Lab: Branch and Conditional Logic in RISC-V

Objective:

- Understand and apply branch instructions (`BEQ`, `BNE`, `BLT`, `BGE`) in RISC-V to implement conditional logic.
- Practice using basic arithmetic instructions (`ADD`, `ADDI`, `SUB`) to modify values based on conditions.

Assignment:

Complete the following exercises by writing RISC-V assembly programs.

Exercise 1: Conditional Arithmetic Operations

1. Initialize two registers (`t0`, `t1`) with predefined values.
2. If `t0` is equal to `t1`, add 4 to `t0`.
3. If `t0` is not equal to `t1`, subtract 4 from `t1`.
4. Store the result in `t2` (if modifying `t0`) or `t3` (if modifying `t1`).

Your code should go here:

```
// Example of expected format:
// Initialize t0 and t1 with values
// Compare t0 and t1
// Apply conditional logic and store result in t2 or t3
```

Exercise 2: Conditional Value Adjustment

1. Assign arbitrary values to two registers, `s0` for `A` and `s1` for `B`.
2. If `A` (stored in `s0`) is less than `B` (stored in `s1`), add `A` to itself (doubling it).
3. If `A` is greater than or equal to `B`, subtract 3 from `B`.
4. Reflect the changes by updating `s0` and `s1` accordingly.

Your code should go here:

```
// Initialize s0 and s1 with values for A and B
// Use branch instructions to check if A < B
// Double A or subtract 3 from B based on condition
```

Exercise 3: Nested Conditional Logic with Value Modification

1. Begin with three initialized registers: `a0`, `a1`, and `a2`, each containing different integer values.
2. Compare `a0` to `a1`. If `a0` is greater than `a1`, check if `a2` is greater than a fixed value (e.g., 10).
    - If `a2` is greater than 10, add 5 to `a0`.
    - Otherwise, subtract 5 from `a0`.
3. If `a0` is less than or equal to `a1`, subtract 2 from `a1`.
4. The final values of `a0` and `a1` should be stored or displayed.

Your code should go here:

```
// Initialize a0, a1, a2 with values
// Compare a0 and a1
// Nested condition: if a0 > a1, check if a2 > 10
// Apply addition or subtraction to a0, or modify a1 based on condition
```

Requirements:

- **Comment your code** to explain the logic behind each instruction.
- **Set up initial conditions** correctly, such as register values, before starting the comparisons.
- **Test each program thoroughly** to ensure it executes as intended.