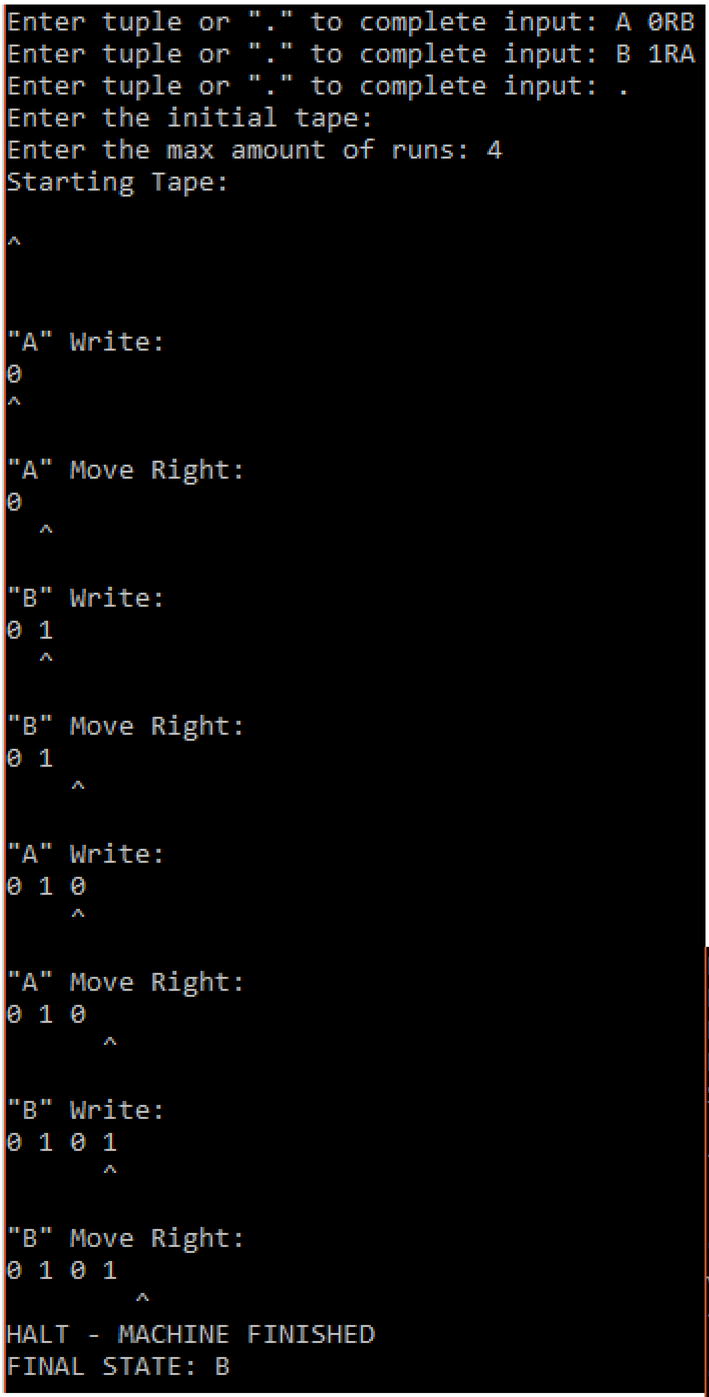
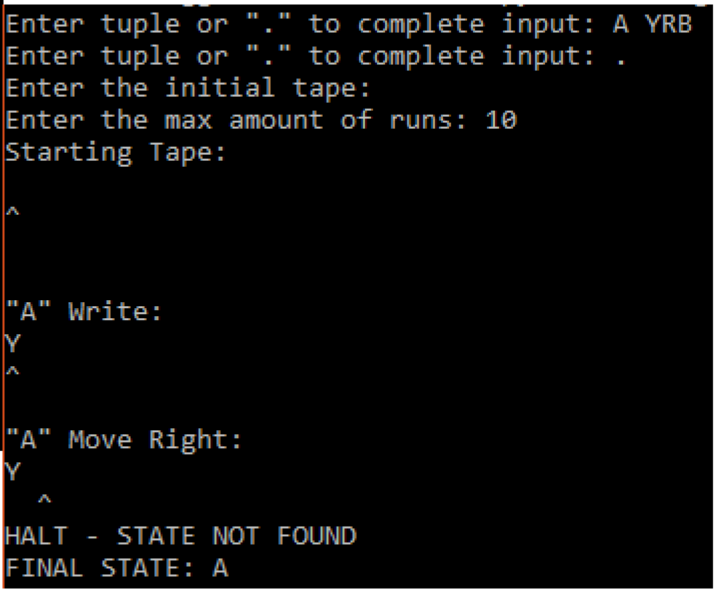
Sean Poston
CS345 Homework 3

1.



```
Enter tuple or "." to complete input: A 0RB
Enter tuple or "." to complete input: B 1RA
Enter tuple or "." to complete input: .
Enter the initial tape:
Enter the max amount of runs: 4
Starting Tape:

^


"A" Write:
0
^

"A" Move Right:
0
  ^

"B" Write:
0 1
  ^

"B" Move Right:
0 1
    ^

"A" Write:
0 1 0
    ^

"A" Move Right:
0 1 0
      ^

"B" Write:
0 1 0 1
      ^

"B" Move Right:
0 1 0 1
        ^
HALT - MACHINE FINISHED
FINAL STATE: B
```



```
Enter tuple or "." to complete input: A YRB
Enter tuple or "." to complete input: .
Enter the initial tape:
Enter the max amount of runs: 10
Starting Tape:

^


"A" Write:
Y
^

"A" Move Right:
Y
  ^
HALT - STATE NOT FOUND
FINAL STATE: A
```

Sean Poston
CS345 Homework 3

2. It seems the way I implemented the machine doesn't allow for this program to be run, or I just can't figure out how to work it out.
   If we take b as the starting tape and start in state 'o', then I don't believe that there's anyway to make these multiple statements, even with multiple 'recursive' tuples.
   Here's why:
   Our starting tape after b would be: 'ee0_0', where '_' is a space, and our starting position is the first 0.
   o wants us to move to the right one, which means our tuple would be: o 1 _ R o
   then o wants us to print x and move left: o 1 x L o
   Already we have a conflict because how does the program choose which state 'o' and read '1' to branch from?

   The only way I see that this would be possible would be to group all the same state, same read tuples into their own sub array that executes in an unchangeable, user-entered order.

Sean Poston
CS345 Homework 3

3. A decidable system, in our context, is one in which there is a method "to determine whether a given word belongs to that language or not."
A recognizable system is one that has an "algorithm that can accept a given string iff the string belongs to that language." (http://kilby.stanford.edu/~rvg/154/handouts/decidability.html)

   Because recognizable and unrecognizable are the opposites of each other, it can't be both of those at the same time.

   Goedel talks about how a there is no "consistent system of axioms whose theorems can be listed by an effective procedure (i.e., an algorithm) is capable of proving all truths" (https://en.wikipedia.org/wiki/G%C3%B6del%27s_incompleteness_theorems).

   I believe this disproves the set of all proofs being decidable because there is no way to prove "all truths."

   However, given different algorithms or proofs, the system would be able to work out if it belongs to the set or "language" as put above.

   Because recognizable is accepted, then unrecognizable must be rejected.

   Thus, the set of all proofs is recognizable, but not decidable.

Sean Poston
CS345 Homework 3

4. Computers we use run mainly using von Neumann architecture which means that a CPU communicates with RAM and input/output devices. A Turing machine has simply “one byte” of stored memory and follows a set of instructions. Logically they are quite similar because they both execute one code at a time and are then switched into a different state. The act of switching states looks differently because a computer will jump around to different memory locations and swap its registers around, whereas a Turing machine will simply execute a new set of commands given by its new state. However, they are logically the same beast; one is just much more efficient than the other.

5. The halting problem states that it’s impossible to know whether a program will terminate or run forever given an input. As in my above question, I said that a Turing machine and a computer can compute the same things. Because of this, if Alan Turing provided evidence that you couldn’t solve it, then you sure can’t solve it now.