NAME____

BRONX COMMUNITY COLLEGE
of the City University of New York
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

CSI33 **Midterm Exam (Sample)**

Directions: Answer all questions. You may refer to the textbook, handouts, or notes taken in class, but you may not discuss your work with other students. Try to answer each question in the space provided; use additional paper if necessary.

1. (30) Draw the abstract syntax tree for this postfix expression:
   `5 2 - 3 * 1 4 + *`
   Next, evaluate the expression by processing each token, going left-to-right, using stack operations. Draw the stack after each token is read (include the operation symbols), labeling each drawing with the token just seen.

2. (15) Draw the box-and-arrow diagram of Python memory after each of the following three Python programs is executed:

   a.
   ```
   List1 = [[[1, 2], 3],4]
   List2 = List1
   ```

   b.
   ```
   from copy import *
   List1 = [[[1, 2], 3],4]
   List2 = copy(List1)
   ```

c.

```
from copy import *
List1 = [[[1, 2], 3],4]
List2 = deepcopy(List1)
```

3. (20) This problem uses this definition of the ListNode class:

```
class ListNode(object):
  def __init__(self, item = None, link = None):
    self.item = item # data
    self.link = link # link
```

a. Using this class definition, write Python code that creates a linked list of nodes containing integer values beginning with 10 followed by 21 and then 2013. That is, the node containing 10 will be the "head" node. (In your code, this node should be the final value of the variable `head`. Do not use class methods for a Linked List class; just use variable names to reference the `ListNode` objects you use.) After each line of code, draw the box-and-arrow diagram for the list structure so far.

b. Next, write Python code that prints "10-21-2013" by traversing this list structure using a `while` loop.

4. (20) Perform a Θ-analysis for the running time of each of the following blocks of Python code. You must show the calculation of how many units of time each code block takes to run. Use the variable n as the size of the input.

a.

```
i = n
while i > 0:
  i = i-2
  print(i)
```

b.

```
for i in range(n):
  j = n
  while j > 0:
    print (j)
    j = j/2
```

5. (15) The **Lucas numbers** are $\{2, 1, 3, 4, 7, \ldots\}$ where 2 is the "0^{th}" term, the "first" term is 1 and the n^{th} term is the sum of the $n-1^{st}$ and $n-2^{d}$ terms thereafter. Write a recursive `Python` function `lucas(n)` that calculates the n^{th} term in this sequence. The function should give 2 for `lucas(0)`, 1 for `lucas(1)`, 3 for `lucas(2)`, and so on.