This problem set has 10 questions, for a total of 100 points. Answer the questions below and mark your answers in the spaces provided. **For all questions, please provide details on how your answer was calculated.**

Your Name: yeury Cuba Liriano

1. For the following C statements, write the corresponding MIPS assembly code. Assume that variables `i, j` are assigned to registers `$s0, $s1`, respectively. Assume that the base address of the arrays `A, B` are in registers `$s2, $s3`, respectively.

   (a) [5 points] `B[5] = A[i+j];`

✓

```
add $t0, $s0, $s1      # index -> i+j
sll $t0, $t0, 2        # mult by 4, 2^N = 2^2 = 4
add $t0, $t0, $s2      # A[i+j] memory address
lw  $t1, 0($t0)        # value A[i+j] from memory to register
sw  $t1, 20($s3)       # store val of B[5] From register to RAM
```

   (b) [5 points] `B[9] = A[i-j];`

✓

```
sub $t0, $s0, $s1      # index -> i-j
sll $t0, $t0, 2        # mult by 4
add $t0, $t0, $s2      # A[i-j] mem addy
lw  $t1, 0($t0)        # value A[i-j]
sw  $t1, 36($s3)       # store val of B[9]
```

2. [10 points] For the MIPS assembly instructions below, what is the corresponding C statement? Assume that variables `f`, `g` are assigned to registers `$s0`, `$s1`, respectively. Assume that the base address of arrays `A`, `B` are in registers `$s2`, `$s3`, respectively.

```
sll  $t0, $s0, 2
add  $t0, $s2, $t0
sll  $t1, $s1, 2
add  $t1, $s3, $t1
lw   $s0, 0($t0)
addi $t2, $t0, 4
lw   $t0, 0($t2)
add  $t0, $t0, $s0
sw   $t0, 0($t1)
```

F, g -> $s0, $s1 (Fx4)
A+(F·4) --> A[F] ?
g·4
B[g] ?
val of A[F]
A[F+4]
val of
A[F+4] + A[F] ?

B[g] = A[F+1] + A[F]

3. [5 points] Show how the value `0x7ba11ad6` would be arranged in memory for a *little-endian* and for a *big-endian* machine. Assume that the data are stored starting at address 0 and that the word size is 4 bytes.

0x 7b | a1 | 1a | d6

Big Endian:

val	7b	a1	1a	d6
Byte address	0x00	0x01	0x02	0x03

Little Endian:

val	d6	1a	a1	7b
Byte Address	0x00	0x01	0x02	0x02

4. Assume that registers `$s0`, `$s1` hold the values `0x90000000` and `0xC0000000`, respectively.

   (a) [5 points] What is the value of `$t0` for the following assembly code? Is the result in `$t0` the desired result, or has there been overflow?

```
sub $t0, $s0, $s1
```

0x90|00|00|00 = 1001 0000... = 2415919104
0xC0|00|00|00 = 1100 0000... = 3221225472
→ decimal from 2's complement = -1879048192
= -1073741824
if we sub 241... - 322... = -8,053,063,68
if we only focus on the first 4 bits, since the rest are 0's, we have 1001 − 1100 = −011 or 9 − 12 = −3

   (b) [5 points] What is the value of `$t0` for the following assembly code? Is the result in `$t0` the desired result, or has there been overflow?

```
add $t0, $s0, $s1
add $t0, $t0, $s0
```

As shown above, if we add 2415919104 + 3221225472 = 5,637,144,576
now this would be an overflow because an integer only holds 4B and if we account for signed two's complement too, then it will also be an overflow

5. [10 points] Provide the type, assembly language instruction, and hexadecimal representation of the instruction described by the following MIPS fields: `op=0, rs=2, rt=4, rd=1, shamt=0, funct=32`.

✓ Type = ——> MIPS R-format Instructions

Assembly language Instruction = add $at, $v0, $a0
($1 $2 $4)

000000 00010 00100 | 00001 | 00000 100000

hexadecimal: 0x00440820

every 4 bits is 1 hex value

6. [10 points] Provide the type, assembly language instruction, and hexadecimal representation of the instruction described by the following MIPS fields: `op=0x21, rs=0, rt=1, const=0x6`.

✓ Type = MIPS I-format

Assembly language = lh $at, 6($zero)

hexadecimal: 0x84010006

op = 0x21 = 100001
rs = 0 = 00000
rt = 00001
const = 00000000 00000110

7. Assume the following register contents for `$t0`, `$t1` respectively: `0xAAAABEEF` and `0x12341234`

(a) [5 points] What is the value of `$t2` for the following sequence of instructions?

```
sll  $t2, $t0, 16
or   $t2, $t2, $t1
```

1 Byte = 2 hex values
16 shifts = 2 Bytes = 4 hex values

0AAAABEEF << 4 = 0X BEEFC0000
turn both to bits then perform or bitwise operation on bits.

$t2 = 0xbeff1234 → decimal

(b) [5 points] What is the value of `$t2` for the following sequence of instructions?

```
sll  $t2, $t0, 8
andi $t2, $t2, -1
```

After shift

0x AAAA BEEF → AABEEF00
after adding -1 to

$t2 = 0x AABEEF00

(c) [5 points] What is the value of `$t2` for the following sequence of instructions?

```
srl  $t2, $t0, 16
andi $t2, $t2, 0x0000FFEF
```

After shifting right 0000AAAA
And 0000FFEF
0000AAAA

Ignore all 0's because it will always give 0 with AND operation: Consion
AAAA = 1010 1010 1010 1010
FFEF = 1111 1111 1110 1111 AND
1010 1010 1010 1010

0x0000AAAA

8. [10 points] For the following `C` statement, write a minimal sequence of MIPS assembly instructions that does the identical operation. Assume variable `A` is in `$t0` and `$s0` is the base address of array `C`.

```
A = C[8] << 8;
```

```
lw  $t0, 32($s0)   # load val at C[8]
sll $t0, $t0, 8    # shift left
```

9. [10 points] Translate the following loop into C. Assume that the integer a is held in register $t1, $s0 holds the integer ans, and $s2 holds the base address of the integer array data.

```
        addi $t1, $0, 0          # a = 0
loop:   lw   $s1, 0($s2)         # s1 = *data
        add  $s0, $s0, $s1       # s0 = ans = ans + *data
        addi $s2, $s2, 4         # data++.
        addi $t1, $t1, 2         # a = a+2  or  a+= 2
        slti $t2, $t1, 100       # if a < 100 then t2 = 1 else t2 = 0
        bne  $t2, $s2, loop      # t2 while( t2 != *data)
```

```
int a = 0;
while (a < 100){
    ans += data[a];
    a+=2;
}
```

10. For the following code, assume that register $t1 contains the address 0x20000100 and the data at that address is 0xa1b2c3d4.

```
lbu $t0, 0($t1)
sw  $t0, 0($t1)
```

Big Endian: 0xa1b2c3d4
Little Endian: 0xd4c3b2a1

(a) [5 points] What value is stored in 0x20000104 on a *little-endian* machine?

0x000000d4

(b) [5 points] What value is stored in 0x20000104 on a *big-endian* machine?

0xa1000000