Q. Determine if destructor is called.

```
                                 program
bool testHalt(String P, string x) {
    void helper() {
        Object* ob = new Object();
        P(x);
        ob.~Object();
            destructor
    }
    return testDestruct(helper, "");
}
```

TD has impl. => TH has impl.

But we know ¬(TH has impl.)

```
// true if all objects are destroyed, false otherwise
bool testDestruct(String P, string x);
```

CANNOT EXIST

1 Countability and the Halting Problem

Prove the Halting Problem using the set of all programs and inputs.

a) What is a reasonable representation for a computer program? Using this definition, show that the set of all programs are countable. *(Hint: Python Code)*

Finite length strings from finite alphabet.

b) We consider only finite-length inputs. Show that the set of all inputs are countable.

Finite length strings from finite alphabet.

c) Assume that you have a program that tells you whether or not a given program halts on a specific input. Since the set of all programs and the set of all inputs are countable, we can enumerate them and construct the following table.

$P_1(x_1)$ halts
$P_1(x_2)$ loops
$P_2(x_3)$ loops

	x_1	x_2	x_3	x_4	...
p_1	H	L	H	L	...
p_2	L	L	L	H	...
p_3	H	L	H	L	...
p_4	L	H	L	L	...
⋮	⋮	⋮	⋮	⋮	⋱

An H (resp. L) in the ith row and jth column means that program p_i halts (resp. loops) on input x_j. Now write a program that is not within the set of programs in the table above.

```
def T(x_i):            TestHalt(P_i, x_i)
    if P_i(x_i) halts:
        loop {}
    else:
        return; // halt
```

TH exists => T exists
But we know that ¬(T exists).

$((P \Rightarrow Q) \wedge (\neg Q)) \Rightarrow \neg P$

d) Find a contradiction in part a and part c to show that the halting problem can't be solved.

If T is not in the table, we have an uncountable set of progs.

2 Fixed Points

Consider the problem of determining if a function F has any fixed points. That is, given a function F that takes inputs from some (possibly infinite) set $\mathscr{X}$, we want to know if there is any input $x \in \mathscr{X}$ such that $F(x)$ outputs x. Prove that this problem is undecidable.

Assume TestFix(F)

```
def F1(x):
    return x**2

def F2(x):
    return x+2
```

$\text{TestFix}(F_1) == \text{True}$

$F_1(0) = 0$

$\text{TestFix}(F_2) == \text{False}$

$F_1(x) \neq x \ \forall x \in \mathscr{X}.$

```
def TestHalt(P, x):
    def helper(y):
        P(x)
        return y
    return TestFix(helper)
```

$\exists TF \Rightarrow \exists TH$

$\neg(\exists TH)$

$\therefore \neg(\exists TF)$.

$P(x)$ halts $\Rightarrow$ helper(y) == y

TF(helper) is True $\Leftarrow$ helper has a fixed point

$P(x) \neg$halt $\Rightarrow$ helper(y) does not return anything

TF(helper) is False $\Leftarrow$ helper has no fixed points

```
def T(xi):
    if TF(Fi, xi) == True:
        return

def G(xi):
    if Fi(xi) == xi:
        return something from X \ {xi}
    else:
        return xi
```

	x_1	x_2	x_3
P_1	Fix	N	F
P_2	F	N	N
P_3	F	F	F

3 Computability

Decide whether the following statements are true or false. Please justify your answers.

(a) The problem of determining whether a program halts in time 2^{n^2} on an input of size n is undecidable.

(b) There is no computer program `Line` which takes a program P, an input x, and a line number L, and determines whether the L^{th} line of code is executed when the program P is run on the input x.