CS 110 – Creative Problem Solving in Computer Science

Stevens Institute of Technology © 2017

Exam 2

Instructor: Adriana Compagnoni

Fall 2017

STUDENT NAME:

HONOR PLEDGE:

Remarks

- This exam is about solving problems by writing programs in a high level language called Python, and using Python datatypes (lists, strings, numbers, booleans,..), assignment, if-elif-else, for and while loops, etc.
- This exam also tests your problem solving abilities, and how you systematically divide a problem into a sequence of steps.
- This exam also tests your ability to demonstrate the dynamic behavior of programs that include conditional execution, and looping by describing their behavior and output.
- This exam is closed notes, closed books, and closed laptops. The use of any electronic devices is stricktly prohibited.
- Please refrain from communicating with other students during the exam.
- Please do not forget to put your name **on every page** you submit.
- This exam is timed. You have 50 minutes to answer all the questions. Please take a minute to read through the exam and budget your time.

Good luck!

Exercises

1. (35 points) Consider the following Python function `pair`(a,b) that returns True if a and b are a DNA base pair, and False otherwise.

```
def pair(a,b):
    if   (a =='A' and b == 'T'):
        return True
    elif  a =='T' and b == 'A':
        return True
    elif  a =='C' and b == 'G':
        return True
    elif  a =='G' and b == 'C':
        return True
    else:
        return False
```

   Write a Python function `matching_base_pairs(s1,s2)` using loops that counts the number of matching base pairs in two DNA strings of the same length.

   Test Cases:

```
>>> matching_base_pairs('ATTC','TAAG')
4
>>> matching_base_pairs('ATAC','TAAG')
3
>>> matching_base_pairs('ATA','TAAG')
ATA and TAAG DNA strings do not have the same length.
>>>
```

2. (30 points) Remember the function `how_many_times`(x,lst) from the Mastermind program. Write a hand trace or execution of `how_many_times`(1,[2,1,1]). Write enough details to show you understand the execution of the program.

```
def how_many_times(x,lst):
    count = 0
    for y in lst:
        if x == y:
            count = count + 1
    return count
```

3. (35 points) Write a recursive version of the previous function, `how_many_times_rec(x,lst)`.

```
>>> how_many_times_rec(1,[2,1,1,0])
2
>>> how_many_times_rec(1,[])
0
>>> how_many_times_rec(3,[2,1,1,0,2])
0
>>>
```