recap

Goal: determine if program P halts on input x

① construct a program to determine this.

```
def Test Halt (P, x):
    if P halts on x
        return true
    if P loops forever on x
        return false
```

② now construct another program

```
def Turing (P):
① if Test Halt (P, P) = true
        loop forever
② else
        halt
```

"switching the outcomes"

programs are strings so they can also be inputs.

③ analyze Turing (Turing)

if T(T) halted, we know it went into case ②

but the conditional for case ② is Turing halting on input Turing.

if T(T) looped, we know it went to case ①

but being in case ① means Turing (Turing) halted.

So Test Halt cannot exist.

recap

Easier Halting Problem?

```
def TestEZHalt(P):
    if P halts on 0:
        return true
    if P loops on 0:
        return false
```

we claim TestEZHalt works for any program!

let's see if it can exist.

trying to reduce Halting Problem to TestEZHalt

well then, we could just solve the Halting Problem.

```
def TestHalt(P, x):
    def P'(y):
        P(x)
        return
    return TestEZHalt(P')
```

running TestEZHalt will also check for us if P(x) halted or not.

We know this is not possible, so TestEZHalt cannot exist either.