Connor Kalina

Java Abacus Computer

SCOPE:

The following documentation outlines an implementation of an “abacus” computer in java. An abacus computer has only two functions, increment, and decrement that it can perform on it’s registers’ values. Through this project, we prove that using just these two functions, we can build much more complicated and useful methods and data structures. Implemented below are the following functions: Assignment, test for equality, addition, multiplication, less than or equal, and division. Using these functions I was able to build an associative memory data structure, and the methods required to use it.

FUNCTIONS:

- User can input code with described syntax into an input file
- Instructions are created from text file and sent to abacus computer
- Abacus computer computes instructions and displays a trace of its computation.
- Final associative memory configuration is displayed

PERFORMANCE:

Due to the nature of an associative data structure using exponential functions on large integers, and the abacus computer’s inherent ability to manipulate register values by only one at a time, the performance of this structure is lacking. It is recommended when using the associative memory data structure, that only small register values and arguments are used. For example, register 2 value 5, not register 5 value 9.

However, if the associative memory data structure is not used, exponentiation and manipulation of extremely large numbers is less likely to be needed, and larger numbers can be used with acceptable performance. For example, the program can generate the first 200 prime values in just a few seconds. The compute() function in the AbacusComputer class has example math functions that can easily be modified to demonstrate the effectiveness of its functions.

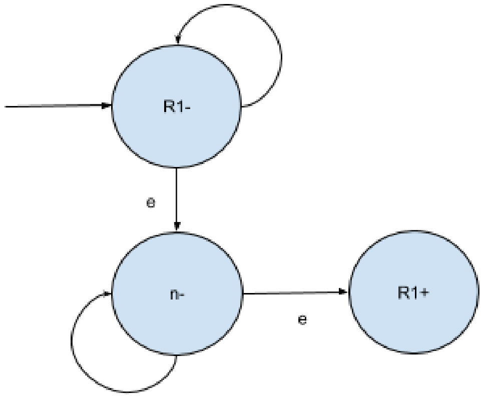
USAGE DIRECTIONS:

- To use the program simply compile the code with command “javac Wrapper.java“
- Make sure the input file is in the same directory as the class files
- Then, run it with the command “java Wrapper input.txt”
- The process will run the instructions as specified in the input file
- To not use the associative memory data structure, edit the compute() method in the AbacusComputer class with math tests as desired.
- Example input files are provided in the ExampleInput folder, use the file’s name as the argument. Ex: java Wrapper ifRemainder.txt

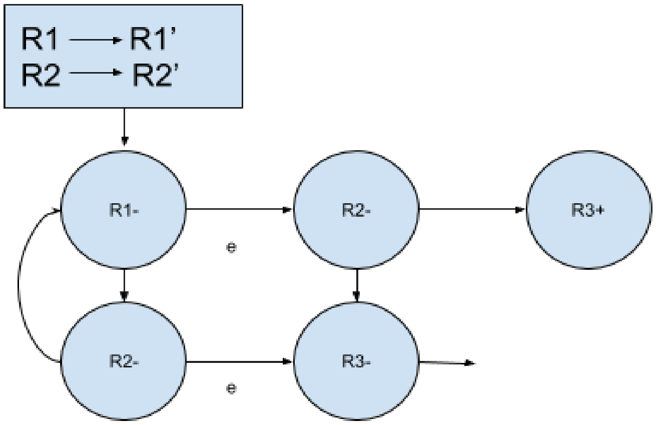
Connor Kalina

Assignment function:
Syntax : R1=n
Puts n into R1, after R1 is reset

e: take path when register value is empty (0)

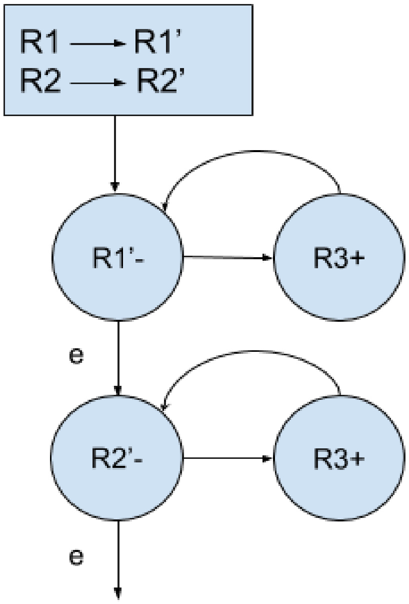


Test Equality
Syntax : R1==R2,R3
Tests if two register values are equal, sets third register value to 1 if they are, and 0 if they aren't

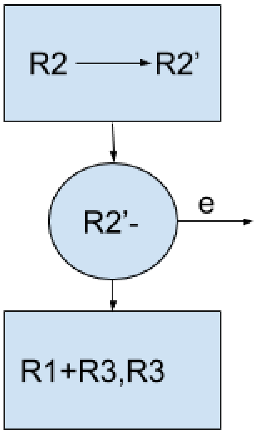


Connor Kalina

Addition
Syntax : R1+R2,R3
Adds two register values together, stores the result in a third register.

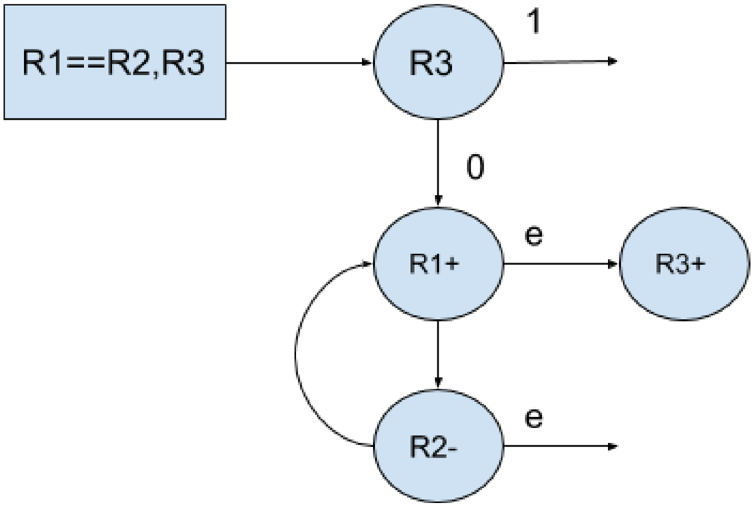


Multiplication
Syntax : R1*R2,R3
Adds R1 to R3, R2 times



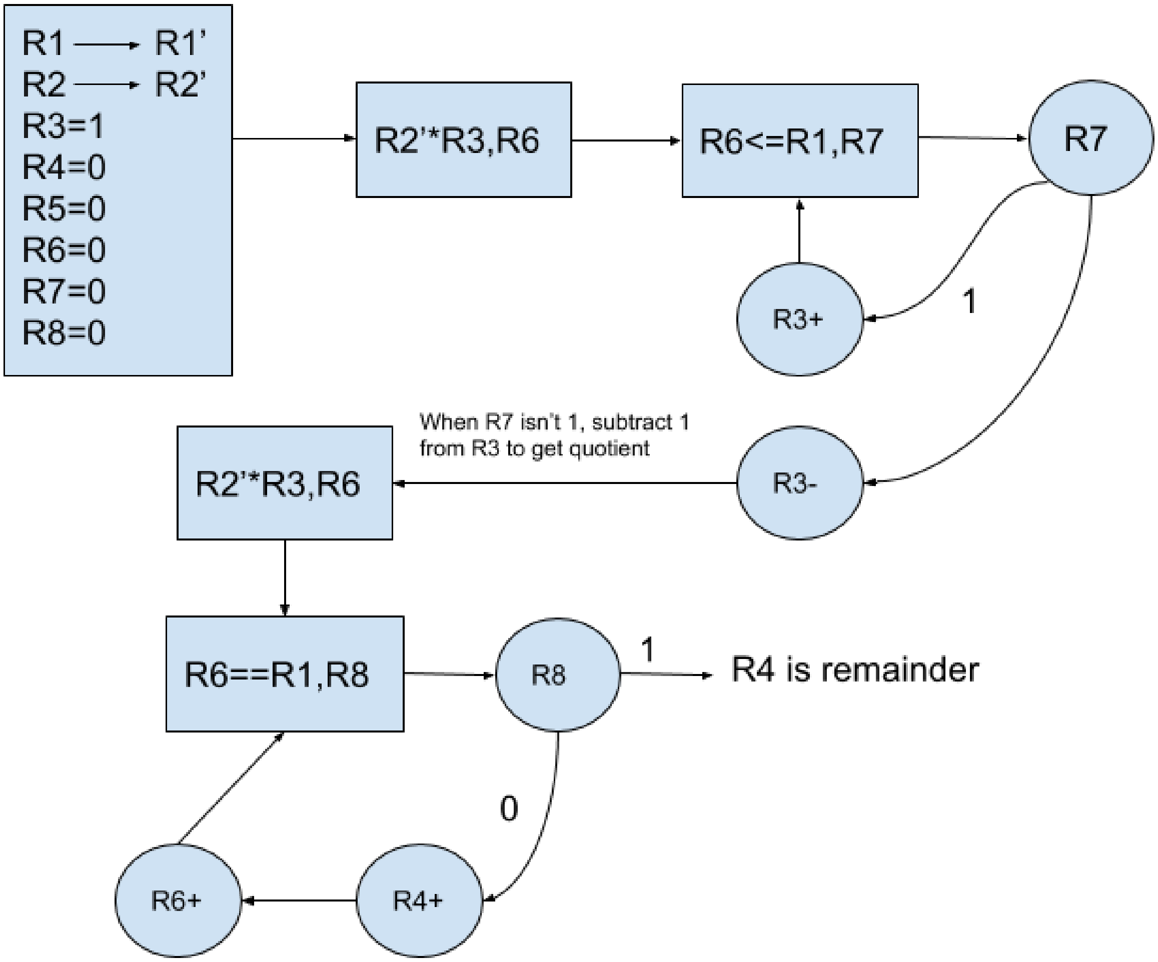
Connor Kalina

Less than or Equal
Syntax : R1<=R2,R3
Sets R3 to 1 if R1 is less than or equal to R2, otherwise 0



Connor Kalina

Division
Syntax : R1/R2,R3,R4
Divides R1 by R2, and sets the quotient in R3 and remainder in R4



R1 → R1'
R2 → R2'
R3=1
R4=0
R5=0
R6=0
R7=0
R8=0
R2'*R3,R6
R6<=R1,R7
R7
1
R3+
When R7 isn't 1, subtract 1 from R3 to get quotient
R3-
R2'*R3,R6
R6==R1,R8
R8
1
R4 is remainder
0
R4+
R6+