

**Solved By Hina Shahzadi**

**Current Paper CS401P Final**

**Term**

**VIRTUAL  
UNIVERSITY  
OF PAKISTAN**

**VU ALL SUBJECT  
ASSIGNMENT'S QUIZ'S  
GDB'S  
ALL PROJECT'S  
PAID ATTEMPTED**

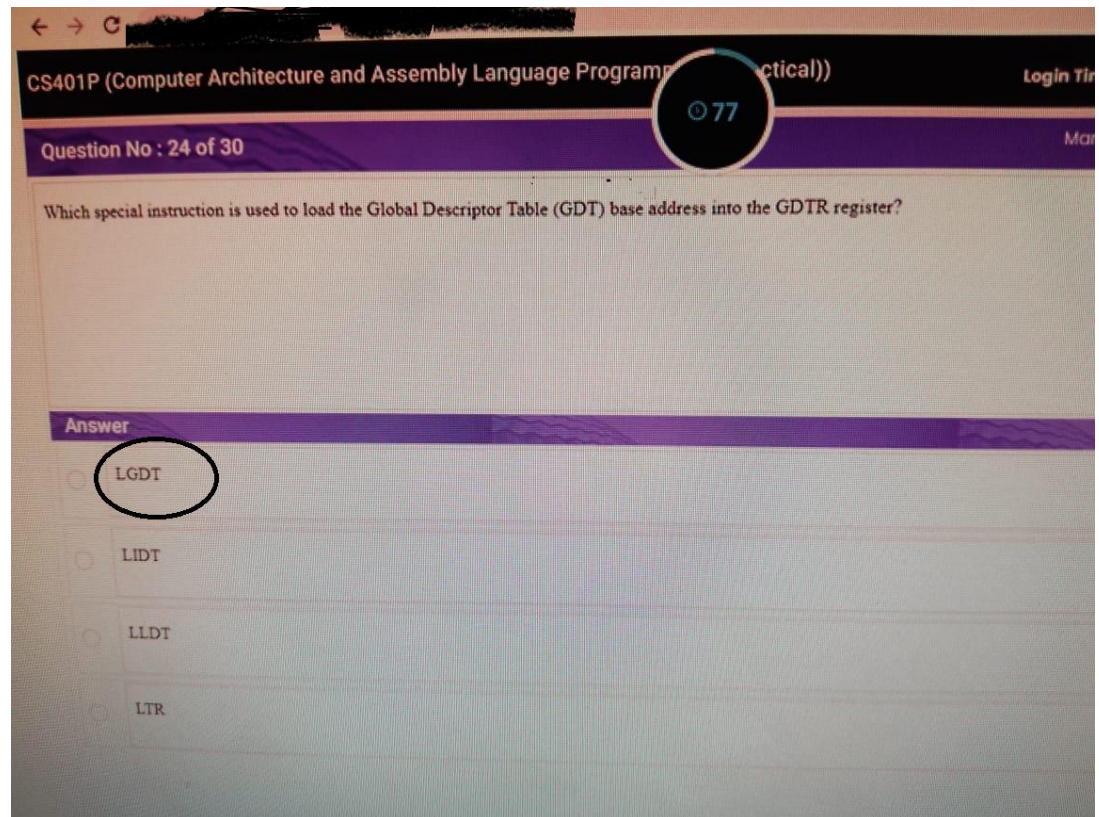
**RESULT IN SHA ALLAH  
90 TO 100%**



**HINA SHAHZADI**

**03249983800**

# Solved By Hina Shahzadi



The screenshot shows a quiz interface for a course titled "CS401P (Computer Architecture and Assembly Language Programming (Practical))". The question number is 24 of 30. The question asks: "Which special instruction is used to load the Global Descriptor Table (GDT) base address into the GDTR register?". Below the question, there is an "Answer" section with four radio button options: LGDT, LIDT, LLDI, and LTR. The "LGDT" option is selected and circled in black.

CS401P (Computer Architecture and Assembly Language Programming (Practical))

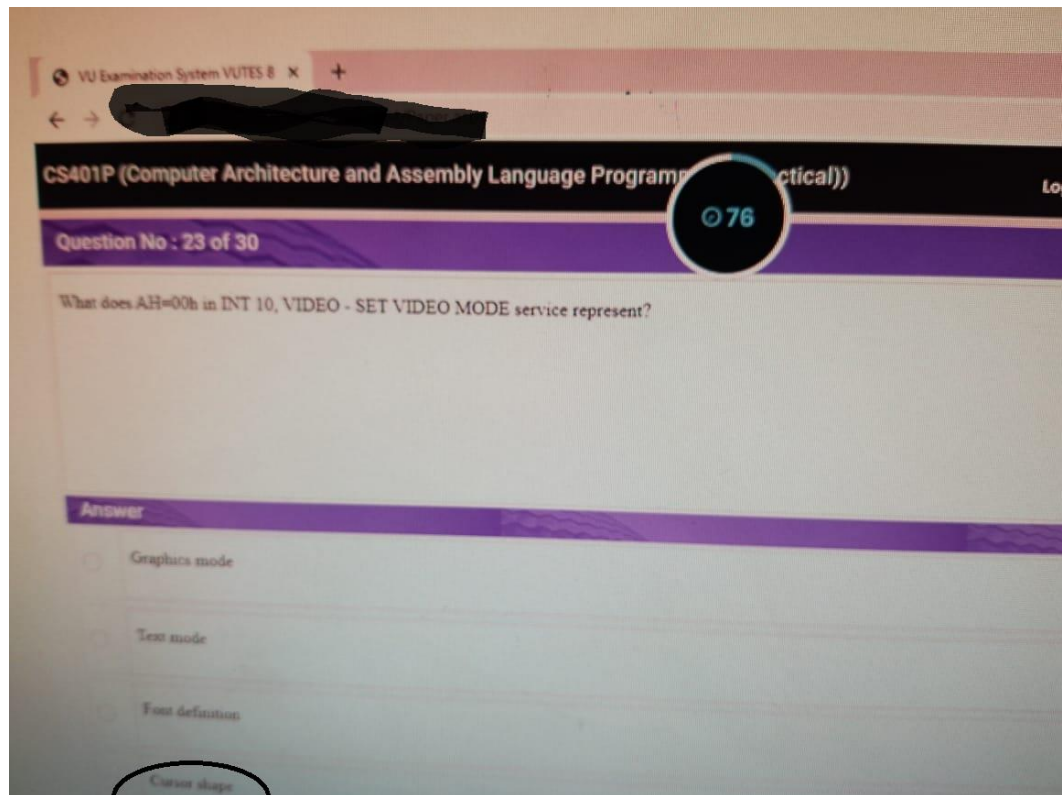
Question No : 24 of 30

Which special instruction is used to load the Global Descriptor Table (GDT) base address into the GDTR register?

Answer

- LGDT
- LIDT
- LLDI
- LTR

# Solved By Hina Shahzadi



**INT 10 - VIDEO - SET TEXT-MODE CURSOR SHAPE**

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 22 of 30

76

In the context of multitasking, what does the term "context switch" refer to?

**Answer**

- Switching between different tasks or threads
- Changing the display mode of the screen
- Allocating memory for new processes
- Processing keyboard input

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

76

Question No : 21 of 30

How many string instructions are there in Intel 8088 processor?

Answer

5

6

7

8

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 20 of 30

76

What is the purpose of the "AL" register when using INT 10h, AH=06h for scrolling the screen up in assembly language?

**Answer**

- Number of lines to scroll
- Video mode
- Background color
- Row position

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 19 of 30

75

In the context of assembly language, what is the purpose of the "CMP" instruction?

**Answer**

- Compare two values
- Copy a value
- Compute a sum
- Concatenate strings

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

75

Question No : 18 of 30

What is the purpose of DOSBox in the context of assembly language programming?

Answer

- DOSBox is an assembler for writing assembly language code.
- DOSBox is an operating system designed for assembly programming.
- DOSBox is an emulator that allows running MS-DOS and executing assembly programs.
- DOSBox is a debugger specifically built for assembly language debugging.

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

75

Question No : 17 of 30

In the context of subroutine execution, which stack-related instruction is used to remove data from the stack in 8088 as

Answer

POP

PUSH

CALL

RET

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

75

Question No : 16 of 30

What happens when a carry is generated during effective address calculation in segment wraparound?

Answer

It is added to the effective address.

It is Ignored.

It is dropped, resulting in a wraparound.

It is stored for future calculations.

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 15 of 30

75

In two's complement representation, what is the key distinction between signed and unsigned numbers?

**Answer**

The magnitude of the number

The sign of the number

Both magnitude and sign

Offset -

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 14 of 30

75

\_\_\_\_\_ operation can be used for selective bit inversion.

Answer

XOR

NOT

OR

AND

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 13 of 30

74

The Sun Spark processor belongs to a \_\_\_\_\_ processor family.

Answer

- CISC
- RISC
- Haswell
- Broadwell

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 12 of 30

The default IRQ 0 is generated \_\_\_\_\_ times per second.

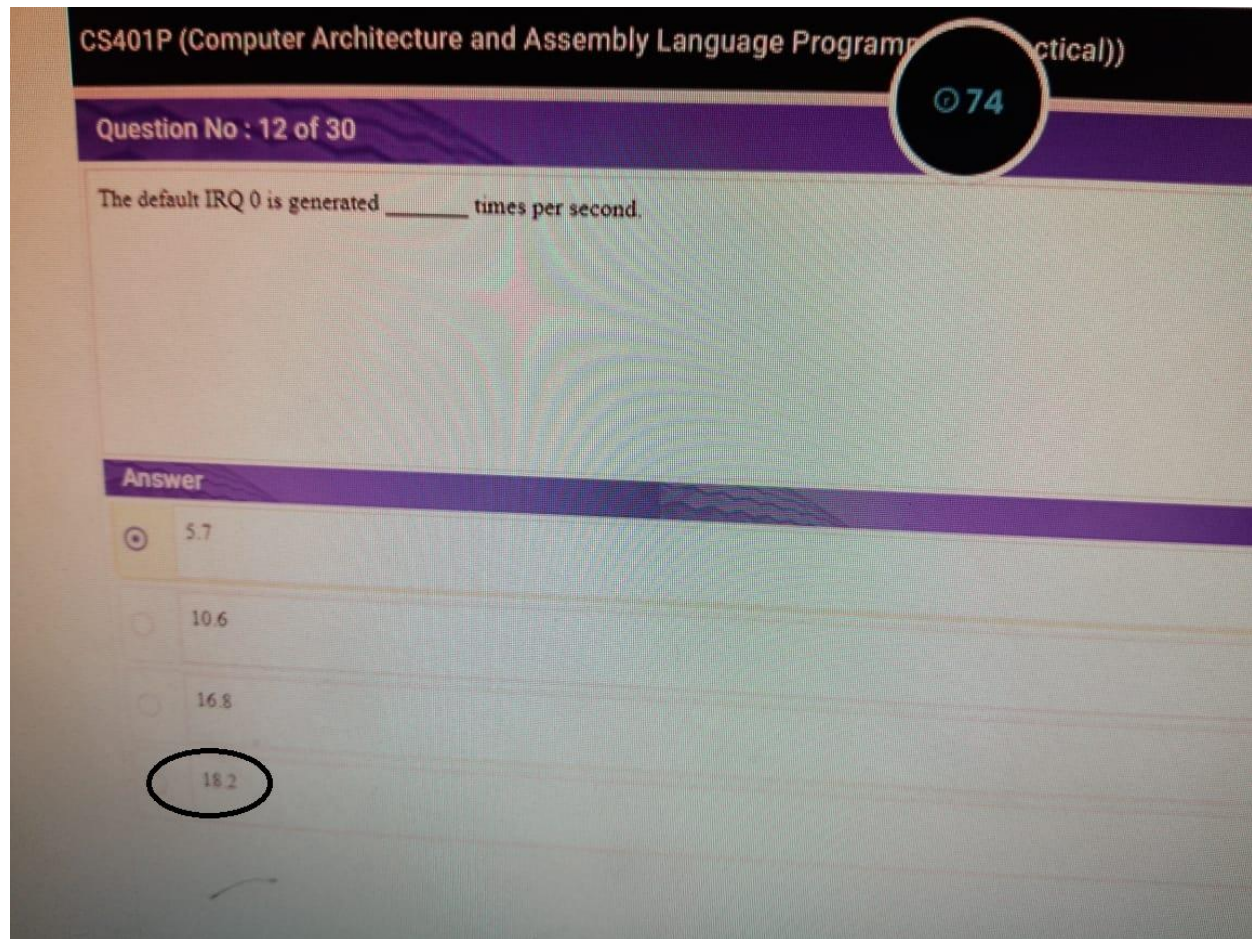
Answer

5.7

10.6

16.8

18.2

A screenshot of a quiz interface. At the top, it says 'CS401P (Computer Architecture and Assembly Language Programming (Practical))'. Below that, 'Question No : 12 of 30' is displayed. The question text is 'The default IRQ 0 is generated \_\_\_\_\_ times per second.' There are four radio button options: 5.7, 10.6, 16.8, and 18.2. The option '18.2' is circled in black. A purple bar with the word 'Answer' is above the options. A circular icon with '74' is in the top right corner.

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 11 of 30

In INT 21 - WRITE STRING TO STANDARD OUTPUT, the value of AH needs to be \_\_\_\_\_

Answer

- 06h
- 07h
- 08h
- 09h

A screenshot of a quiz interface. At the top, it says 'CS401P (Computer Architecture and Assembly Language Programming (Practical))'. Below that, 'Question No : 11 of 30' is displayed. The question text is 'In INT 21 - WRITE STRING TO STANDARD OUTPUT, the value of AH needs to be \_\_\_\_\_'. Below the question, there is a section labeled 'Answer' with four radio button options: '06h', '07h', '08h', and '09h'. The '09h' option is selected and circled in black. In the top right corner of the quiz area, there is a circular icon with the number '74' inside.

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 10 of 30

74

RCR is also known as \_\_\_\_\_.

**Answer**

- Rotate to Carry Right
- Rotate through Carry Right
- Rotate till Carry Right
- Rotate times Carry Right

A screenshot of a quiz interface. At the top, it shows the course name 'CS401P (Computer Architecture and Assembly Language Programming (Practical))' and the question number 'Question No : 10 of 30'. A circular icon with the number '74' is visible. The question asks 'RCR is also known as \_\_\_\_\_.' Below the question, there is an 'Answer' section with four radio button options: 'Rotate to Carry Right' (which is selected), 'Rotate through Carry Right', 'Rotate till Carry Right', and 'Rotate times Carry Right'. The option 'Rotate through Carry Right' is circled in black.

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 9 of 30

74

INT \_\_\_\_ with service number \_\_\_\_ can be used to print a string on the screen.

Answer

10, 11

10, 13

16, 11

16, 13

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 8 of 30

To write a Terminate and Stay Resident (TSR) program, interrupt \_\_\_\_\_ and service number \_\_\_\_\_ will be used.

Answer

21, 4C

20, 4C

21, 31

20, 31

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

074

Question No : 7 of 30

In the Interrupt Vector Table (IVT), the first 2 bytes store \_\_\_\_\_ and the later 2 bytes store \_\_\_\_\_

**Answer**

- segment of ISR, segment of ISR
- offset of ISR, offset of ISR
- segment of ISR, offset of ISR
- offset of ISR, segment of ISR

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 6 of 30

The \_\_\_\_\_ starts at head 0, track 0 and sector 11 in DOS structure.

Answer:

- first copy of FAT
- second copy of FAT
- 11'th bit of FAT
- copy of 11'th FAT

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 5 of 30

73

While using INT 21 - CREATE OR TRUNCATE FILE, we set AH = \_\_\_\_\_

Answer

3Ah

3Bh

3Ch

3Dh

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 4 of 30

73

If we turn on the least significant bit of CR0, the processor switches to \_\_\_\_\_ mode.

Answer

- secure
- public
- private
- protected

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

Question No : 3 of 30

73

The VESA VBE 2.0 mode allows \_\_\_\_\_ access to the video memory.

**Answer**

- protected
- private
- direct
- indirect

# Solved By Hina Shahzadi

Question No : 2 of 30 73

\_\_\_\_\_ instruction is used to get VESA- SuperVGA Mode Information.

**Answer**

- `mov ax, 0x4f01`
- `mov ax, 0x4f02`
- `mov bx, 0x4f01`
- `mov bx, 0x4f02`

# Solved By Hina Shahzadi

CS401P (Computer Architecture and Assembly Language Programming (Practical))

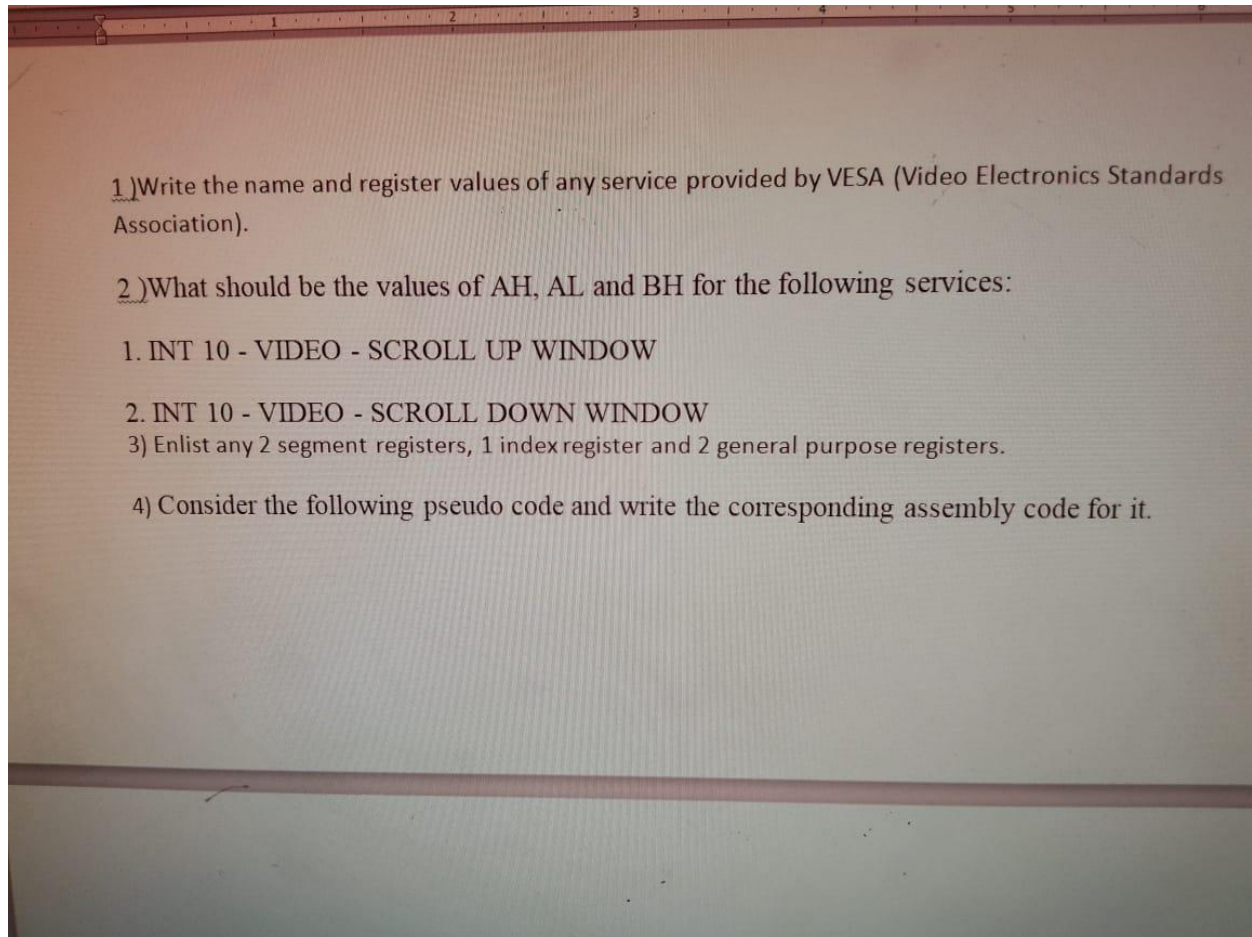
Question No : 1 of 30

\_\_\_\_\_ processor uses both big-endian and little-endian notations based on requirements.

Answer

- Intel
- Motorola
- SPARC
- AMD

# Solved By Hina Shahzadi



Q:1

```
INT 10 - VESA - Get SuperVGA Information
AX = 4F00h
ES:DI -> buffer for SuperVGA information
Return:
AL = 4Fh if function supported
AH = status
INT 10 - VESA - Get SuperVGA Mode Information
AX = 4F01h
CX = SuperVGA video mode
ES:DI -> 256-byte buffer for mode information
Return:
AL = 4Fh if function supported
AH = status
ES:DI filled if no error
INT 10 - VESA - Set VESA Video Mode
AX = 4F02h
```

Q2:

# Solved By Hina Shahzadi

## **INT 10 - VIDEO - SCROLL UP WINDOW**

AH = 06h

AL = number of lines by which to scroll up (00h = clear entire window)

BH = attribute used to write blank lines at bottom of window

CH, CL = row, column of window's upper left corner

DH, DL = row, column of window's lower right corner

## **INT 10 - VIDEO - SCROLL DOWN WINDOW**

AH = 07h

AL = number of lines by which to scroll down (00h=clear entire window)

BH = attribute used to write blank lines at top of window

CH, CL = row, column of window's upper left corner

DH, DL = row, column of window's lower right corner

Q3:

# **Solved By Hina Shahzadi**

## **Segment Registers:**

1. CS (Code Segment)
2. DS (Data Segment)

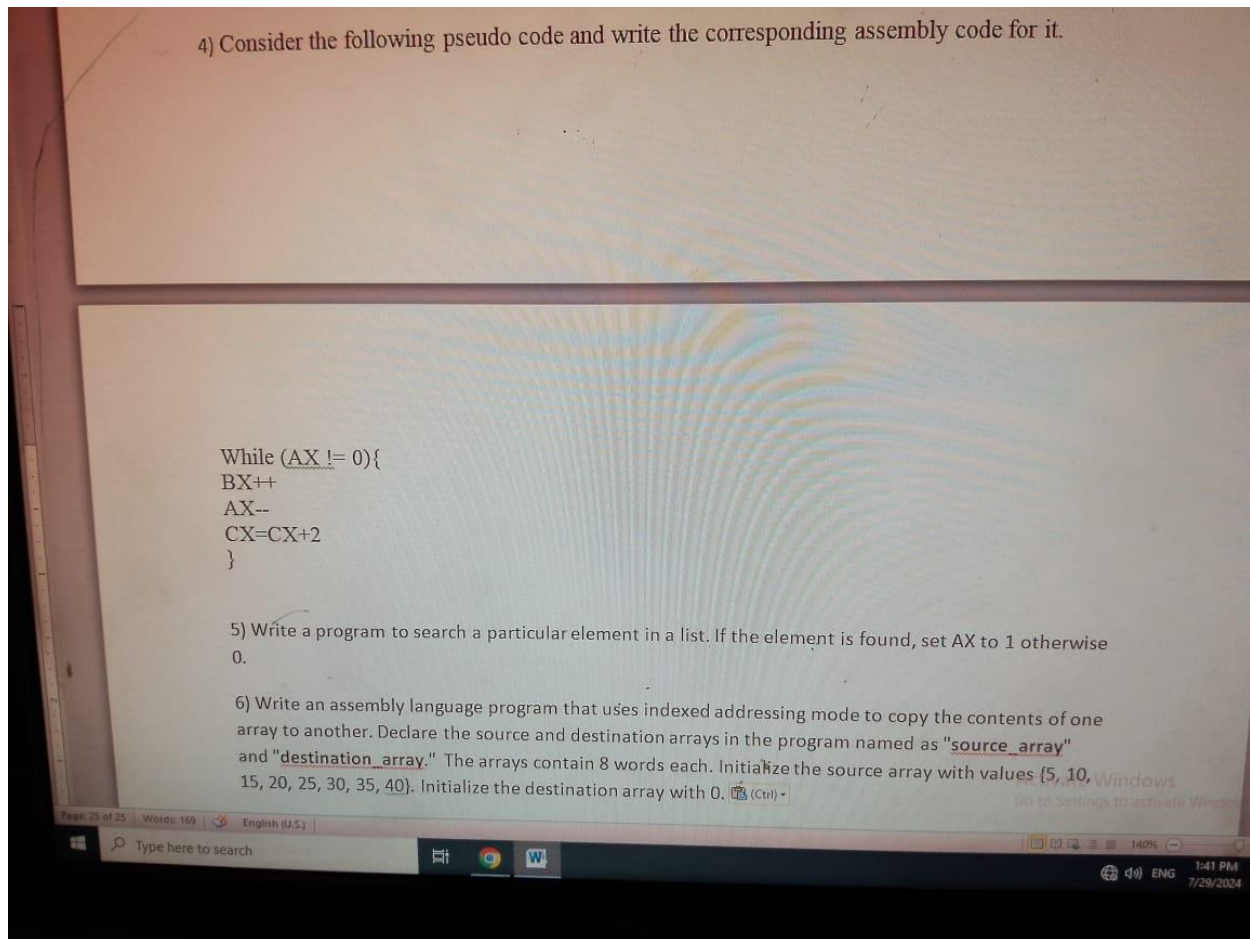
## **Index Register:**

1. SI (Source Index)

## **General Purpose Registers:**

1. AX (Accumulator Register)
2. BX (Base Register)

# Solved By Hina Shahzadi



Q 4:

Assume AX, BX, and CX are already initialized

start\_loop:

```
cmp AX, 0 ; Compare AX with 0
```

```
je end_loop ; If AX is zero, jump to end_loop
```

```
inc BX ; Increment BX
```

```
dec AX ; Decrement AX
```

```
add CX, 2 ; Add 2 to CX
```

# Solved By Hina Shahzadi

```
jmp start_loop ; Jump back to start_loop
```

```
end_loop:
```

```
; Loop ends here
```

Q 5:

```
section .data
```

```
list db 10, 20, 30, 40, 50 ; Define the list of elements
```

```
list_size equ 5 ; Define the size of the list
```

```
element_to_find db 30 ; Define the element to search for
```

```
section .bss
```

```
found resb 1 ; Define a variable to store the found flag
```

```
section .text
```

```
global _start
```

```
_start:
```

```
mov cx, list_size ; Set CX to the size of the list
```

```
lea si, [list] ; Load the address of the list into SI
```

```
mov al, [element_to_find] ; Load the element to find into AL
```

```
xor ax, ax ; Clear AX, initially assume not found
```

```
xor bx, bx ; Clear BX, this will be our index
```

```
search_loop:
```

```
cmp bx, cx ; Compare index with list size
```

```
je not_found
```

**Q 6:**

# Solved By Hina Shahzadi

section .data

source\_array dw 5, 10, 15, 20, 25, 30, 35, 40 ; Define and initialize source array

destination\_array dw 0, 0, 0, 0, 0, 0, 0, 0 ; Define and initialize destination array

section .bss

; No uninitialized data required for this program

section .text

global \_start

\_start:

mov si, 0 ; Initialize SI to 0 (source index)

mov di, 0 ; Initialize DI to 0 (destination index)

mov cx, 8 ; Set CX to 8 (number of elements)

copy\_loop:

mov ax, [source\_array + si] ; Load word from source array into AX

mov [destination\_array + di], ax ; Store word from AX to destination array

add si, 2 ; Increment source index by 2 (word size)

add di, 2 ; Increment destination index by 2 (word size)

loop copy\_loop ; Decrement CX and repeat until CX is 0

end\_program:

; Exit the program (for illustration, typically you'd do this via a system call)

mov ah, 0x4c ; DOS interrupt for program exit

int 0x21 ; Call DOS interrupt