Richardson Chayden
Mark: 6/8 (total score: 6/8)

+108/1/26+

CS 218 - Worksheet 7

☐0 ☐1 ☐2 ■3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ■5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ■9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ■7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ■7 ☐8 ☐9

← Encode last 5 of NSHE ID

Enter Name: First Last
Chayden Richardson

For each of the following questions, select only the best answer.

Question 1 Assuming IEEE, what is the 32-bit floating point format, what is the representation of -6.5 in hex? Recall that:

$$(-1)^S : 1.F : 2^{E-127}$$

where S(1-bit) : E(8-bits) : F(23-bits).

- ☐ 0xC1800000
- ☐ 0x40D00000
- ☐ 0xC0C00000
- ☒ 0xC0D00000

1/1

Question 2 When arguments are passed using **values**, it is referred to as:

- ☐ None of these.
- ☐ Pass-by-name.
- ☐ Pass-by-value-reference.
- ☒ Pass-by-value.
- ☐ Pass-by-reference.

1/1

Question 3 When arguments are passed using **addresses**, it is referred to as:

- ☐ Pass-by-value.
- ☐ None of these.
- ☐ Pass-by-value-reference.
- ☐ Pass-by-name.
- ☒ Pass-by-reference.

1/1

Question 4 According to the standard calling convention, as discussed in class, what is the purpose of the initial push's and final pop's within most functions?

- ☒ Make space on the stack for the local variables.
- ☐ Store the arguments, if any, on the stack.
- ☐ No specific purpose, but part of the standard calling convention.
- ☐ Preserve register(s) for the calling routine.

0/1

Question 5 What is the purpose of the **add rsp, <immediate>** after the call statement?

- ☐ None of these responses are correct.
- ☐ Clear any stack-based arguments off the stack.
- ☐ Clear the preseved registers from the stack.
- ☒ Clear the arguments.

0/1

Question 6 If three (3) arguments are passed on the stack, what is the value for the **<immediate>**?

- ☐ 16
- ☐ 32
- ☐ 40
- ☐ 8
- ☒ 24

1/1

Question 7 If a function is called 73 times, how many copies of the function code are placed into memory?

- [ ] None of these responses are correct.
- [x] 1
- [ ] 73
- [ ] 12

1/1

Question 8 Given the following code fragment, what is in the **eax** and **edx** registers?

```
    list   dd   2, 7, 4, 5, 6, 3

    mov    rbx, list
    mov    rsi, 1
    mov    rcx, 2
    mov    eax, 0
    mov    edx, dword [rbx+4]
lp: add    eax, dword [rbx+rsi*4]
    add    rsi, 2
    loop   lp
    imul   dword [rbx]
```

- [x] eax = 0x00000018 and edx = 0x00000000
- [ ] eax = 0x00000000 and edx = 0x00000024
- [ ] eax = 0x00000004 and edx = 0x00000000
- [ ] eax = 0x00000024 and edx = 0x00000007

1/1