MIPS 3

Shift left logical (sll) move all the bits to the left by the specified number of bits (fill empty space with 0s)

`s11 $t2, $t0, 2`

Shift right logical (srl) move all the bits to the right (again fill empty with 0s)

`srl $t2, $t0, 2`

Shift right arithmetic (sra) move all the bits to the right (fill empty bits with the sign bit, 1 for negative 0 for positive)

`sra $t2, $t0, 2`

```
000000110000 >> 2 # shift right 2
-> 000000001100
```

Mips Labels

- LA- Load address
  - Syntax: `LA $reg, label`
  - Function- Stored the address the label is pointing at into the register
- LI- Load Immediate
  - Syntax- `LI $reg, immediate`
  - Function- Stores the given immediate value into the register
- Both of these instructions are actually a combination of 2 instructions (LUI and ORI)

```
if(i == j){
        k = k + i;
}
bne $t2, $t3, L1 #if (t2 != t3) goto l1
addu $t4, $t4, $t2
```

```
slt $t3, $t1, $t2
# set t3 to 1 if less t1 < t2.
# else clea t3 to be 0

slti $t3, $t1, 100
```

```
# set t3 to be 1 if t1 < 100
# else clear t3 to be 0
```

- Translate into mips the following

```
if(a > b){
        c = a;
}
```

MIPS

```
slt $t5, $t3, $t2 # b < a
beq $t5, $zero, L1 # if t5 == 0
or $t4, $t2, $zero #c = a
```

`slt` sets the target to 1 if the value of source 1 is less than source 2
In the above example we needed to switch the values for the comparison statement. If we wanted to add an else statement we need an unconditional jump