

1. (25 pts) \_\_\_\_\_

Name: \_\_\_\_\_

2. (30 pts) \_\_\_\_\_

This examination has five problems, each with approximate points indicated. The maximum possible total is 150 points.

3. (40 pts) \_\_\_\_\_

Show all your work. You may use the ECE 291 Laboratory Notes, one page of your own notes, and an unnecessary calculator.

4. (30 pts) \_\_\_\_\_

5. (25 pts) \_\_\_\_\_

**Problem 1.** (25 pts) Write short sequences of instructions to accomplish each task. You may use registers `AX`, `BX`, `CX`, `DX` without saving their contents. No comments are necessary.

(a) Memory bytes labeled `P` and `Q` contain unsigned values  $p$  and  $q$ , respectively. Place the smaller of  $p$  and  $q$  into register `DH`.

---

---

---

---

---

---

(b) Memory words labeled `S` and `T` contain values  $s$  and  $t$ , respectively, in two's complement representation. If the product  $st$  is divisible by 3 (with zero remainder), then set memory byte labeled `F` to 1; otherwise, set this byte to 0. Assume `F` is declared with `RESB`, and overflow does not occur.

---

---

---

---

---

---

---

---

---

---

---

---



**Problem 3.** (40 pts) You have used Turbo Debugger to suspend the execution of a program just after a `call Subr` instruction, which is three bytes long.

```
#exam1#Subr
cs:04B5▶60          pusha                ax 1776
cs:04B6 81FB0800    cmp      bx,8        bx 0002
cs:04BA 7739        ja      #exam1#.Exit cx 1406
#exam1#.Loop       dx 1308
cs:04BC 8A879401    mov     al,[bx+0194]  si 1492
cs:04C0 FEC8        dec     al           di 1867
cs:04C2 88879A01    mov     [bx+019A],al bp 0000
cs:04C6 4B         dec     bx           sp 01FE
cs:04C7 81FB0000    cmp     bx,0         ds 0F35
cs:04CB 7DEF        jge    #exam1#.Loop es 0F05
cs:04CD BA9A01      mov     dx,019A      ss 0F15
cs:04D0 E83650      call   dspmsg        cs 0F35
cs:04D3 61         popa                ip 04B5
cs:04D4 C3         ret

ds:0190 6A 6F 79 24 43 46 42 24 joy$CFB$      ss:0200 1945
ds:0198 50 41 43 4B 45 52 53 24 PACKERS$      ss:01FE 13A2
```

(a) Express each answer in hexadecimal. If there is insufficient information, write “Unknown.”

(i) The value of the word at `ds:0198` at this time:

(ii) The 20-bit linear address of next instruction to be executed:

(iii) The offset of `.Exit`, the target of the `ja` instruction (the second byte is the displacement in two’s complement representation):

(iv) The offset in the Code Segment of the most recently executed `call Subr` instruction:

(b) Now suppose the subroutine is executed starting from this point.

(i) Determine the values of the flag bits immediately after the first execution of `cmp bx,8`:

SF = \_\_\_\_\_ CF = \_\_\_\_\_ ZF = \_\_\_\_\_ OF = \_\_\_\_\_

(ii) Find the value of the word at `ss:01FC` after execution of `cmp bx,8`: \_\_\_\_\_ (hex)

(iii) How many times is the `dec bx` instruction executed in this call to `Subr`? \_\_\_\_\_

(iv) What does this subroutine print out with the call to `dspmsg`? \_\_\_\_\_

**Problem 4.** (30 pts) The following subroutine `Prime` should determine whether an input number  $n$  is prime by trying all possible divisors from 2 to  $n-1$ :

Input (AX) = input integer  $n$ ; assumes  $n \geq 2$

Output (CL) = 1 if  $n$  is prime, 0 if  $n$  is not prime

Although there are no syntax errors, this subroutine has seven mistakes. Identify six mistakes; for each, explain the mistake and specify a correction, using the line numbers.

```

1  SaveAX    RESB    1    ; Save input number n
2  Prime:
3      PUSHA
4      MOV    [SaveAX], AX
5      MOV    CL, 1      ; Assume input n is prime
6      MOV    DI, 2      ; Trial divisor
7  .Loop:
8      MOV    AX, SaveAX
9      CMP    DI, AX      ; If trial divisor > n
10     JA     .Exit      ; then n is prime
11     DIV    DI
12     CMP    DX, 0      ; If remainder is zero
13     JNE    .Loop
14     MOV    CL, 0      ; then n is not a prime
15     RET
16  .Exit:
17     POPA
18     RET

```

1. \_\_\_\_\_

\_\_\_\_\_

2. \_\_\_\_\_

\_\_\_\_\_

3. \_\_\_\_\_

\_\_\_\_\_

4. \_\_\_\_\_

\_\_\_\_\_

5. \_\_\_\_\_

\_\_\_\_\_

6. \_\_\_\_\_

\_\_\_\_\_

