CS220: Computer Organization
Quiz#3

Name:
Roll No.:

General instructions: In all the questions, you will assume 32-bit big-endian MIPS ISA.

1. Consider the following sequence of MIPS instructions. Write down what would `EXIT` and `START` be replaced with in the encoding of the `blez` and `bne` instructions respectively in the MIPS ISA. **(1.5+1.5 points)**

```
       xor  $1, $1, $1
       blez $2, EXIT
START: lw   $13, 0($3)
       addi $13, $13, 1
       sll  $13, $13, 2
       sw   $13, 0($3)
       addi $3, $3, 4
       addi $1, $1, 1
       slt  $4, $1, $2
       bne  $4, $0, START
EXIT:
```

Solution: Assuming that the address of the first instruction is A, the addresses of the `blez` and `bne` instructions are respectively $A + 4$ and $A + 36$. The addresses of the `START` and `EXIT` labels are respectively $A + 8$ and $A + 40$. So, the distance from the `blez` instruction to its target (i.e., `EXIT`) is 36 bytes and the distance from the `bne` instruction to its target (i.e., `START`) is -28 bytes. So, `EXIT` will be encoded as 9 (these many instructions away) in the `blez` instruction and `START` will be encoded as -7 in the `bne` instruction.

2. Consider the following MIPS instruction sequence. What is the final hexadecimal value in $t0? **(2 points)**

```
addi $t0, $0,  0x42
sll  $t0, $t0, 0x10
addi $t0, $t0, 0xabcd
addi $t1, $0,  0x3
srav $t0, $t0, $t1
```

Solution: The hexadecimal value in $t0 is shown below alongside each instruction after the instruction completes execution.

```
addi $t0, $0,  0x42      // $t0 has 0x0000 0042
sll  $t0, $t0, 0x10      // $t0 has 0x0042 0000  (shift left by 16 bits)
addi $t0, $t0, 0xabcd    // $t0 = 0x00420000 + 0xffffabcd (sign-extended immediate)
addi $t1, $0,  0x3       //     = 0x0041abcd
srav $t0, $t0, $t1       // $t0 has 0x00083579 (shift right arithmetic by 3 bits)
```

3. Consider the following MIPS instruction sequence. Assume that initially $t0 contains 0x10000000 and $t1 contains 0x10000004. Initially, the word stored at address 0x10000000 is 0x12fedcba and the word stored at address 0x10000004 is 0xabcdef12. What is the final hexadecimal value of the word stored at address 0x10000004? **(3 points)**

```
lb $t0, 2($t0)
sh $t0, 2($t1)
```

Solution: The byte layout is shown below where the first column shows the address and the second column shows the byte stored at that address. Note big-endian layout.

```
0x10000000    0x12
0x10000001    0xfe
0x10000002    0xdc
0x10000003    0xba
0x10000004    0xab
0x10000005    0xcd
0x10000006    0xef
0x10000007    0x12
```

The `lb` instruction accesses the byte at address 0x10000002 and sign-extends it. So, $t0 has 0xffff ffdc after the `lb` instruction. The `sh` instruction will copy the least significant half word (i.e., 0xffdc) from this and store it starting at address 0x10000006. So, the new byte stored at address 0x10000006 would be 0xff and the new byte stored at address 0x10000007 would be 0xdc. Therefore, the word stored at address 0x10000004 would be 0xabcdffdc.

4. Consider a function `f` written using the C language. The function `f` calls another function `g` having five arguments all of type integer. If the function `f` can allocate all its local variables in registers without spilling and does not have any caller-saved registers, how much stack space in bytes should be allocated for `f`? **(2 points)**

Solution: One argument of `g` needs to be spilled to stack. Additionally, $ra needs to be saved on stack before calling `g`. So, a total of eight bytes would be allocated by `f` on stack.