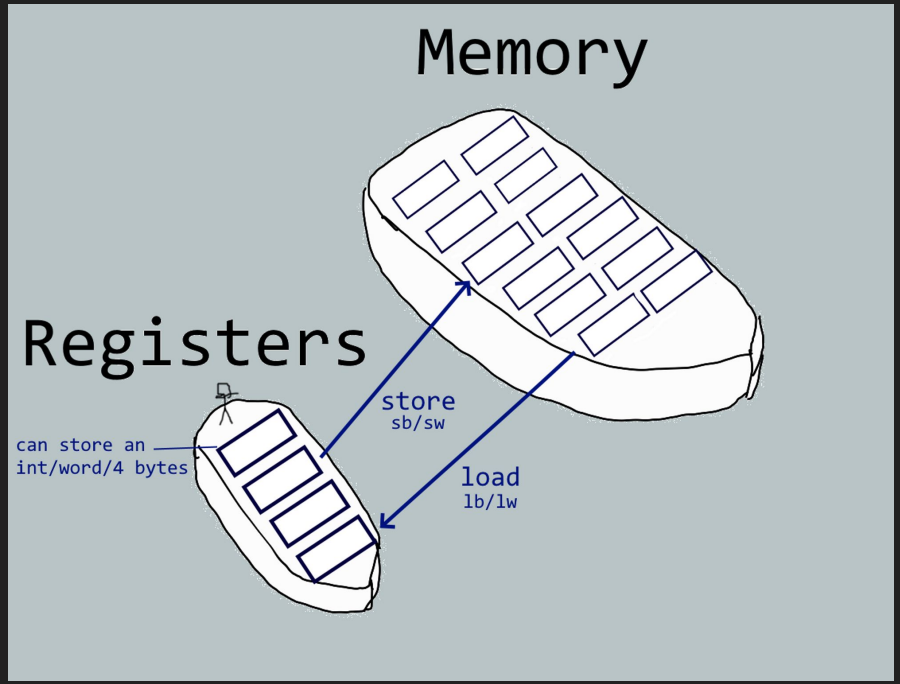
Week 3

Aren’t registers enough?

Aren’t registers enough?

- Sometimes we want to store A LOT of numbers (definitely more than 32)
- This is when we will store data in memory (RAM)
- We can either:
  - **load** data from memory into a register
  - **store** data from a register into memory



MIPS Memory Directives

- Tutorial Q2

- There are others, but less important:
  - .half
  - .float
  - .ascii
- Tip: (saves you writing 7, 7, 7, 7, 7)

```
b:   .byte 7:5      # int8_t b[5] = {7,7,7,7,7};
```

MIPS Memory Directives

3. Give MIPS directives to represent the following variables:

a. `int u;`
b. `int v = 42;`
c. `char w;`
d. `char x = 'a';`
e. `double y;`
f. `int z[20];`

MIPS Memory Directives

3. Give MIPS directives to represent the following variables:

a. `int u;`
b. `int v = 42;`
c. `char w;`
d. `char x = 'a';`
e. `double y;`
f. `int z[20];`

Answers:

a. `u: .space 4`
b. `v: .word 42`
c. `w: .space 1`
d. `x: .byte 'a'`
e. `y: .space 8`
f. `z: .space 80` (20 * 4-byte ints)

MIPS Memory Instructions

We have two types:

- **Load** instructions
  - lw / lh / lb
- **Store** instructions
  - sw / sh / sb

Usage:

- **Address = base + offset*sizeof(element)**

```
lw   $t0, bb
```

```
la   $t1, cc
lw   $t0, ($t1)
```

```
la   $t1, cc
lw   $t0, 8($t1)
```