## CS61C Discussion 3 – Memory and MIPS

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>add</td>
<td>add dest, src0, src1</td>
<td>add $s0, $s1, $s2</td>
</tr>
<tr>
<td>sub</td>
<td>sub dest, src0, src1</td>
<td>sub $s0, $s1, $s2</td>
</tr>
<tr>
<td>addi</td>
<td>addi dest, src0, immediate</td>
<td>addi $s0, $s1, 12</td>
</tr>
<tr>
<td>lw</td>
<td>lw dest, offset(base addr)</td>
<td>lw $t0, 4($s0)</td>
</tr>
<tr>
<td>sw</td>
<td>sw src, offset(base addr)</td>
<td>sw $t0, 4($s0)</td>
</tr>
<tr>
<td>beq</td>
<td>beq src0, src1, branchAddr</td>
<td>beq $t0, $t1, label</td>
</tr>
</tbody>
</table>

**Hint:** `blt, ble, bgr, bgt, and jr` may also be useful...

### C

```c
// $s0 -> a, $s1 -> b
// $s2 -> c, $s3 -> z
int a=4, b=5, c=6, z;
z = a+b+c+10;

// $s0 -> int *p = intArr
// $s1 -> a
p[0] = 0;
int a = 2;
p[1] = a;
p[a] = a;

// $s0 -> a, $s1 -> b
int a = 5, b = 10;
if (a + a == b) {
    a = 0;
} else {
    b = a - 1;
}

/*What does this do? (Not C, in English) */
addi $s0, $0, 0
addi $s1, $0, 1
addi $t0, $0, 30
loop: beq $s0, $t0, done
    add $s1, $s1, $s1
addi $s0, $s0, 1
    j loop
done:  # done!
```

### MIPS

```mips
int sum(int n) {
    return n ? n + sum(n - 1) : 0;
}
```
Implement `str eq`, which returns true (remember what values evaluate to logical true and false!) if its two char pointer arguments point to two equal, null-terminated strings (and false otherwise), in both C and MIPS.

```c
int str eq (const char* a, const char* b)

//your code here
```

What are the instructions to branch on each of the following conditions?

- `$s0 < $s1$
- `$s0 <= $s1$
- `$s0 > 1$
- `$s0 >= 1$

What are the 3 meanings `unsigned` can have in MIPS?

What is the distinction between zero extension and sign extension? When do we use each?