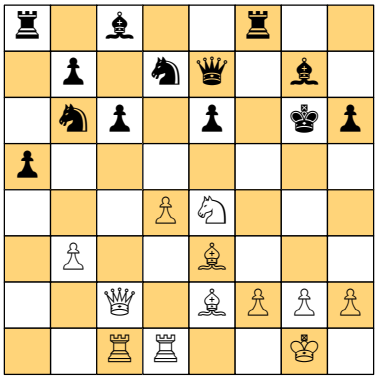
02-201 Homework 1

Fall 2015

DUE: Wednesday, September 16 in class.

Reading: Read chapters 3 and 4 of the "Introduction to Go Programming" textbook, which is online here: `http://www.golang-book.com/books/intro`

1. Chess encoding. Propose a reasonable way to encode the positions of the pieces of a chess board digitally. For example, describe a way to encode a position like this:



A chessboard has a kings, some number of pawns, queens, bishop, rooks, knights for each player (black and white). Note that there may be more than 1 queen or 2 bishops, etc. if the player has promoted a pawn. Your encoding should handle this case. You don't have to describe this at the level of bits, but rather how we could represent this in a computer that can only deal with numbers.

2. Jumping CPU. Assume your CPU can execute statements of the form:

$$Ri \leftarrow C$$

where Ri is a register and C is an integer; this sets register Ri to C. Assume you have registers $R1$ through $R6$. Your computer can also execute statements of the form:

$$Ri \leftarrow Rj \ op \ Rk$$

where Ri, Rj and Rk are registers, and op is one of $+, -, \times$. (So far these statements are the same kind as we described in lecture.)

Your CPU also has another instruction:

$$\text{Jump to line } N \text{ if register } Ri \text{ equals } 0$$

When this instruction is encountered, if Ri equals 0, the CPU "jumps" to line numbered N and starts executing from there. If Ri doesn't equal 0, then the instruction does nothing (and the CPU moves onto the next statement).

For example:

1. $R1 \leftarrow 1$
2. $R2 \leftarrow 10$
3. $R3 \leftarrow 0$
4. $R3 \leftarrow R3 + R2$
5. $R2 \leftarrow R2 - R1$
6. Jump to line 4 if $R2$ equals 0

This program will compute the sum of the integers $1, \ldots, 10$ and put it into $R3$.

(a) Modify the above program to compute the sum of the integers $1, \ldots, 100$.

(b) Modify the above program to compute the sum of the *even* integers in $1, \ldots, 100$.

(c) Modify the above program to compute the sum of the integers $5, 6, 7, 8, \ldots, 10$. Hint: you can do this by adding a line and modifying one line in the above program. (You might need to use a new register.)

(d) Recall that the first two Fibonacci numbers are $f_1 = 1$ and $f_2 = 1$ and $f_i = f_{i-2} + f_{i-1}$. Write a program using the instructions above to compute the 100th Fibonacci number.

3. Using pseudocode in the style of lectures, Write a program `ReverseInteger(n)` that takes a positive integer n, and returns the integer formed by reversing the decimal digits of n. For example:

- $1234 \rightarrow 4321$
- $20000 \rightarrow 2$
- $1331 \rightarrow 1331$

You can use the operation:

$$x \bmod y$$

which returns the remainder after x is divided by y.