Description: https://lh4.googleusercontent.com/bpfGlrXcwL2InVVTEmmoD_RjmplIRjQ_qTei0xro8dNAn1_LWfEzxzwbA6Ph-12qiEAOhlUhNU8lPPq-x6jsbRG0r59XzrcxgbNNAZbAQvOMofzDXY8UgF2dYyKOfh8XmWHIBgv7CYrN-Z6srCJdzCEa-Cl_sPh8A67eEpwJKJBpnFroi2ANV9WYy4jU1nQkymy8v_s

**“lab 11”**

**COURSE :**

**CALD**

**SUBMITTED TO :**

**Sir shoaib**

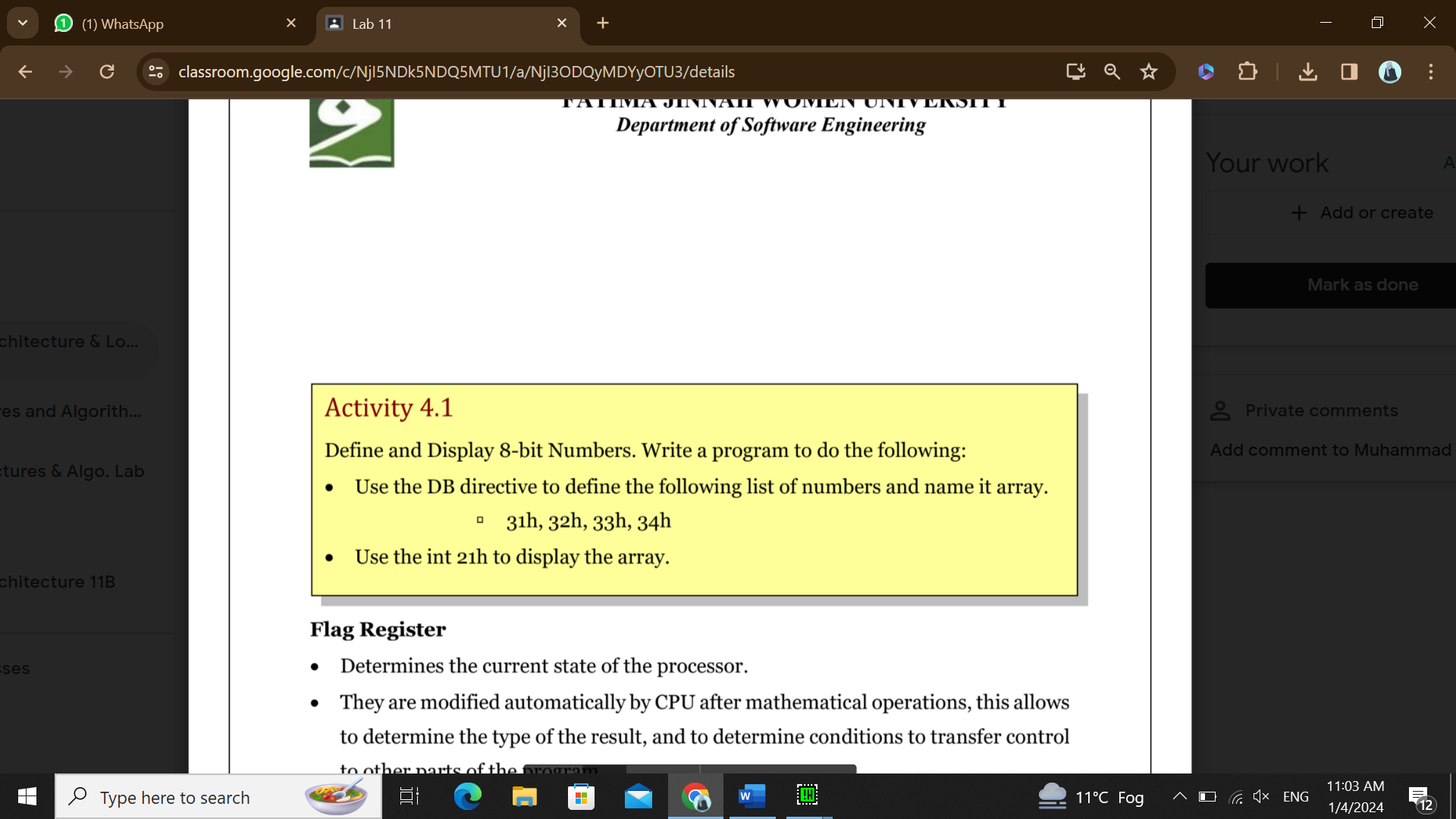
**SUBMITTED BY :**

**Rabia Batool (2022-BSE-064)**

**SECTION :**

**B**

**Activity**



**Code**

.model small

.stack 100h

.data

array db 31h, 32h, 33h, 34h ; Define the array of 4 hexadecimal numbers

.code

main proc

mov ax, @data

mov ds, ax ; Load the data segment

mov dl, [array] ; Access and load the first element of the array

mov ah, 2 ; AH = 2 for displaying a character

int 21h ; Display the character

mov dl, [array + 1] ; Access and load the second element of the array

int 21h

mov dl, [array + 2] ; Access and load the third element of the array

int 21h

mov dl, [array + 3] ; Access and load the fourth element of the array

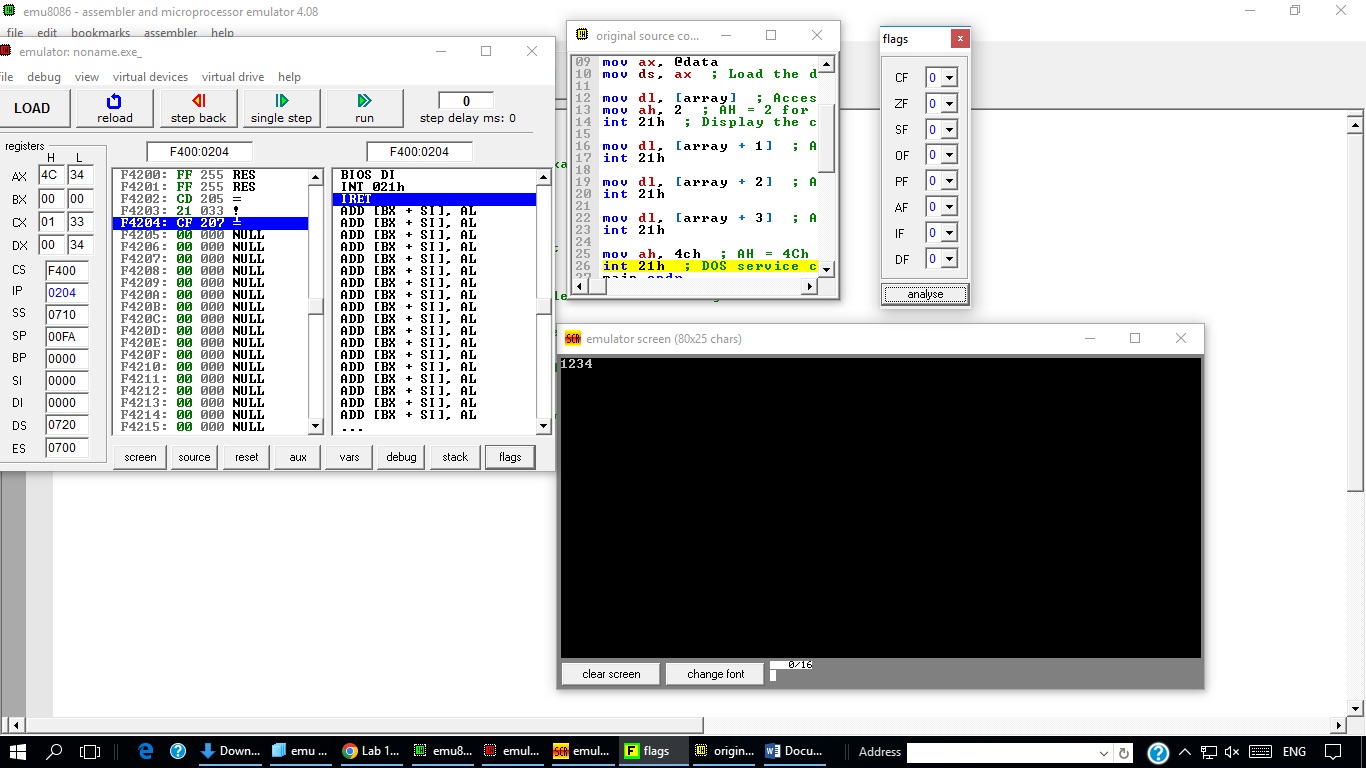
int 21h

mov ah, 4ch ; AH = 4Ch for terminating the program

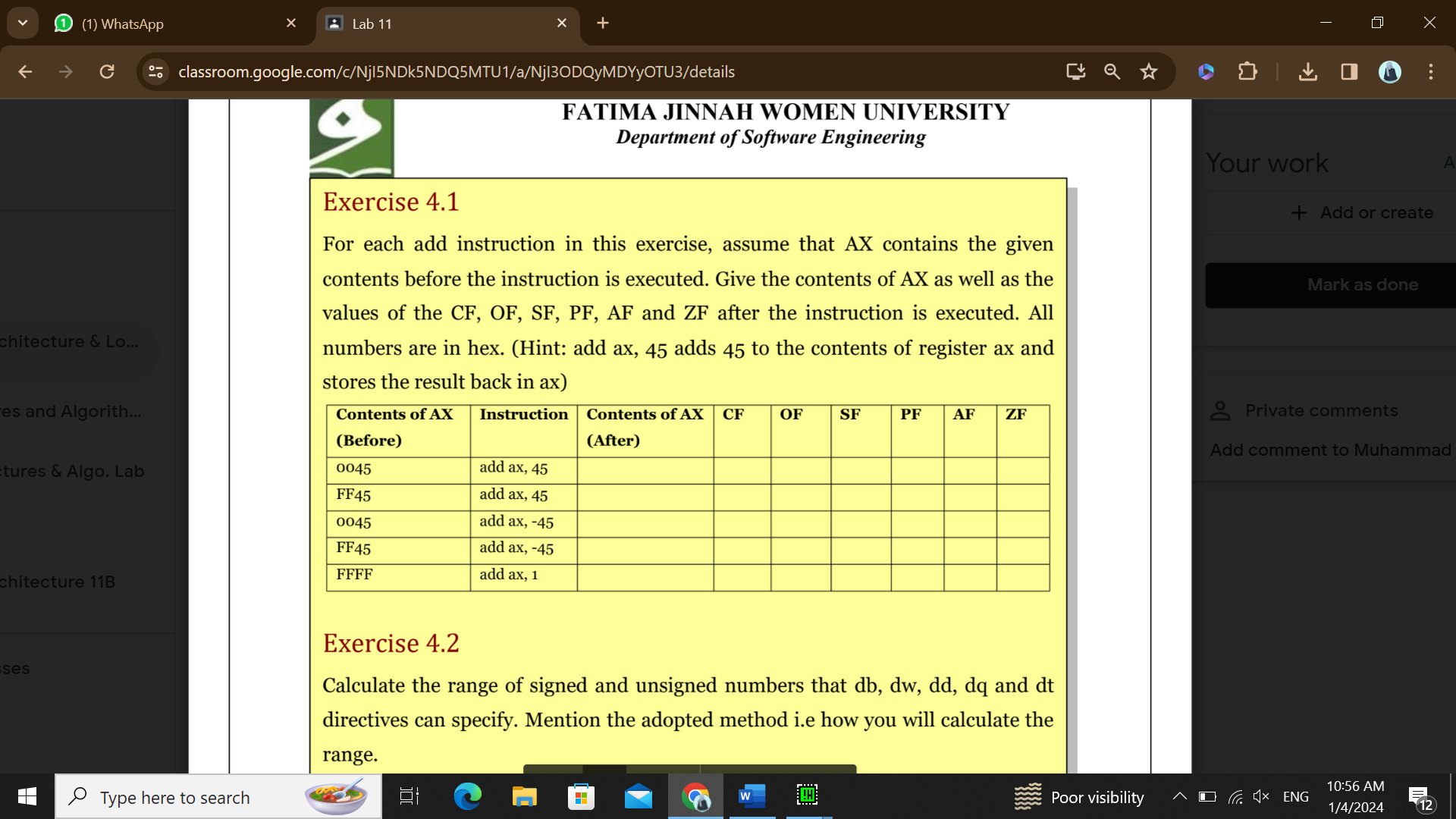
int 21h ; DOS service call to terminate the program

main endp

end main



**Exercise**



**Code for add ax, 45:**

.code

main proc

mov ax, 0045

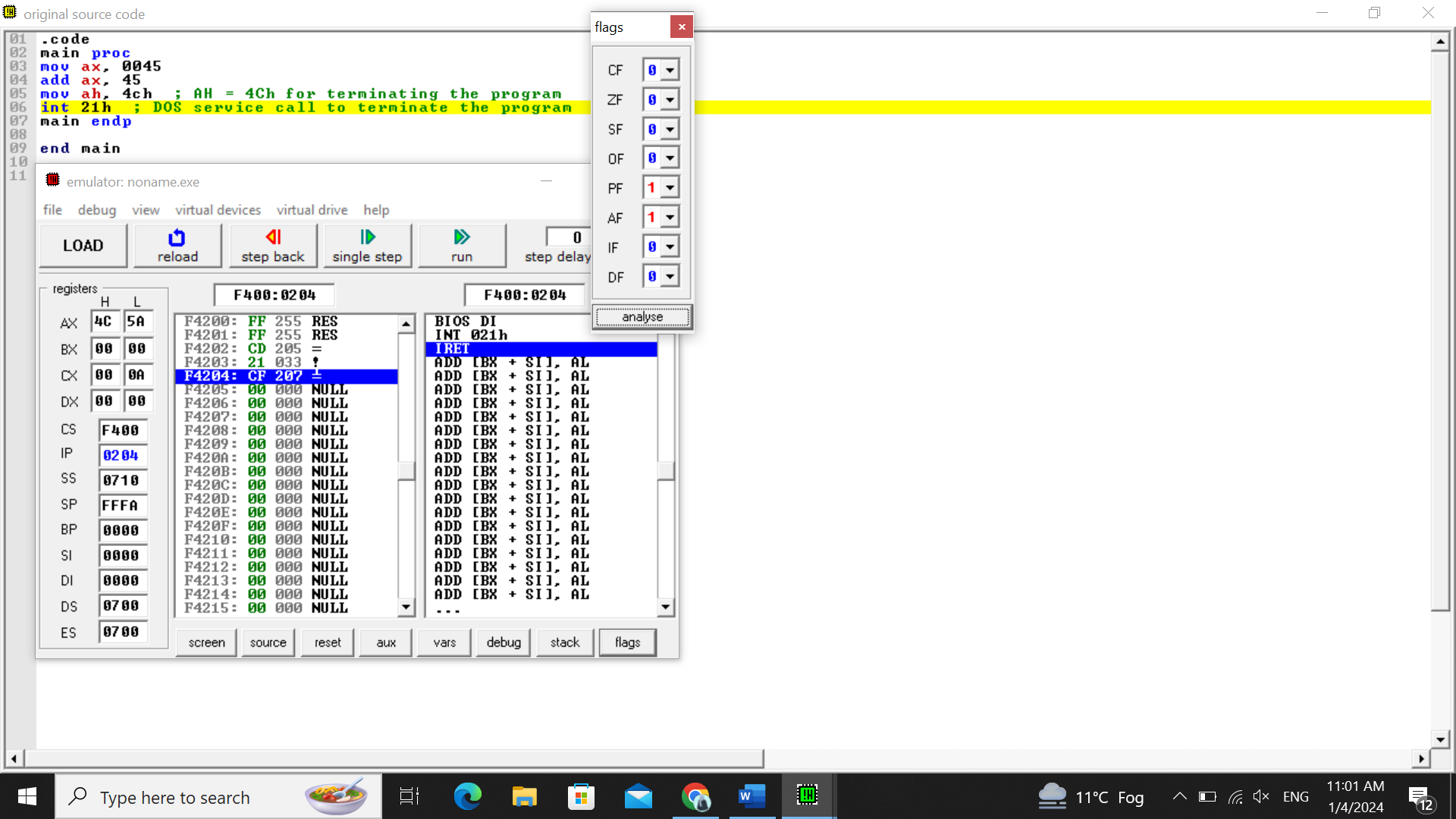
add ax, 45

mov ah, 4ch ; AH = 4Ch for terminating the program

int 21h ; DOS service call to terminate the program

main endp

end main



**Code for add ax, -45:**

.code

main proc

mov ax, 0045

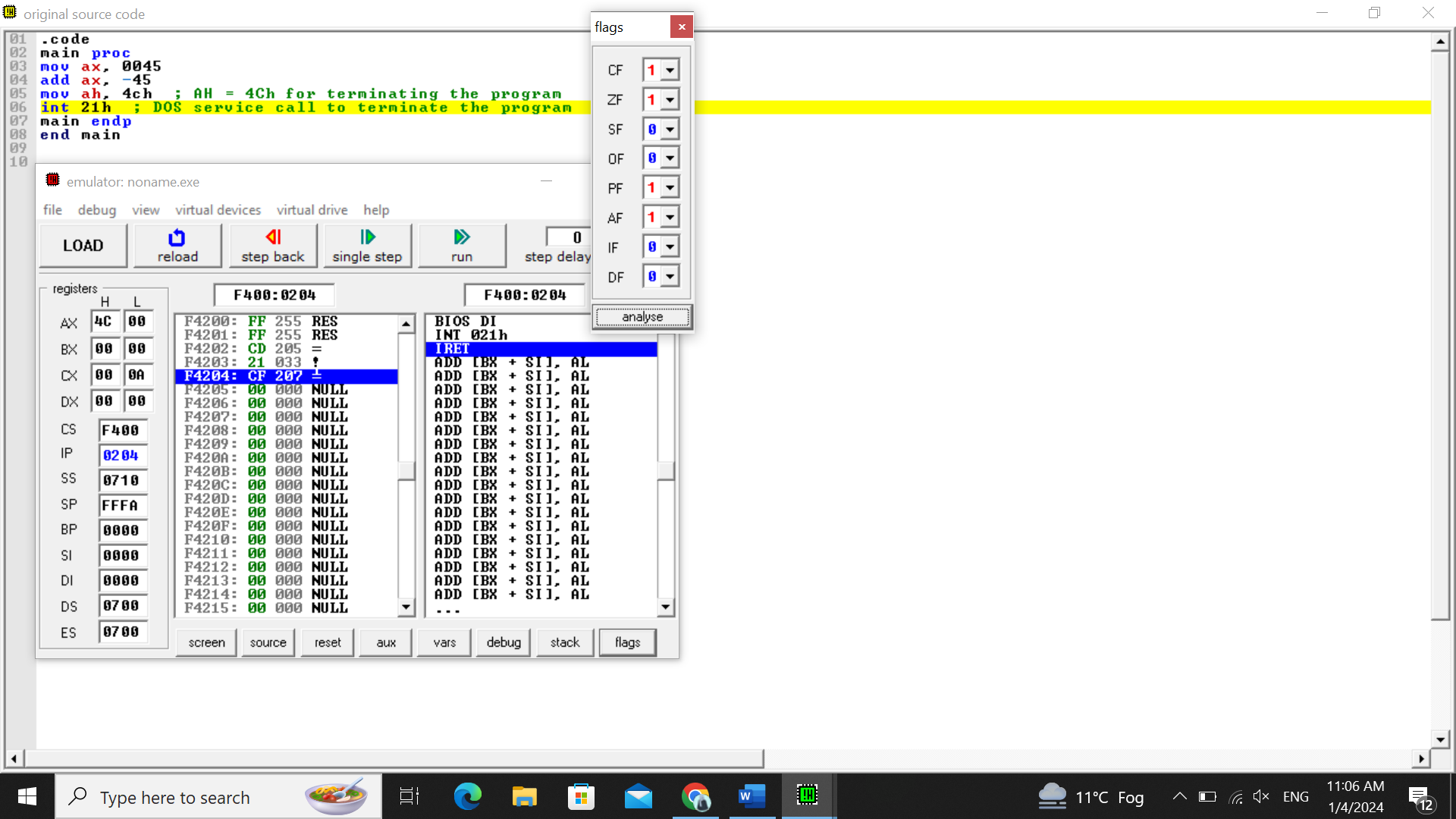
add ax, -45

mov ah, 4ch ; AH = 4Ch for terminating the program

int 21h ; DOS service call to terminate the program

main endp

end main



**PROGRAM-03**

**PROGRAM-04**

A screenshot of a computer screen

Description automatically generated

**Code:**

.model small

.stack 100h

.data

array db 36h, 37h, 39h, 40h ; Define the array of 4 hexadecimal numbers

.code

main proc

mov ax, @data

mov ds, ax ; Load the data segment

mov dl, [array + 3] ; Access and load the first element of the array

mov ah, 2 ; AH = 2 for displaying a character

int 21h ; Display the character

mov dl, [array + 2] ; Access and load the second element of the array

int 21h

mov dl, [array + 1] ; Access and load the third element of the array

int 21h

mov dl, [array] ; Access and load the fourth element of the array

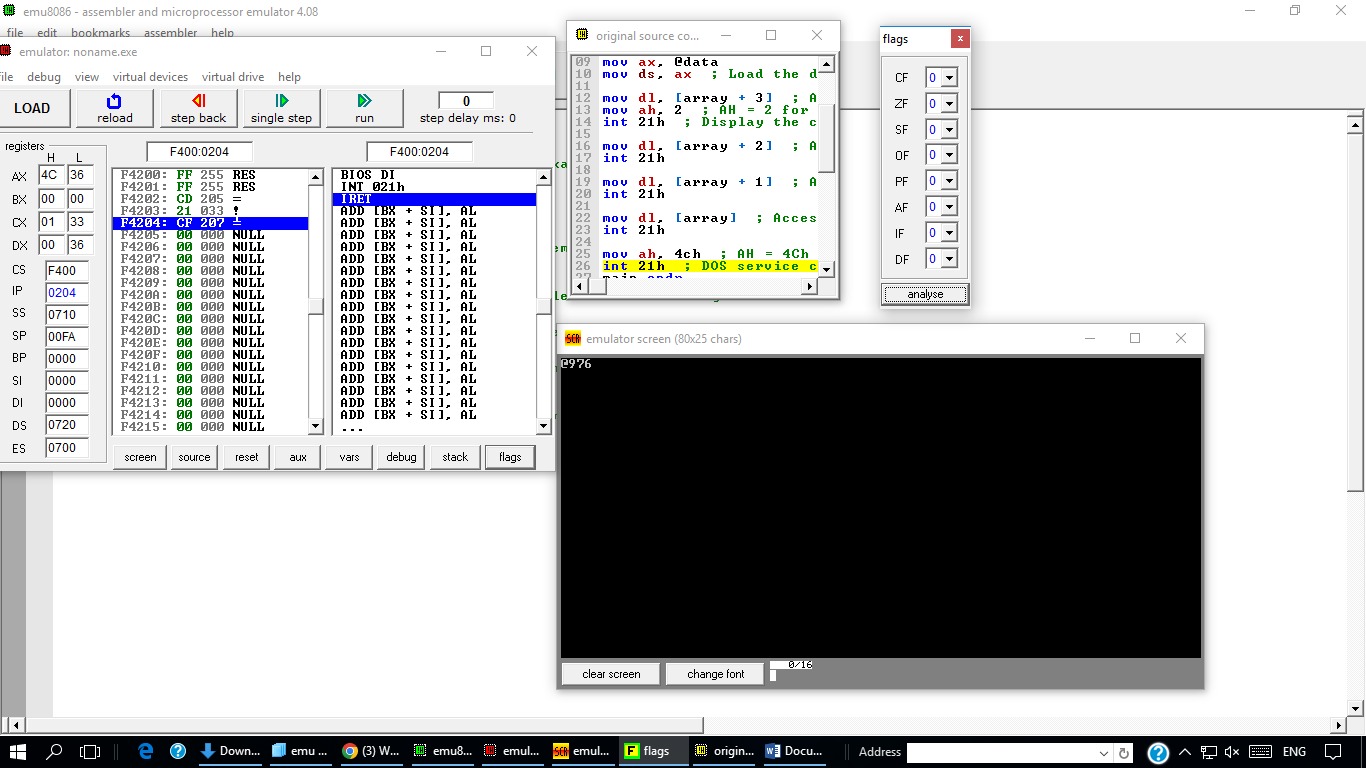
int 21h

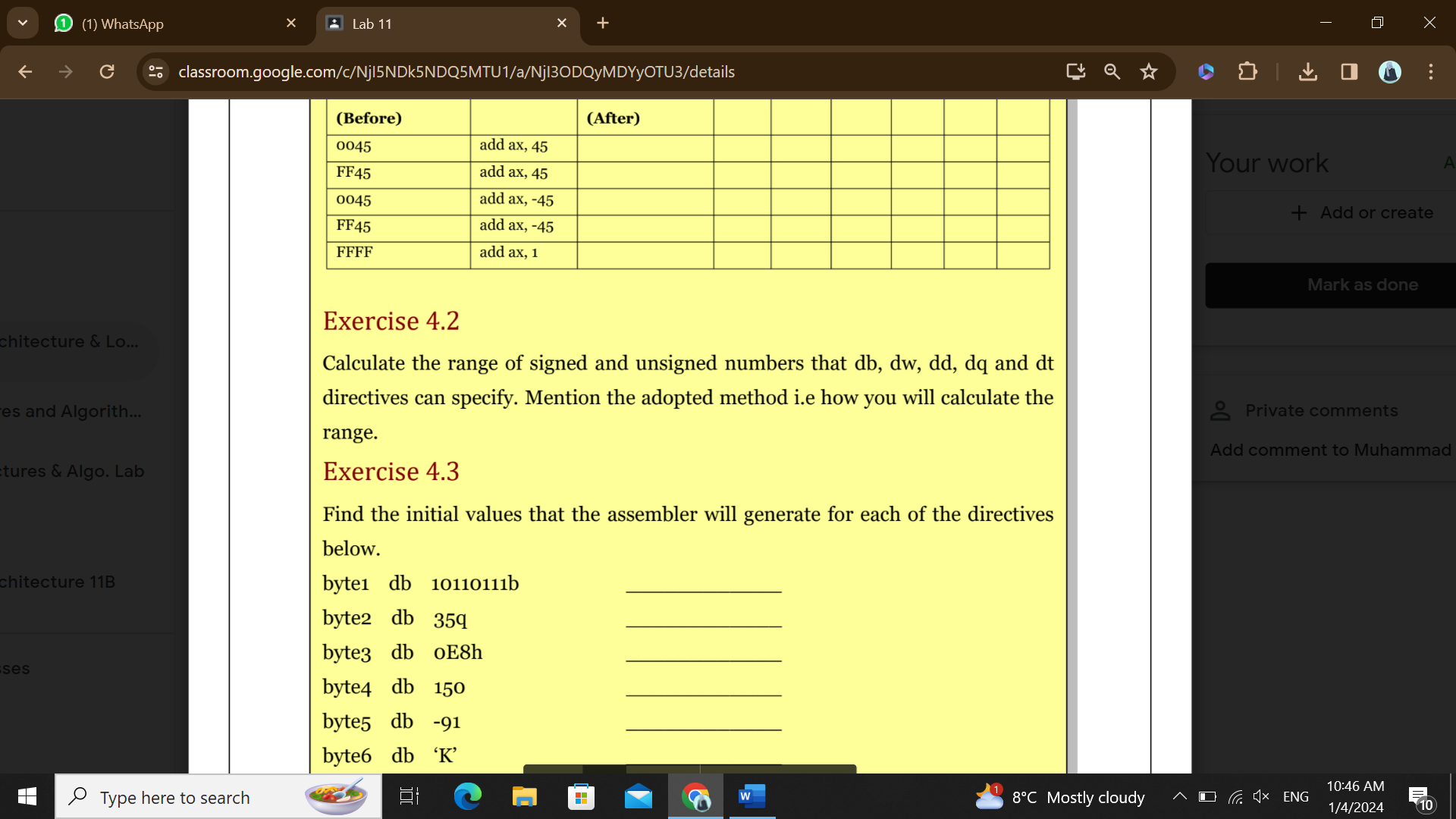
mov ah, 4ch ; AH = 4Ch for terminating the program

int 21h ; DOS service call to terminate the program

main endp

end main





**db (define byte):**

* **Signed Range:** -128 to 127
* **Unsigned Range:** 0 to 255
* **Calculation method**: For signed, it's -2^(n-1) to 2^(n-1)-1, and for unsigned, it's 0 to 2^n-1, where n is the number of bits (8 bits in this case).

**dw (define word):**

* **Signed Range:** -32,768 to 32,767
* **Unsigned Range:** 0 to 65,535
* **Calculation method:** Similar to db, but with 16 bits.

**dd (define doubleword):**

* **Signed Range:** -2,147,483,648 to 2,147,483,647
* **Unsigned Range:** 0 to 4,294,967,295
* **Calculation method:** Similar to db but with 32 bits.

**dq (define quadword):**

* **Signed Range:** -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
* **Unsigned Range:** 0 to 18,446,744,073,709,551,615
* **Calculation method:** Similar to db but with 64 bits.

**dt (define ten-byte):**

* **Signed Range:** Approximately -1.8 x 10^308 to 1.8 x 10^308
* **Unsigned Range:** 0 to approximately 3.6 x 10^308
* **Calculation method:** The dt directive is not as standard as the others, and the range can vary depending on the floating-point format used. In general, it's designed to handle floating-point values with higher precision.

