Arithmetic Operations
&
Control Structures

Arithmetic Operations

- add R1, R2, R3 → R1 = R2 + R3
  - if R2 / R3 is $zero, used for copying content from the other register to R1
- addi R1, R2, immediate value → R1 = R2 + immediate value
  - if R2 is $zero, used for storing a constant value in R1
- sub R1, R2, R3 → R1 = R2 - R3
- subi R1, R2, immediate value → R1 = R2 - immediate value
- mul R1, R2, R3 → R1 = R2 * R3
- div R1, R2, R3 → R1 = quotient of R2 / R3
- div R1, R2 → **lo** = quotient of R1 / R2
  hi = remainder of R1 / R2

Store in memory

```
.data
        ans:            .word       0
.text
        addi $t0, $zero, 429412
        addi $t1, $zero, 10

        div $t4, $t0, $t1
        sw $t4, ans
```

Control Structures

Basic Components

- Label
- Branch instructions
  - Unconditional: **j** label
  - Conditional
    - beq R1, R2, label
    - bne R1, R2, label

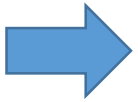
    - slt/sgt/sle/sge R1, R2, R3 ➔ R1 = 1 if R2 </>/≤/≥ R3, R1 = 0 otherwise
      bne R1, $zero, label

    - blt/bgt/ble/bge R1, R2, label

If - Else if - Else

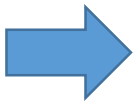
- if (condition 1)
  - …… // if
- else if (condition 2)
  - …… // else if
- else
  - …… // else
- ……. // later part



- Branch on **condition 1** to label1
- Branch on **condition 2** to label2
- …… # else
- j after
- label1:
  - ……. # if
  - j after
- label2:
  - ……. # else if
  - j after
- after:
  - …….. # later part

for to while

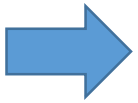
- for(i = 0; i < 10; i++)
  - …… // body
- ….. // later part



- i = 0
- while (i < 10)
  - …… // body
  - i++
- ….. // later part

Loop

- while (condition)
  - …… // body
- ….. // later part



- label:
  - Branch on negation of **condition** to exitLabel
    - …… # body
    - j label
- exitLabel:
  - ….. # later part

Assignments

- Handling overflow in integer arithmetic operations
- Floating point arithmetic operations
- Logical operations