Control Flow Operations

4 min

In Assembly, we use branches and jumps to provide control flow in our programs.

A *branch statement* is used to define a conditional statement. Think of it as a fork in a river. We are sitting in our kayak and trying to make a decision. If something is true, take the fork on the left; if it's not true, take the fork on the right.

Since we are dealing with binary values, most branching operations center around arithmetic operations and their results.

There are two primary ways we branch:

- On equality
    - BEQ (Branch on Equal)
    - BNE (Branch on Not Equal)
- On reference to zero
    - BGTZ / BGEZ (Branch on > zero / Branch on ≥ zero)
    - BLTZ / BLEZ (Branch on < zero / Branch on ≤ zero)

The other way we can control the flow of our program is to jump directly to a specific instruction using a *jump statement*, followed by the memory location that contains the instruction. This is similar to making function calls or returning back out of a function.

- Jump to Register 31 and execute the instruction stored there:

```
J $31
```

to Clipboard

In Assembly, we jump to a memory location and execute the code stored there as its own binary instruction. This can start a separate subroutine or simply perform the function of starting a loop over.

Instructions

1. Checkpoint 1 Passed

1.

Create a new variable, answer_1, and set it equal to the opcode used to go to a specific instruction if two registers do not equal each other.

Make sure to capitalize your answer.

Hint

This type of conditional is handled by a branch operation. There are many branch operations that cover a variety of scenarios that you may come across in your code, reference the exercise for the one that fits this scenario.

2. Checkpoint 2 Passed

2.

Create another variable, answer_2, and set it equal to the opcode used to execute an instruction stored in a specific memory location.

Hint

These types of instructions are very useful in the execution of loops and function calls.

control_flow_operations.py

```
# Write your answers here:

answer_1 = 'BNE'

answer_2 = 'J'
```