BRONX COMMUNITY COLLEGE

of The City University of New York

DEPARTMENT OF MATHEMATICS and COMPUTER SCIENCE

CSI 33

Midterm Exam Review

1. You will be given about five True/False and Multiple Choice questions
2. You will be asked to find asymptotic running time in theta-notation of a given program.

   see page 36/8 (it is from HW assignment)
3. Review *Linear Search*, *Binary Search*, *Selection Sort*, *Merge Sort*, along with their run time complexity. You need to remember how each of the algorithms works!
4. Review the notions of encapsulation, polymorphism and inheritance.
5. Review `Python list`, `Linked Lists`, `Stack` and `Queue`, as well as `Card`, `Deck`, `Hand` **ADT**s. You need to remember their implementations, and should know the run-time complexities of their methods.
6. An example question:
   The integers $20, 12, 7, 14, 2, 5, 3$ and 8 are inserted in that order into a container object. Give the order in which these values are retrieved, if the container is
   (a) a stack,
   (b) a queue.
7. Review how to write unit tests.

   See HW3 (and the class materials), in-class work for Sections 5.1 - 5.2, see the textbook.
8. Review Python memory model with pictorial representations.

   See Sections 4.2-4.3 lecture materials
9. An example question for pictorial representation of Python memory:
   Give the output of the following program and a pictorial representation of the memory at the stages 1, 2, and 3.

```
def f1(a,b,c):
    a.append(b)
    a.append(c)
    a = b+c
    b += 10
    print(a,b)
    return a

def main():
    x=[1]
    y,z = 10,20
    y = f1(x,y,z)
    print(x,y,z)
```

10. Review post-fix, infix and pre-fix notations. You might be asked to evaluate an expression written in one of the notations, or to re-write expression written in one notation in the other one.

11. Review `copy()` and `deepcopy()` methods and their use.

12. Review recursion

13. Example questions:

    (a) trace `recPower(3,6)` and figure exactly how many multiplications it does (see p.214/4)

    (b) Define a recursive function that produces the list of all possible bit strings of length n. The function should get a positive integer n as a parameter, and should return the list.

14. Review the "interesting programs":

    balanced parentheses, anagrams, fibonacci numbers, palindromes, post-fix notation expression evaluation

15. Review iterators (Section 4.5)