1. This problem is recognizable but undecidable.

**Proof of recognizability**: write a machine that, when given input , simulates on input and reads the state of for every step, seeing what the value of is. If in any of the steps, then return true.

**Proof of undecidability**: Let’s design a program that takes input :

M'(x):

M\_m = M, but there's a a new line of code in the very

beginning that initiates y and sets it to 0,

everything inside main is copied to another function f,

rewrite main to only have two things:

call f and then set y to 1

M\_m(x) //run M\_m on x

Observe see that if halts, then modifies in the end and halts; and if doesn’t halt, then also doesn’t halt, and never gets to modify .

Suppose for now that there exists a program to rewrite into according to the above pseudocode specification.

Assume there exists a program that decides the Modifies a Variable problem.

Then there exists a program that decides the halting problem:

boolean haltChecker(string M, string x){

M' = rewrite(M);

return modifiesY(M',x);

}

And this would be a correct program that decides the Halting Problem if actually could decide the Modifies a Variable Problem. This contradicts the undecidability of the Halting Problem. Thus, our assumption that exists is wrong.