Microprocessors Final

A

NAME & SURNAME :

NUMBER:

SIGN:

İmportant note! Please write your answers legibly. Illegible or misspelled statements will not be accepted.

1. Write a program that takes the square of each element of a "`numbers`" array with a total of 4 elements, each element of which is byte in size, and adds it to the "`square`" array. (Let each element of the square array be word size.) ***Fill in the blanks in the program. (30pts)***

```
data segment
    numbers db 2,5,4,8
    square dw 4 dup(?)
ends
stack segment
    dw   128  dup(0)
ends
code segment
start:
    mov ax, data
    mov ds, ax
    lea bx, numbers
    lea bp, square
    mov si, 0
    mov di, 0
    mov cx, 4
goTo:
    mov al, [bx+si]
    mul al
    mov [bp+di], al
    inc si
    inc di, 2
    loop goTo
mov ax, 4c00h
int 21h
ends
end start
```

2. Write a program that finds how many 1 values are in the variable `numbers` db 00111000b. Store the number of 1 values found in a variable named `numberOne`. ***Rewrite the program by correcting the errors.* (20pts)**

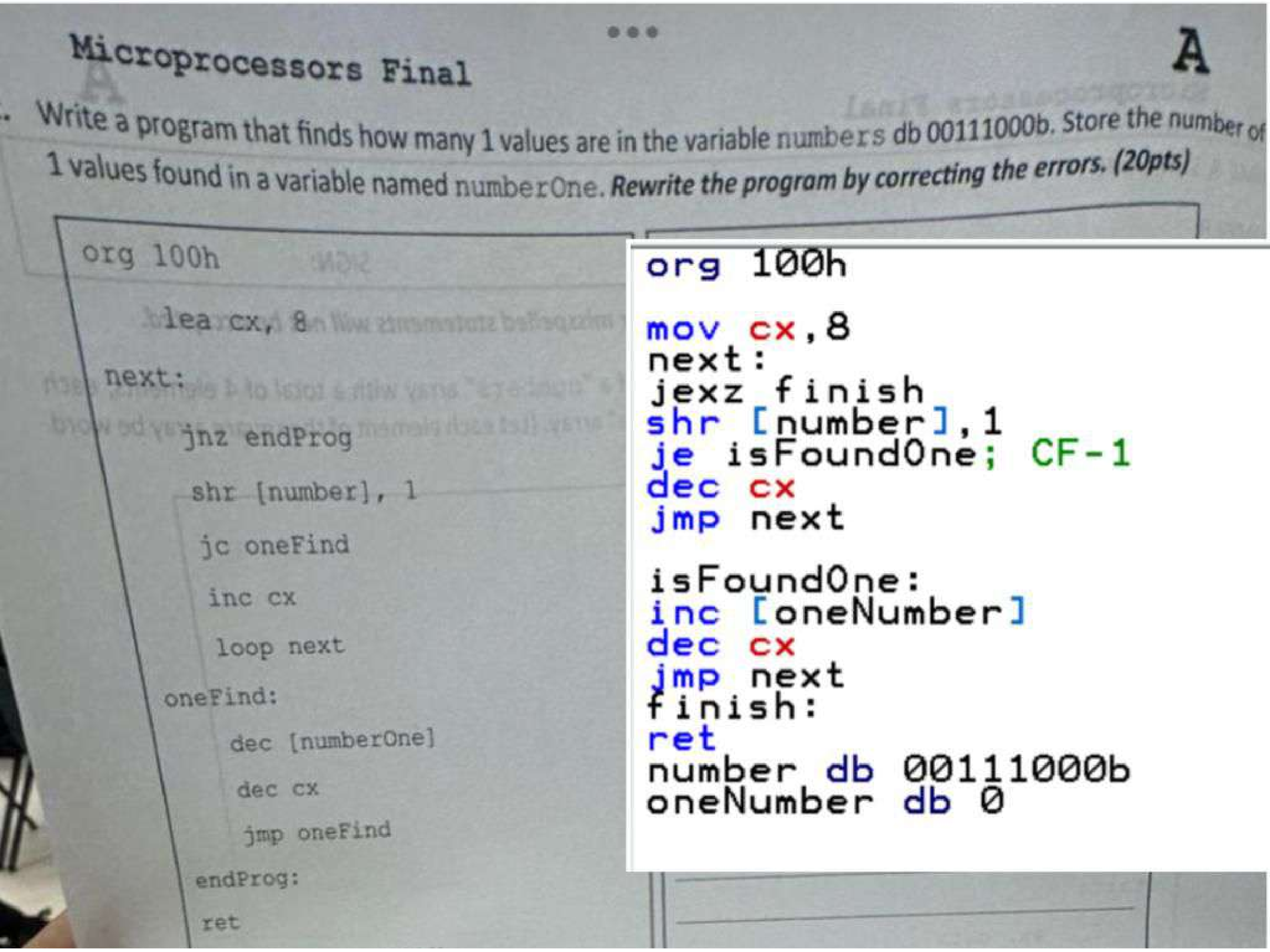
```
org 100h

 lea cx, 8

next:

     jnz endProg

     shr [number], 1

      jc oneFind

      inc cx

       loop next

oneFind:

        dec [numberOne]

        dec cx

        jmp oneFind

endProg:

ret

number db 00111000b

numberOne db 0
```

```
Org 100h
 mov cx, 8
 mov al, [numbers]
 mov [numberOne], 0
 next:
   SHR al, 1
     jnc skipOne
     inc [numberOne]
 skipOne:
        loop next
 endProg:
   mov ah, 4ch
     int, 21h
 numbers db 00111 0000b
 numberOne db 0
 End
```

3. Write a program that converts a given ascii format number '47' to packed bcd format using MACRO. **(20pts).** *You can also use the page as two columns if needed.*

```
Org 100h

Pack_BCD Macro
Sub AH, '0'
Sub AL, '0'
SHL AH, 4
ADD AL, AH   ; Result will
RET            be in AL.


ENDM
```

```
mov AX, '47'

Pack_BCD

INT 20H

END
```



Microprocessors Final

A

. Write a program that finds how many 1 values are in the variable numbers db 00111000b. Store the number of 1 values found in a variable named numberOne. ***Rewrite the program by correcting the errors. (20pts)***

```
org 100h

    lea cx, 8

 next:

     jnz endProg

      shr [number], 1

      jc oneFind

       inc cx

       loop next

   oneFind:

        dec [numberOne]

        dec cx

        jmp oneFind

   endProg:

   ret
```

```
org 100h

mov cx,8
next:
jexz finish
shr [number],1
je isFoundOne; CF-1
dec cx
jmp next

isFoundOne:
inc [oneNumber]
dec cx
jmp next
finish:
ret
number db 00111000b
oneNumber db 0
```