

Problem 3. (30 pts) Here is a variation of the `QSort` subroutine from Machine Problem #3. This subroutine is called with the beginning offset `bo` in `SI` and the ending offset `eo` in `DI`.

Offset in Code Segment		Offset in Stack Segment
	<code>QSort:</code>	
83D1	<code>PUSH SI ; Save bo on stack</code>	
83D2	<code>MOV BX, PList ; Set up inputs</code>	01E0 _____
83D5	<code>MOV [BX], SI ; for call</code>	
83D7	<code>MOV [BX+2], DI</code>	01E2 _____
83DA	<code>CALL Partition ; to Partition</code>	
83DD	<code>PUSH WORD [BX+4] ; Save po on stack</code>	01E4 _____
83E0	<code>CMP [BX+4], DI ; If po < eo</code>	
83E3	<code>JAE .Next</code>	01E6 _____
83E5	<code>MOV SI, [BX+4]</code>	
83E8	<code>ADD SI, 2 ; then call</code>	01E8 _____
83EC	<code>CALL QSort ; QSort(po+2, eo)</code>	
	<code>.Next:</code>	01EA _____
83EF	<code>POP DI ; Put po into DI</code>	
83F0	<code>POP SI ; Put bo into SI</code>	01EC _____
83F1	<code>CMP SI, DI ; If bo < po</code>	
83F3	<code>JAE .QExit</code>	01EE _____
83F5	<code>SUB DI, 2 ; then call</code>	
83F9	<code>CALL QSort ; QSort(bo, po-2)</code>	01F0 _____
	<code>.QExit:</code>	
83FC	<code>RET</code>	01F2 _____

(a) Assume that just before the execution of the three-byte `CALL QSort` instruction at offset 7210 in the Code Segment, $(SP) = 01F2$, $(SI) = 0300$, and $(DI) = 0304$, and at this time the values of the three words in memory starting at offset 0300 are 4444, 5555, and 6666. (All values are in hexadecimal.) Determine the hexadecimal contents of words at addresses `SS:01E0` through `SS:01F2` just before the first execution of the `RET` instruction at offset 83FC. If there is insufficient information to determine a particular word, then write “Unknown.”

(b) Why does `QSort` save the partition offset `po` on the stack instead of in a fixed memory location labeled `PartOff`? Write your answer in clear English.

(c) Suppose there are 100 (decimal) unused words on the stack just before the execution of `CALL QSort` on an input array of n words, this array of words is already sorted in increasing order, and the `Partition` subroutine pushes 8 words onto the stack. What is the maximum value of n for which this call to `QSort` will **not** overflow the stack?

Problem 4. (38 pts) Assume that the computer is connected to only three devices, D1, D2, and D3, which send interrupt signals periodically. There are no other external interrupt sources. All interrupt service routines are nonpreemptive.

(a) Some information about the devices and their service routines is given below. Complete the table; write “Unknown” when there is insufficient information to determine a value.

Device	D1	D2	D3
Interrupt type	0Ah (high priority)	0Bh	0Ch (low priority)
Service load	_____ %	18%	2%
Interrupt rate	250 Hz	_____ Hz	_____ Hz
Service time (milliseconds)	2 ms	3 ms	_____ ms
Worst case interrupt latency	_____ ms	2.4 ms	_____ ms

(b) Suppose that the computer has just executed a 3-byte MOV instruction and is responding to an interrupt signal from D1. Immediately before the execution of the first instruction of the D1 interrupt service routine, the following values are known (in hexadecimal):

(CS) = 0229 (SP) = 0340 M(SS:0342) = 1FBC
 (IP) = 1406 M(SS:0340) = 0290 M(SS:0344) = 0A41

For parts (i) to (v), write your answers in hexadecimal. If there is insufficient information to determine answer, write “Unknown.”

(i) Value of word at linear address 00028: _____

(ii) Value of word at linear address 0002A: _____

(iii) Linear address of last executed MOV instruction: _____

(iv) Value in DS register: _____

(v) Value in FLAGS register: _____

(Hint: the interrupt flag IF is bit 9 in the FLAGS register; see Lab Notes page 21)

(c) You want to attach a fourth device D4 whose interrupt rate is 100 Hz, with a service time of 3.1 ms. Explain why you should NOT attach D4 to the computer at any priority, even if you could make the interrupt service routines preemptive. (Hint: what is the service load of D4?)

Problem 5. (28 pts) Here is the beginning of an implementation of the `Init` subroutine for Machine Problem #4. This code fragment should initialize the interrupt vector for the type 1Ch interrupt to point to `TimerInt`. (`Init` does **not** need to save and restore registers it uses.)

```

1   TVEC EQU 70h
2   Init:
3       MOV AX, CS           ; Install timer interrupt
4       MOV [ES:TVEC], AX   ; service routine
5       MOV AX, TimerInt    ; in this code segment
6       MOV [ES:TVEC+2], AX

```

Although there are no syntax errors, this code fragment has four logical mistakes. Identify each mistake and explain clearly, in detail, why it is a mistake by describing what might happen when the program fragment is executed. Explain how to fix each mistake, using the line numbers.

1. _____

2. _____

3. _____

4. _____
