Spring 2022 *Name*: Ivan Zelenkov, ID: 2600950
CSCI 3301 Homework # 1
Due: **Thursday February 24, 2022 (11:59 pm), via Moodle.**

The rules:

- All work must be your own. You are not to work in teams on this assignment. You are not to use materials from previous offerings of this course.
- Format: Submit as a single file (via moodle) containing a PDF file. Email me (ayn@cs.uno.edu) assignment only if moodle is not working.
- You may use the textbook and lecture notes, but do NOT search the Internet for solutions.
- The submission deadline is strict. Therefore, please submit on time.

Total Marks = 100

(Q1) [9 points]

Assume \$t0 holds the value 0x00101000. What is the value of \$t2 after the following instructions? Explain.

```
slt $t2, $zero, $t0           # $zero < 160, therefore $t2 = 1
bne $t2, $zero, ELSE          # $zero != 1, therefore jump to label ELSE
j DONE
ELSE: addi $t2, $t2, 2        # $t2 = 1 + 2 = 3
DONE:
```

(Q2) [3 × 5=15 points] Assume the following register contents:

\$t0 = 0xAAAAAAAA, \$t1 = 0x12345678

(a) For the register values shown above, what is the value of \$t2 for the following sequence of instructions?

\$t0 = 0xAAAAAAAA = 1010 1010 1010 1010 1010 1010 1010 1010
\$t1 = 0x12345678 = 0001 0010 0011 0100 0101 0110 0111 1000

```
sll $t2, $t0, 4   # shift left $t0 on 4 bits, 1010 1010 1010 1010 1010 1010 1010 0000

or $t2, $t2, $t1          # OR, 1010 1010 1010 1010 1010 1010 1010 0000
                          #       0001 0010 0011 0100 0101 0110 0111 1000
                          #----------------------------------------------------------
                          #      1011 1010 1011 1110 1111 1110 1111 1000 = -1161888008 =
                                                                      0xBABEFEF8
```

(b) For the register values shown above, what is the value of $t2 for the following sequence of instructions?

```
sll $t2, $t0, 4   # shift left $t0 on 4 bits, 1010 1010 1010 1010 1010 1010 1010 0000

andi $t2, $t2, -1         # AND, 1010 1010 1010 1010 1010 1010 1010 0000
                          #         1111 1111 1111 1111 1111 1111 1111 1111
                          #------------------------------------------------------------
                          #        1010 1010 1010 1010 1010 1010 1010 0000 = -1431655776 =
                                                                        0xAAAAAAA0
```

(c) For the register values shown above, what is the value of $t2 for the following sequence of instructions?

```
srl $t2, $t0, 3              # 0001 0101 0101 0101 0101 0101 0101 0101

andi $t2, $t2, 0xFFFFFFFF    # AND, 0001 0101 0101 0101 0101 0101 0101 0101
                             #         1111 1111 1111 1111 1111 1111 1111 1111
                             #----------------------------------------------------------
                             #     0001 0101 0101 0101 0101 0101 0101 0101 = 357913941 =
                                                                       hex 15555555
```

(Q3) [3 × 6 =18 points] Consider the following MIPS loop:

```
LOOP: slt $t2, $zero, $t1
      beq $t2, $zero, DONE
      addi $t1, $t1, -1
      addi $s2, $s2, 2
      j LOOP
DONE:
```

(a) Assume that the register $t1 is initialized to the value 10. What is the value in register $s2 assuming the $s2 is initially zero?

The loop repeats ten times, each time adding the integer 2 to \$s2 and decrementing \$t1 until the value is 0. Thus, the value of \$s2 will be **20**.

(b) For each of the loops above, write the equivalent *Java* / C code routine. Assume that the registers \$s1, \$s2, \$t1, and \$t2 are integers A, B, *i*, and temp, respectively.

Java

```
int i = 10;
int B = 0, temp = 0;

while (0 < i) {
        i--;
        B += 2;
}
```

(c) For the loops written in MIPS assembly above, assume that the register \$t1 is initialized to the value N. How many MIPS instructions are executed?

Each LOOP executes 5 instructions. But if the condition of \$t1 becomes 0, then only 2 instructions will be executed.

If the register \$t1 is initialized to the value N > 0, then the routine above will guarantee to execute 5N + 2 instructions, or 2 instructions if N = 0.

(Q4) [3 × 6 =18 points]

Assume that for a given program 70% of the executed instructions are arithmetic, 10% are load/store, and 20% are branch.

(a) Given the instruction mix and the assumption that an arithmetic instruction requires 2 cycles, a load/store instruction takes 6 cycles, and a branch instruction takes 3 cycles, find the average CPI.

$$CPI_{average\ for\ arithmetic\ instructions} = 0.7 * 2 = 1.4$$

$$CPI_{average\ for\ load\ and\ store} = 0.1 * 6 = 0.6$$

$$CPI_{average\ for\ branch} = 0.2 * 3 = 0.6$$

$$CPI_{average\ total} = 1.4 + 0.6 + 0.6 = \mathbf{2.6}$$

(b) For a 25% improvement in performance, how many cycles, on average, may an arithmetic instruction take if load/store and branch instructions are not improved at all?

2.6 * (1 – 0.25) = $\frac{0.7 * 2}{improvement\ factor}$ + 0.1 * 6 + 0.2 * 3;

1.95 = $\frac{1.4}{improvement\ factor} + 1.2$;

Improvement factor = $\frac{0.75}{1.4}$;

Improvement factor = 0.536

(c) For a 50% improvement in performance, how many cycles, on average, may an arithmetic instruction take if load/store and branch instructions are not improved at all?

For 100 instructions:

260 * (1 – 0.5) = $\frac{0.7 * 2}{improvement\ factor}$ + 10 * 6 + 20 * 3;

130 = $\frac{1.4}{improvement\ factor} + 120$;

Improvement factor = $\frac{10}{1.4}$;

Improvement factor = 7.14

(Q5)

[2 x 5 = 10 points]

(a) Provide the ***type***, ***assembly language instruction***, and ***binary representation*** of instruction described by the following MIPS fields:

op=0, rs=3, rt=2, rd=3, shamt=0, funct=34

Refer to the figure 2.5, I can say that the instruction is subtraction because funct = 34 and it is a **R-format** because op=0, shamt=0, and address is n.a. Therefore, the assembly of this is:

sub $v1, $v1, $v0

Now, convert the above decimal representation into binary:

op	rs	rt	rd	shamt	funct
000000	00011	00010	00011	00000	100010

(b) Provide the ***type***, ***assembly language instruction***, and ***binary representation of instruction*** described by the following MIPS fields:

op=0x23, rs=1, rt=2, const=0x4

Refer to the figure 2.5, I can say that the instruction is load word because op = 35 in decimal, rd = n.a., shmat=n.a., funct=n.a, and we have address 4. Therefore, it is an **I-format**, and the assembly of this is:

lw $v0, 4($at)

Now, convert the above decimal representation into binary:

op	rs	rt	rd	const
100011	00001	00010	00011	0000000000000100

(Q6)

[15 points]

Translate the following C code to MIPS. Assume that i is in $s1, j is in $s2, the base address of A is in $s6 and B is in $s7.

```
if( i < 10 ) {
        B[ i ] = A[3] + j
        i = i + 1
}
```

```
addi $t0, $t0, 10      # $t0 = 10

bge $s1, $t0, Exit     # If i is greater than 10, then jump to the Exit

lw $t1, 12($s6)        # Temporary reg $t1 gets A[3]
add $t1, $t1, $s2      # Temporary reg $t1 gets A[3] + j

sll $t2, $s1, 2        # Temporary reg $t2 = i * 4
add $t2, $t2, $s7      # $t2 = address of B[i]

sw $t1, $t2            # Stores A[3] + j back into B[i]

addi $s1, $s1, 1       # i = i + 1

Exit:
```

(Q7)

[15 points]

Translate the following code to MIPS. Assume that i is in $a0, k is in $a1, the base address of A is in register $s6.

```
public int quizFunc (int i, int k){
        while( A[ i ] >= k ) {
                i = i - 1
        }
    return i;
}
```

```
quizFunc:
    Loop: sll $t1, $a0, 2          # Temporary reg $t1 = i * 4
          add $t1, $t1, $s6        # $t1 = address of A[i]
          lw  $t0, 0($t1)          # Temporary reg $t0 = A[i]

          blt $t0, $a1, Exit       # If A[i] < k, then jump to Exit

          addi $a0, $a0, -1        # i = i - 1

          j Loop                   # go to Loop

    Exit:
          add $v0, $a0, $zero      # initialize a return value register $v0 by value of i
          jr $ra                   # return to the caller
```