Quiz 5: Uncomputability

Collaboration: This quiz is open note but individual. You may use any resources that were provided to you by the course, or that you had recorded as part of your own notes prior to when you first viewed this quiz. You may not discuss this quiz with your cohort-mates, class-mates, tutor, friends, family, or anyone else except the course staff (who will answer clarification questions only).

Problem 1: Rice's Theorem

Rice's theorem states that any non-trivial semantic property of a Turing Machine is uncomputable. A property of Turing Machines is semantic if its truth/falsehood will match for any two Machines which compute the same function. A semantic property is trivial if it is true for all Turing Machines, or else false for all Turing Machines.

For each subproblem, indicate whether or not Rice's Theorem applies. If it applies, explain why, and answer if the problem is computable or uncomputable. If it does not apply, just indicate why it doesn't apply (it is not necessary to determine whether or not it is computable if Rice's theorem does not apply).

(a) Given the description of a Turing Machine, does that machine ever exit its start state when running on input 011010?

(b) Given the description of a Turing Machine, Does there exist an input for which that Turing machine will output 011010?

(c) Given the description of a Turing Machine, does that machine return 1 when running on input 011010?

(d) Given the description of a Turing Machine, does that machine halt when running on input 011010? (Note from Nate: this one may not be as straightforward as you might think at first. Think extra hard about the definition of a semantic property compared to the definition of the language of a machine.)

Problem 2: Bounds Checking

I a programmer tries to access an invalid index in a python list then the program will throw a **list index out of range** error. For example, the following python program would produce a **list index out of range** error since a list of length 4 has no index 4.

```
my_list = [0,1,2,3]
print(mylist[4])
```

This code, however, will not:

```
my_list = [0,1,2,3]
print(mylist[0])
```

Using a reduction proof, show that, in general, the problem of determining whether a given Python program will throw a **list index out of range** error is not computable. That is, show that there does not exist an always-halting Turing Machine which, when given a python program, returns 0 if the program does not throw a **list index out of range** error, and will return 1 if it does.

For this problem, you should assume an "idealized" version of Python with no other implementation limits.

