CS 110 A – Creative Problem Solving in Computer Science

Stevens Institute of Technology © 2016

Homework 8

Instructor: Adriana Compagnoni

This homework is about recursion in Python.

Rewrite these old friends using recursion instead of for and while loops.

Exercises

1. Read Chapter 5 of Computing for Biologists.
2. (a) (20 points) Write a recursive Python function `replace`(x, y, lst) that returns the result of replacing every x by y in lst.

```
>>> replace(1,2,[1,2,3,1])
[2, 2, 3, 2]
>>> replace(1,2,[])
[]
>>> replace (4,5,[1,2,2])
[1, 2, 2]
>>>
```

   (b) (15 points) Write a hand trace or execution of `replace (4,5,[1,4,2])`.

3. (30 points) Write a recursive Python function `first_n_primes`(n,lst) that given a list of numbers lst prints out the first n prime numbers in the list, and prints error messages, when it cannot complete the task.

   Test cases:

```
>>> first_n_primes(6,[4,6,8,10,21,76,49])
I didn't find any primes.
>>> first_n_primes(6,[11,21,3,4,5,62,7,8,49,10,2])
11
3
5
```

```
7
2
I only found 5 primes.
>>> first_n_primes(3,[11,21,3,4,5,62,7,8,49,10,2])
11
3
5
>>>
```

4. (35 points) This exercise is part of the Mastermind game problem we saw in class. Write a recursive Python function `findBlack`(code, guess, colors), that finds all the indices where code and guess are identical (code[i] == guess[i]), and also keeps count of how many matches it found for each color. The function returns a list with two elements. The first element is a list containing the string "dummy" if code and guess do not match, and "black" otherwise. The second element is a list containing the number of matches for each color.

```
>>> findBlack([1,2,3],[3,2,1],6)
[['dummy', 'black', 'dummy'], [0, 0, 1, 0, 0, 0]]
>>> findBlack([1,1,2,4,4,5],[1,1,5,4,4,0],6)
[['black', 'black', 'dummy', 'black', 'black', 'dummy'], [0, 2, 0, 0, 2, 0]]
>>> findBlack([1,1,2,4,4,5],[1,1,5,4,4,5],6)
[['black', 'black', 'dummy', 'black', 'black', 'black'], [0, 2, 0, 0, 2, 1]]
```

Course Outcomes

This homework is an assessment instrument for the following Course Outcomes.

1. **Problem solving** - Systematically divide a problem into a sequence of steps. (BS-CS B analyze) (BS-CyS B analyze)
2. **Programming** - Compose a solution to a problem using a high-level language. (BS-CS C design) (BS-CyS C design)
3. **Execution** - Demonstrate the dynamic behavior of programs that include conditional execution, looping, and recursion by describing their behavior and output. (BS-CS A apply)