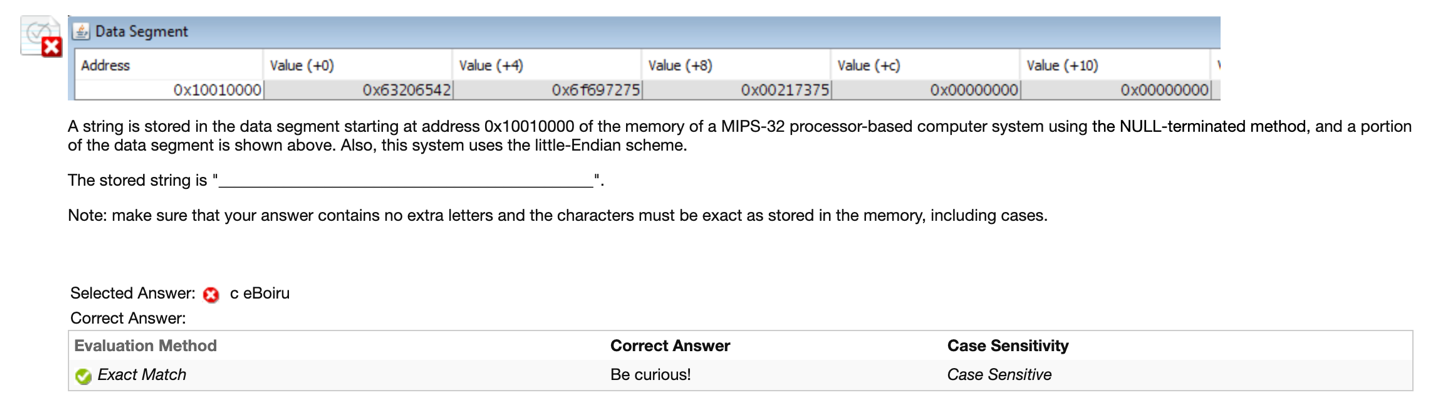
Exam I Rework

Q3:



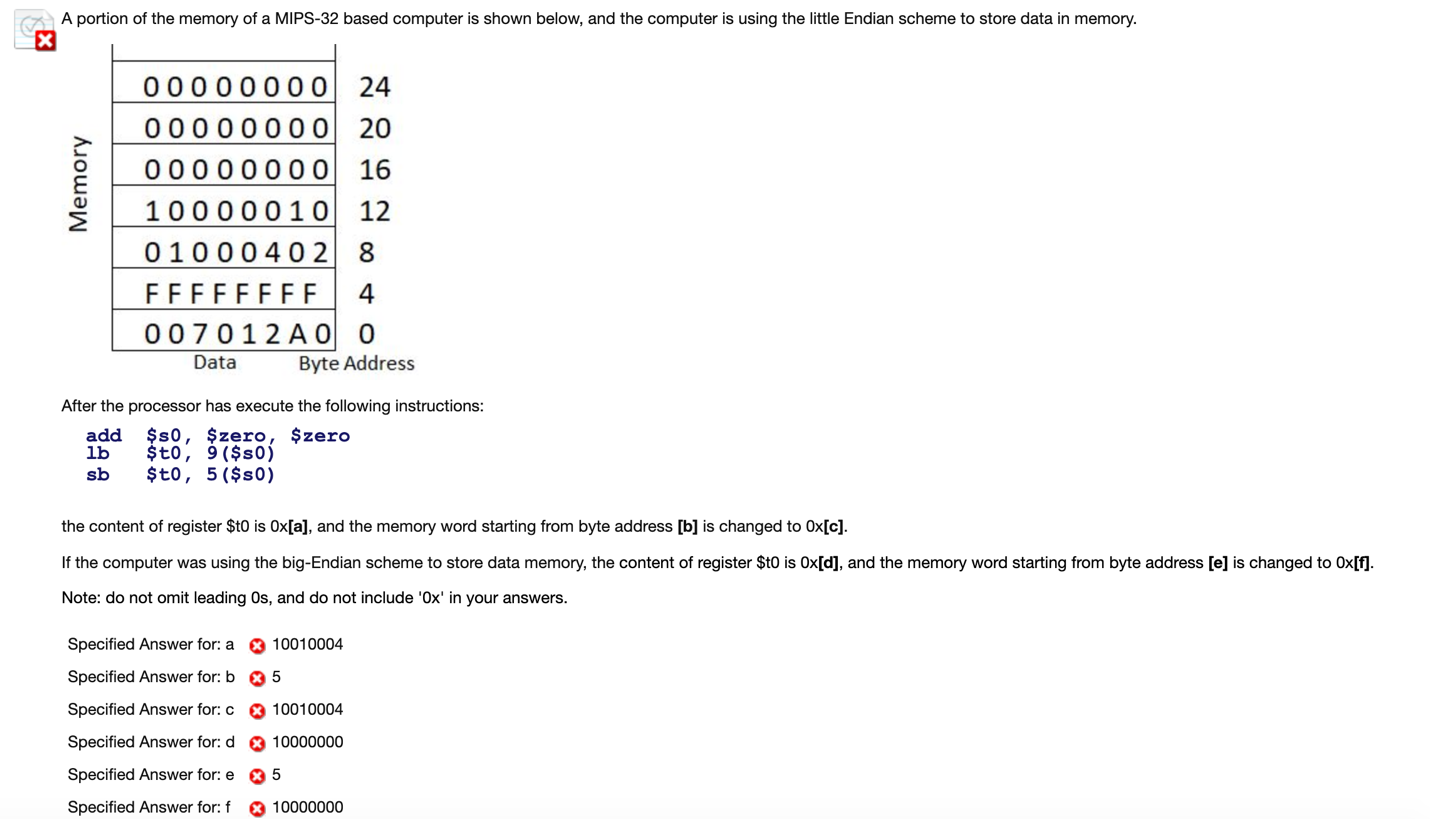
Cause of Error:

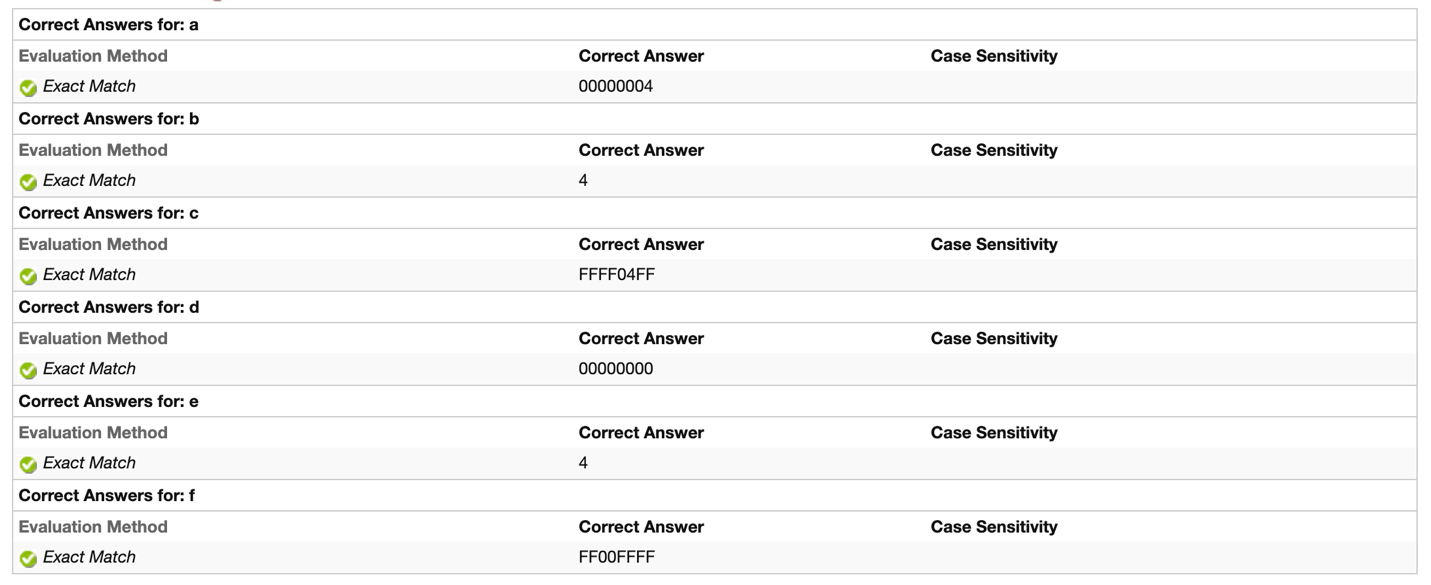
Little-Endian should read from left to right. I read “c eBoiru\0” from right to left.

Rework Solution:

The binary code for the string should be “0x426520637572696f75732100”. I stop at “00” because this is “\0” which is the end of the string. Look up in the reference sheet and I get ”Be curious!”.

Q4:





Cause of Error:

I mistook “lb” and “sb” to “lw” and “sw”. And I did calculate wrong even in load/save word situation… In addition, memory word start from {n | n mod 4 == 0};

Rework Solution:

Add $s0, $zero, $zero # set $s0 t0 0x00000000

Lb $t0, 9($s0) # load $t0 with the byte at address: $s0 + 4 \* 9

Sb $t0, 5($s0) # save $t0 to the byte at address: $s0 + 4 \* 5

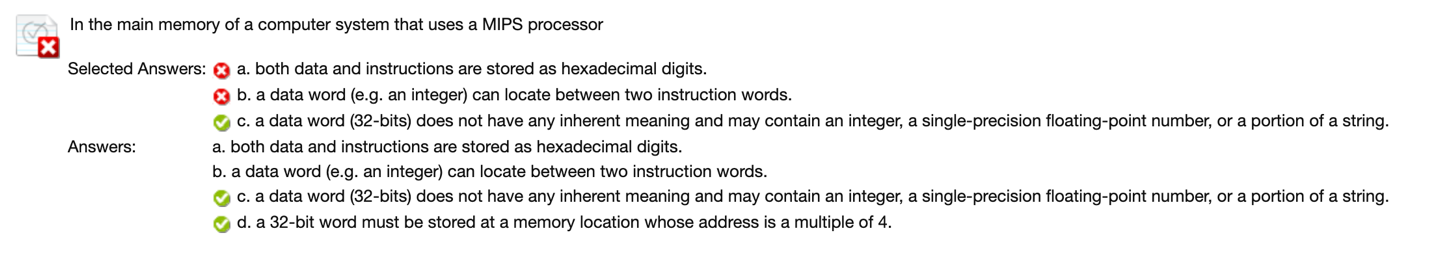
Little-Endian:

Load 0x04 from 0x00000036 and save to $t0 with “0x00000004”, then save to 0x00000020. Memory start from 0x00000004 is changed to “0xFFFF04FF”.

Big-Endian:

Load 0x00 from 0x00000036 and save to $t0 with “0x00000000”, then save to 0x00000020. Memory start from 0x00000004 is changed to “0xFF00FFFF”.

Q5:



Cause of Error:

Carelessness and bad-mastered basic concept.

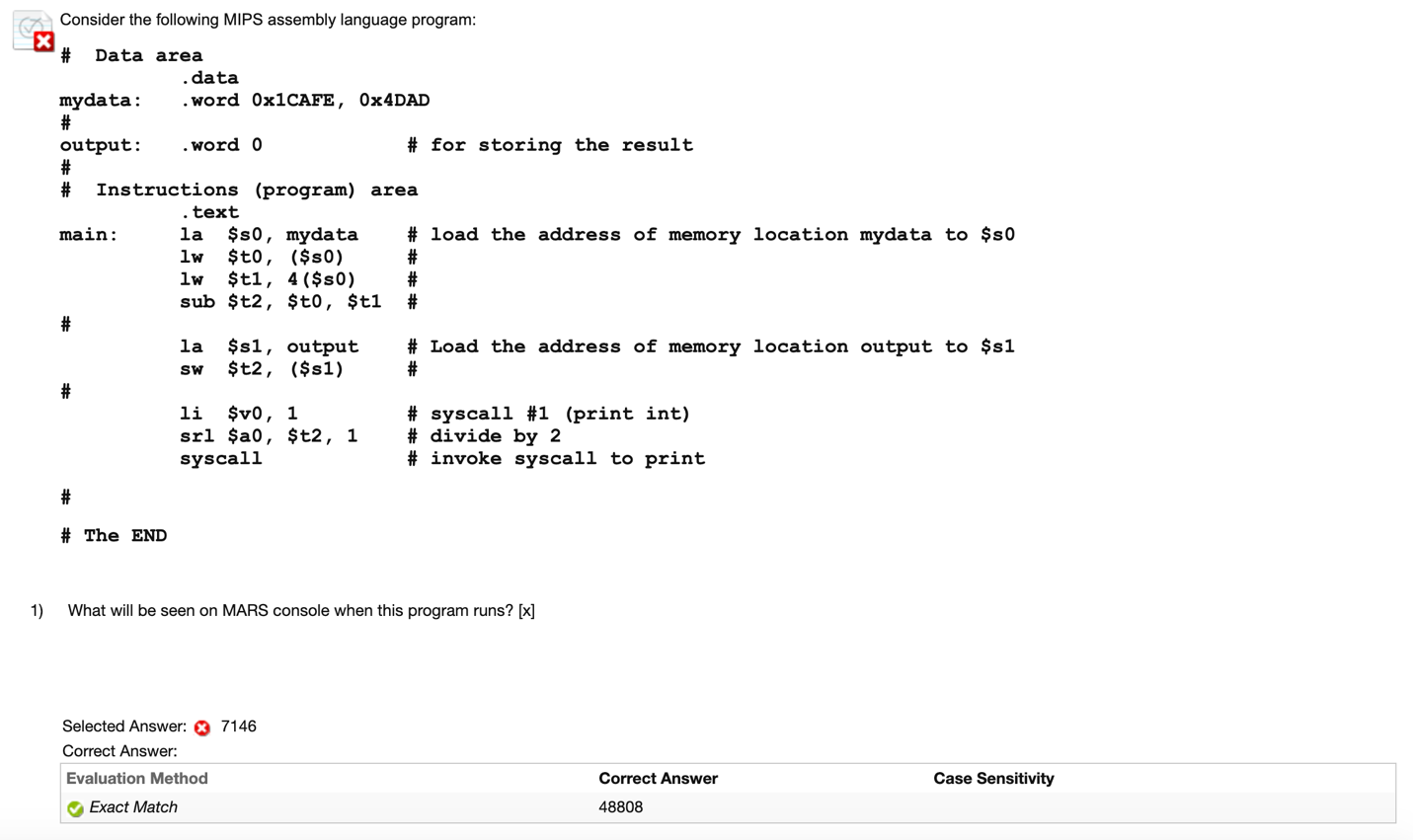
Rework Solution:

a. Binary

b. Can not between two instruction word

d. Address starts from 0 and interval is 4

Q8:



Cause of Error:

I expect mydata would be stored like “1CAFE4DAD”.

Rework Solution:

$t0: 0x0001CAFE = 0000 0000 0000 0001 1100 1010 1111 1110

$t1: 0x00004DAD = 0000 0000 0000 0000 0100 1101 1010 1101

-$t1: = 1111 1111 1111 1111 1011 0010 0101 0011

$t2 = $t0 - $t1 = 0000 0000 0000 0001 0111 1101 0101 0001

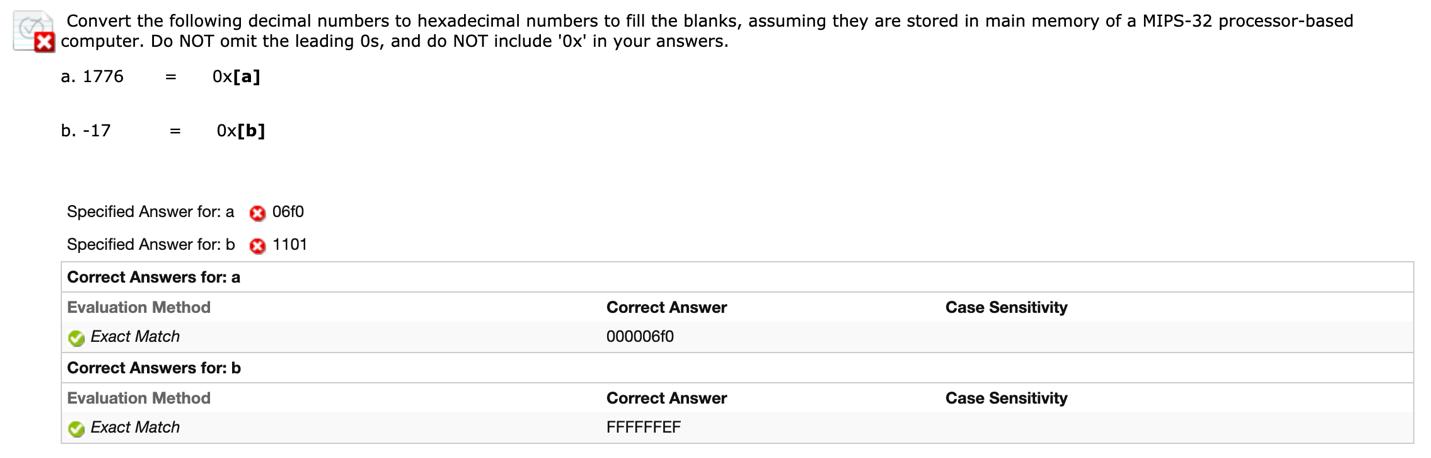
$a0 = $t2 << 1 = 0000 0000 0000 0000 1011 1110 1010 1000

= 0xBEA8

= 11 \* 16^3 + 14 \* 16^2 + 10 \* 16 + 8

= 48804

Q9:



Cause of Error:

Did not recognize is 32-bit and a stupid mistake in get negative number.

Rework Solution:

1. 1776 = 111 \* 16 + 0

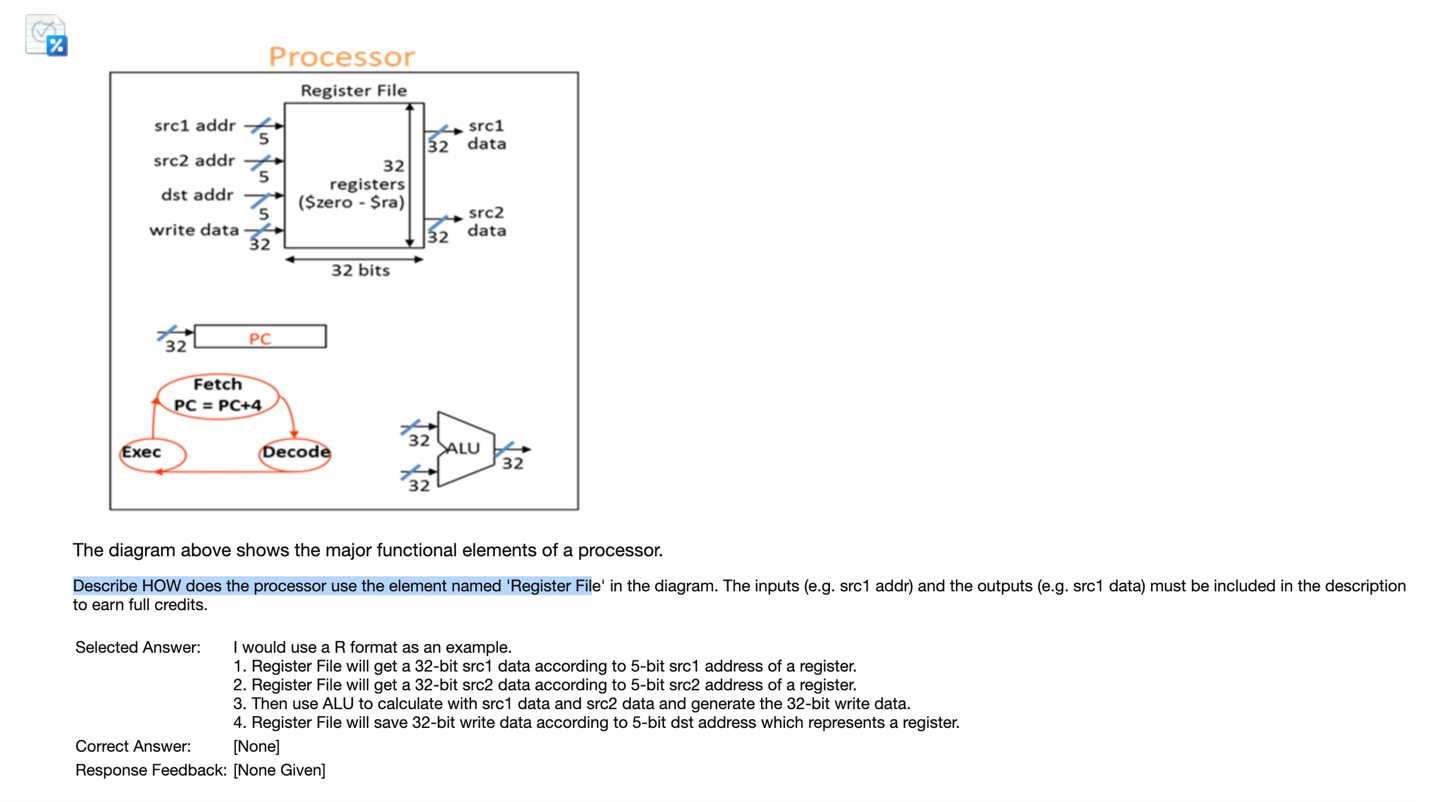
111 = 6 \* 16 + 15

1776 = 6 \* 16^2 + 15 \* 16 + 0 = 0x6F0 = 0x000006F0

1. 17 = 1 \* 16 + 1 = 0x11 = (00010001)2

-17 = (11101111)2 = 0xEF = 0xFFFFFFEF

Q10:



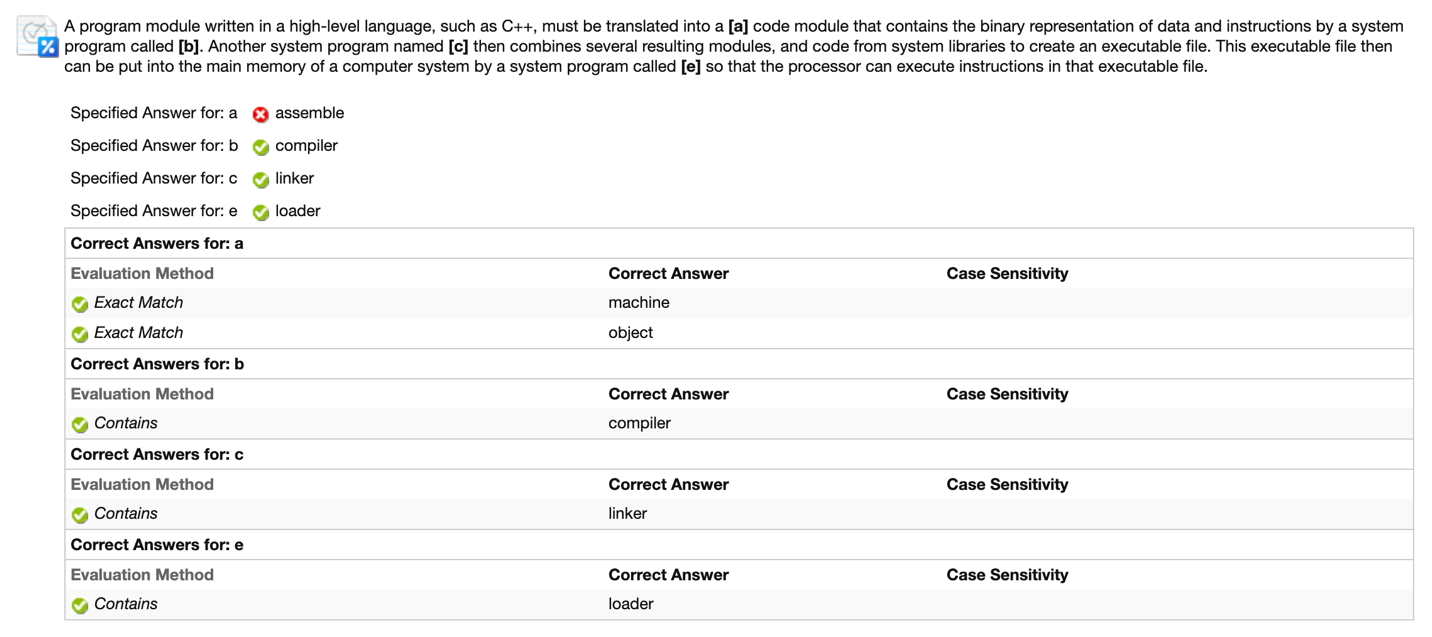
Cause of Error:

My note is terrible. I can only remember a general idea and here it is. I make some improvement from the text book.

Rework Solution:

1. The processor’s 32 general-purpose registers are stored in Register File
2. For each data word to be read from the registers, we need a 5-bit input (e.g. src1 address) to the register file that specifies the register number to be read and a 32-bit output (e.g. src1 data) from the register file that will carry the value that has been read from the registers.
3. To write a data word, we need two inputs: one to specify the register number to be written and one to supply the data to be written into the register.
4. When we get the result, we need a 5-bit input (dst address) to the register file that specifies the register number to be read and a 32-bit output (write data) from the register file that will save the result.

Q11:



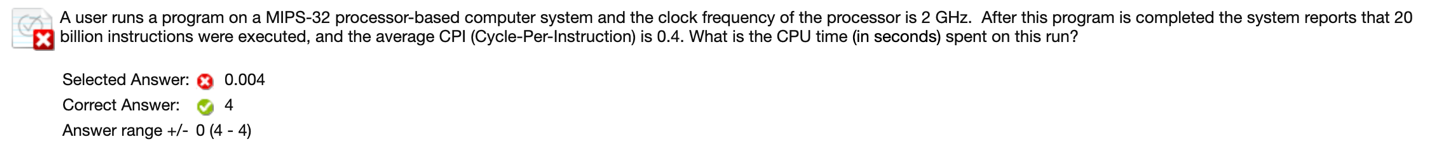
Cause of Error:

A think set that I should fill some words related with system programs.

Rework Solution:

Compiler transfers a high-level language into a machine code module.

Q16:



Cause of Error:

CPU time = Instruction Count \* CPI / Clock Rate

= 20 \* 10^9 \* 0.4 / 2 \* 10^12(wrong in GHz)

= 0.004 s

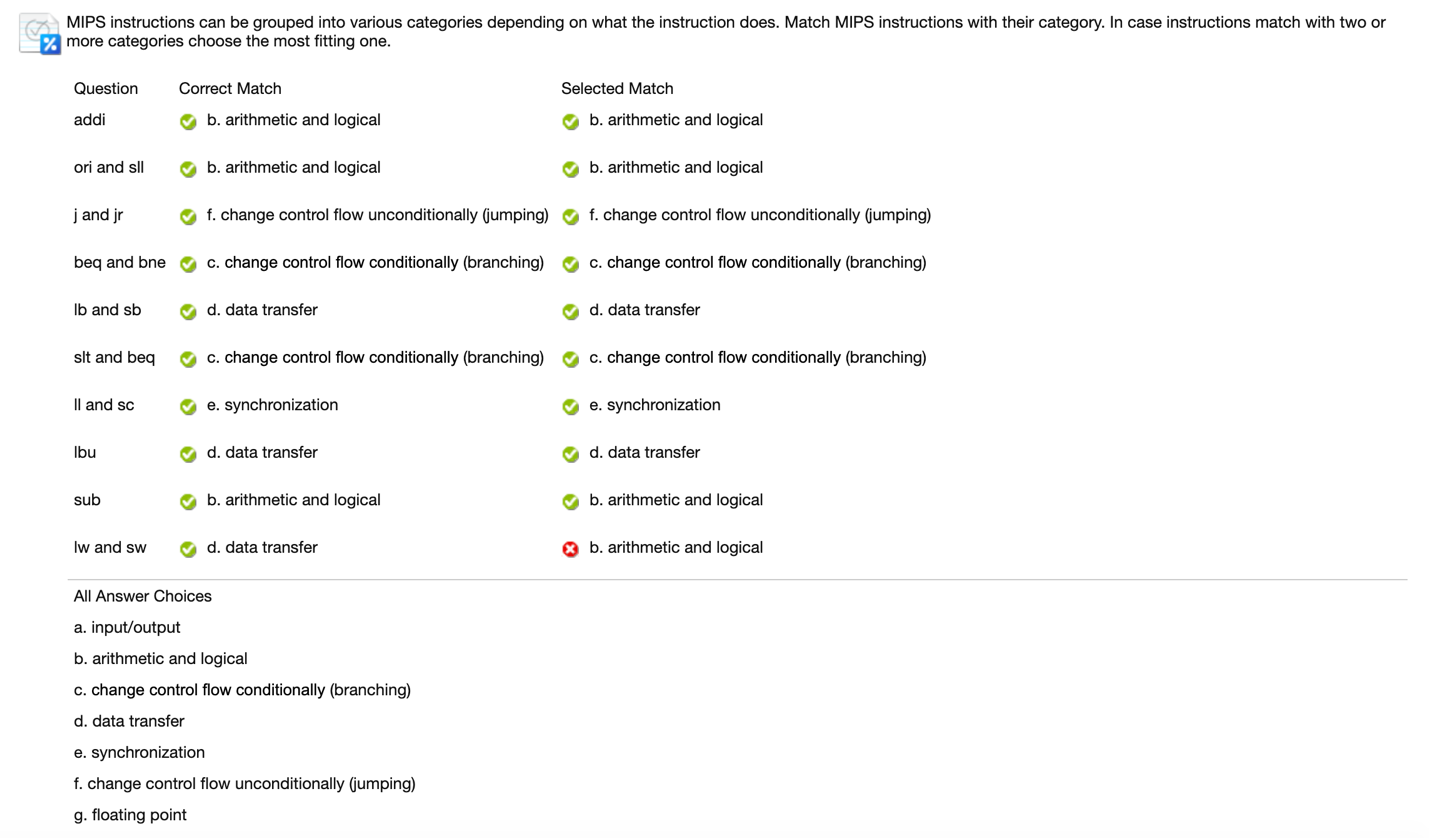
Rework Solution:

CPU time = Instruction Count \* CPI / Clock Rate

= 20 \* 10^9 \* 0.4 / 2 \* 10^9

= 4 s

Q17:



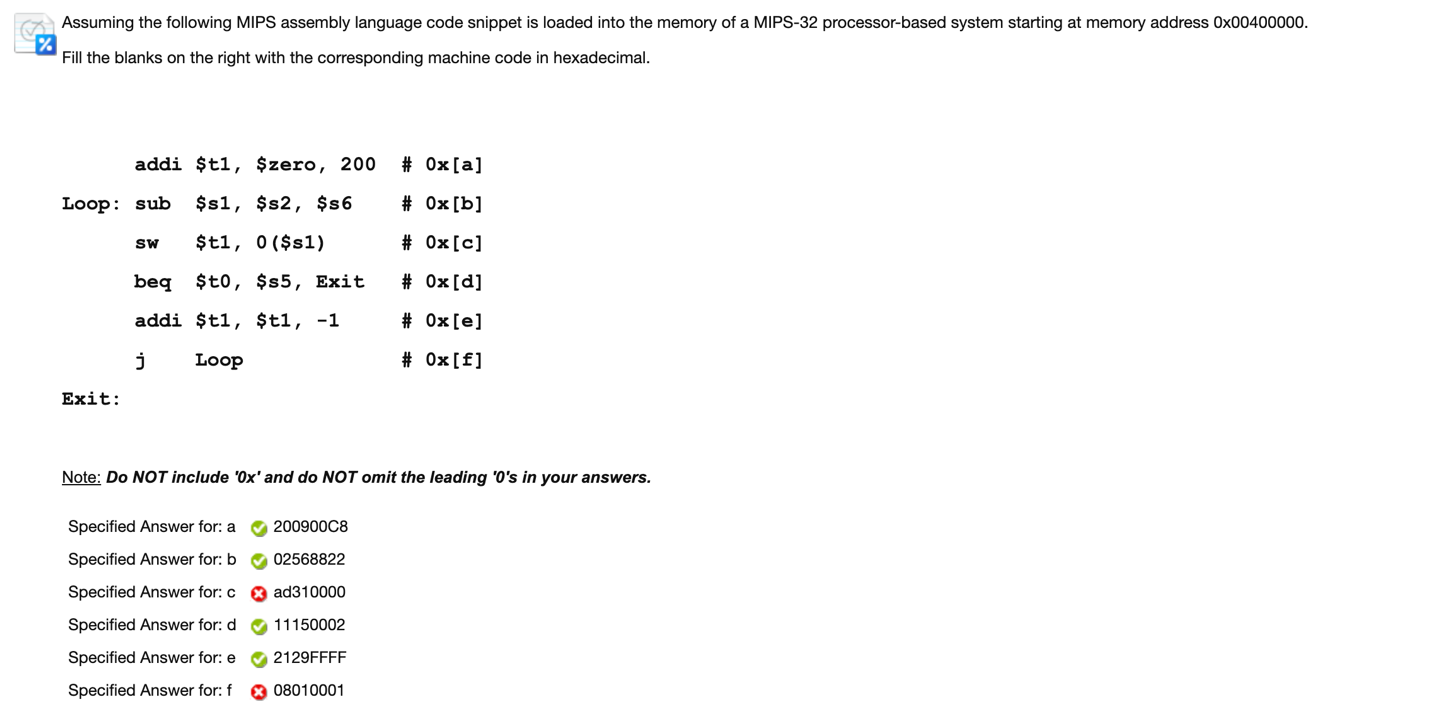
Cause of Error:

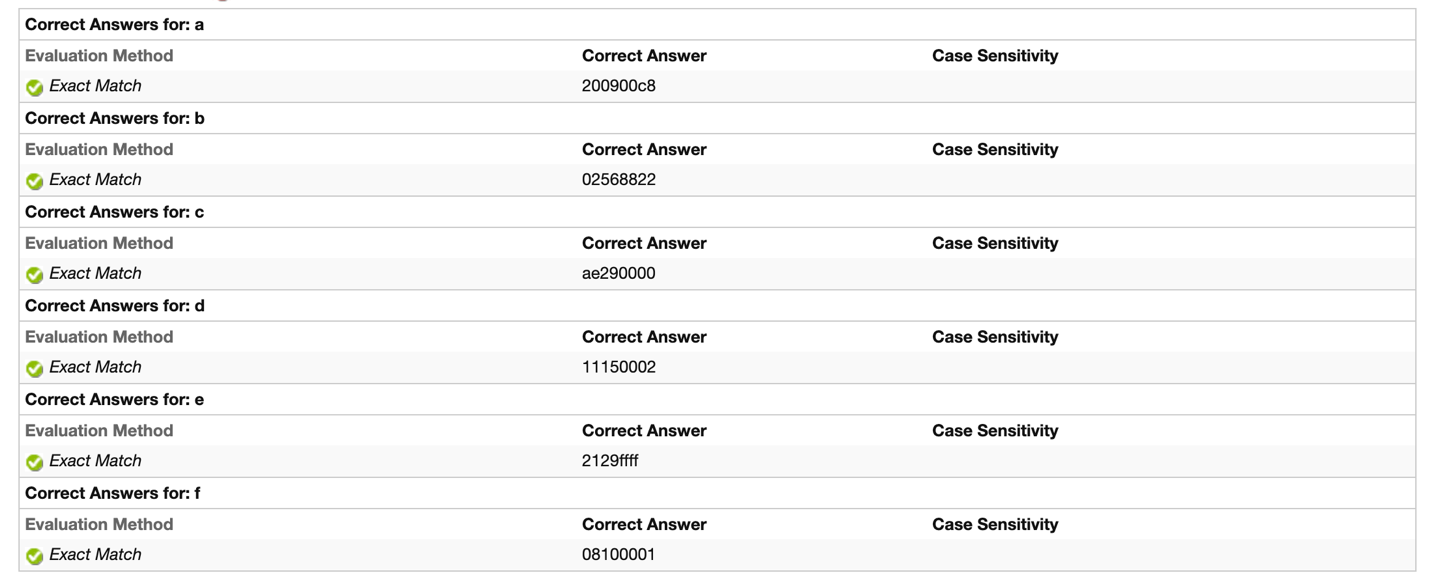
Careless. I did not have enough time, so many low-level mistake happened below.

Rework Solution:

Load/save belongs to data transfer.

Q18:





Cause of Error:

Careless.

c. Sw $t1, 0($s1)

I format

Op rs rt immediate

0x2b 9($t1) 17($s1) 0

101011 01001 10001 00000 00000 000000

0xAD310000

Error in get rt and rs

f. Error in copy…

Rework Solution:

c. Sw $t1, 0($s1)

I format

Op rs rt immediate

0x2b 17($s1)9($t1) 0

101011 10001 01001 00000 00000 000000

0xAE290000

f. j Loop

J format

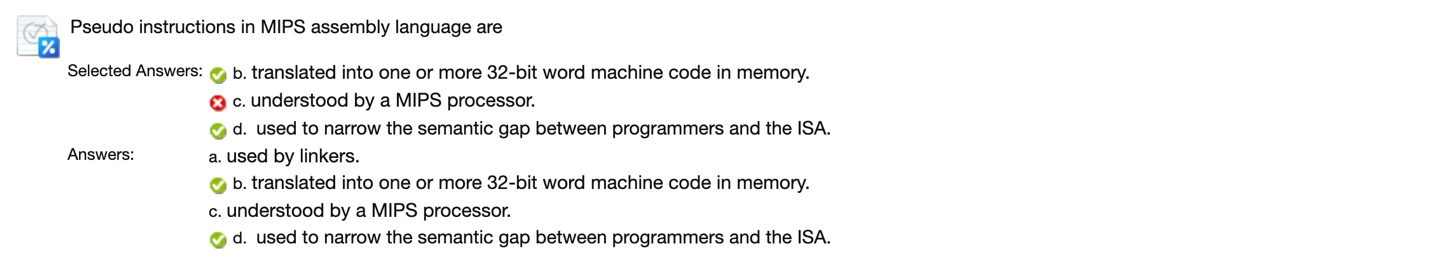
Op address

0x02 (0x00400004)32-bit -> ()26

000010 0000 0100 0000 0000 0000 0000 01

0x08100001

Q19:



Cause of Error:

Careless.

Rework Solution:

A pseudo-instruction expands to several machine instructions which can be understood by a MIPS processor.