# Week 10 Lab RISC-V Assembly language programming

## Objectives

Develop understanding and experience of:

1. Using branch instructions in RISC-V assembly language programming
2. Working on a programming challenge and comparing solutions in Java to assembly language

## Part B RISC-V Assembly Language Big Challenge

Would The following code branch or not branch?

addi t0, zero, 3

addi t2, zero, 4

bge t0, t2, branch

Would this code branch? (assuming the values in the registers remain the same as above)

bge t2, t0, branch

1. Take in three integers from the user, output “In Order”, if the second number is greater than the first and the third is greater than the second. If that is not the case, output “Not in order”.

## Extension Exercises

There are four different programming puzzles here. **Choose one** of them to work on. You might want to consider your solution(s) in Java before working on the assembly language version.

1. FizzBuzz. This is a commonly used exercise in programming and problem solving and is of the level that might be used as an initial question in an interview or assessment for people applying for work as a software developer.

Display the numbers from 1 to 100 (inclusive) but for those digits which are multiples for three display the word “Fizz" instead of the number, and for the multiples of five display “Buzz". For numbers which are multiples of both three and five display “FizzBuzz". Note that a number is a multiple of another if the remainder is zero when you divide one by the other.

For example, the output may look something like this:  
1  
2  
Fizz  
4  
Buzz  
Fizz  
7  
8  
Fizz  
Buzz  
11  
Fizz  
13  
14  
FizzBuzz  
16  
…

1. Each position on a chess board is identified by a combination of a letter and a number as shown in the image above. Use loops to systematically list all the positions. It is up to you whether to start a1, b1, c1 or go a1, a2, a3, but you might want to separate each output with a newline character.

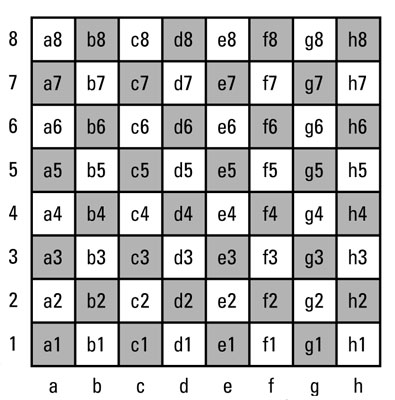


Image from <https://www.dummies.com/games/chess/understanding-chess-notation/>

1. Create a program that asks the user for a number and then outputs a table of powers of two up to that number.

A sample run of the program showing the output and user input would be as follows:

Please enter a whole number 8

Powers of two

=============

Power Value

0 1

1 2

2 4

3 8

4 16

5 32

6 64

7 128

8 256

There are many ways that you could program this, so it will need some planning. You do not need to cater for overflow or invalid input being entered.