory locations. Memory used more efficiently; more jobs can fit. - Reduces **internal fragmentation** - Size of pages is crucial Disadvantages - Increased overhead occurs	- A segment is a logical unit - Main memory is not divided into page frames because the **size** of each segment is **different**. - Jobs are divided into a number of distinct logical units called segments - Memory is allocated **dynamically**. Advantages - Compaction - **External fragmentation** (free memory space between segment = free to use) - Secondary storage handling - Memory is allocated dynamically Disadvantages - **External fragmentation**(defragmentation is required to re-allocate the free memory space together) - Costly memory management algorithms

6

THREE (3) advantages of Virtual Memory.

T'nsam

	-Memory used is more efficiently. -Although we have 2gb of memory but we can run a program of 4gb because only part of the program is loaded into the memory. -Allows the sharing of code and data. -Eliminates external fragmentation when used with paging and eliminates internal fragmentation when used with segmentation.
6	THREE (3) disadvantages of Virtual Memory. Kai Yuan -Applications may run slower if the system is using **virtual memory**. Virtual memory uses secondary storage to temp hold program / data for execution. -Offers lesser hard drive space for your use. (to support virtual memory). -It negatively affects the overall performance of a system. / uses more resources / over head (CPU time, memory, time)

12																	
13	**LRU**																
14	ADDRESS	105	580	2730	3401	1550	1800	385	12220	11700	1888	1750	1150	1112	123	1250	2760	990
15	PAGE SIZE	500																
16	PAGE NUMBER	0	1	5	6	3	3	0	24	23	3	3	2	2	0	2	5	1
17	MEM FRAME 1	0	0	0	0	3		3	3	3			3		2		2	2
18	MEM FRAME 2		1	1	1	1		0	0	0			2		3		3	1
19	MEM FRAME 3			5	5	5		5	24	24			24		0		0	0
20	MEM FRAME 4				6	6		6	6	23			23		23		5	5
21	PAGE FAULT = PAGE MISS = PAGE NOT FOUND				12													