EECE 230 Introduction to Computation and Programming, Sections 4,5,6, and 7
Quiz II

April 6, 2019

- The duration of this exam is 2 hours and 55 minutes. Keep in mind that you need around 10 minutes at the end of the duration of the exam to submit your answers. It is your responsibility to make sure your files are correctly submitted.
- The exam consists of 5 problems for 200 points
- You can use all the material in the exam zip file on moodle (lecture slides, source code, programming assignments, and solutions). Once you download the zip file, moodle will be disconnected.
- At the end of the exam, moodle will reopen for exam submission. If you would like to submit your work before the end of the exam, please talk to the proctors for instructions.
- You are asked to submit a single zip file containing your `Python` files (ending with `.py` extension). Failure to do so may lead to a failing grade on the exam. It is your responsibility to make sure your files are correctly submitted.
- You are **NOT** allowed to use the **web**. You are not allowed to use **USB's** or files previously stored on your machine.
- If you get caught violating the above rules or if you communicate with a person other than the exam proctors during the exam, you will immediately get zero and you will be referred to the appropriate disciplinary committee.
- Cell phones and any other unauthorized electronic devices are absolutely not allowed in the exam rooms. They should be turned off and put away.
- The problems are of varying difficulty. Below is a rough ordering estimate of the problems in order of increasing difficulty.
  - Level 1 (122 points): Problems 1, and nonefficient solutions of Problems 2, 3, and 4
  - Level 2 (58 points): Efficient solutions of Problem 2, 3, and 4
  - Level 3 (20 points): Problem 5
- Detailed comments are worth partial credit.
- Plan your time wisely. Do not spend too much time on any one problem. Read through all of them first and attack them in the order that allows you to make the most progress.
- Good luck!

Problem 1 (45 points). Read file and plot: EECE 230 is fun and easy

Implement the function `readFileAndPlot(fileName)`, which takes as input argument a string `fileName`, opens the file named `fileName` and plots its content as detailed below. We assume that the file contains data of two graphs formatted as follows:

- The first line is the figure title.
- The second line is the x-axis label.
- The third line contains, say, n numbers separated by spaces representing the x-coordinates of both graphs.
- The fourth line is the label of the first graph
- The fifth line contains n numbers separated by spaces representing the y-coordinates of the first graph.
- The sixth and seventh lines are similar to the fourth and sixth, but correspond to the second graph.

You are not asked to check if the file format is as above: assume that this is the case. Your function is supposed to plot the two graphs in two different colors on the same figure, while showing the title, the x-axis label, and the legend.

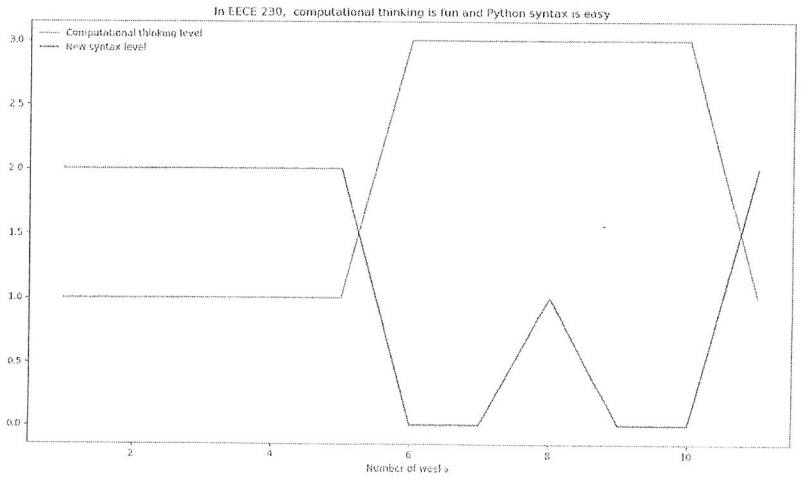
If `nameHandle` is a file handle, we know from class and the assignments how to read the lines one by one using a for loop: `for line in nameHandle`. An alternative method, which is more suitable for this problem, is via the `readline` method. After opening the file, invoking `nameHandle.readline()` returns the first line of the file. Invoking `nameHandle.readline()` again returns the second line, and so on.

Any correct solution is worth full grade.

Test it on the file named "data.txt" consisting of:

```
In EECE 230,  computational thinking is fun and Python syntax is easy
Number of weeks
1 2 3 4 5 6 7 8 9 10 11
Computational thinking level
1 1 1 1 1 3 3 3 3 3 1
New syntax level
2 2 2 2 2 0 0 1 0 0 2
```

You should get the following figure:



Submit your solution in a file called Prob1.py. Inside the file, include your name and ID number (as comments).

Problem 2 (45 points). Check if matrix has equal elements

Write a function `equalElements(M)`, which given an $m \times n$ matrix M, checks whether or not M has equal elements. That is, the function checks whether or not there are $(i_1, j_1) \neq (i_2, j_2)$, where $0 \leq i_1 < m$, $0 \leq j_1 < n$, $0 \leq i_2 < m$, and $0 \leq j_2 < n$, such that $M[i_1][j_1] = M[i_2][j_2]$. If so, the function should return the tuple of tuples $((i_1, j_1), (i_2, j_2))$, for any such (i_1, j_1) and (i_2, j_2). Otherwise, the function should return a tuple of tuples $((-1, -1), (-1, -1))$. If the matrix has more than two equal elements, the indices of any pair of equal elements are a valid answer.

A correct solution is worth 27 points. For full grade, solve it efficiently. $\Theta(mn)$

Test program:

```
import numpy as np
M1 = [[1,2],[3,4]]
M2 = [[1,2],[3,1]]
M3 = [[1,3,0,5],[2,5,2,-1],[5,6,-2,6]]
M4 = [[1,3,0,5],[20,50,2,-1],[51,61,-2,16]]
for M in (M1,M2,M3,M4):
    print(np.matrix(M))
    print(equalElements(M),"\n")
```

Output:

```
[[1 2]
 [3 4]]
((-1, -1), (-1, -1))

[[ 1  2]
 [3  1]]
((0, 0), (1, 1))

[[ 1  3  0  5]
 [ 2  5  2 -1]
 [ 5  6 -2  6]]
((0, 3), (1, 1))

[[ 1  3  0  5]
 [20 50  2 -1]
 [51 61 -2 16]]
((-1, -1), (-1, -1))
```

Submit your solution in a file called Prob2.py. Inside the file, include your name and ID number (as comments).

Problem 3 (45 points). Find closest element

Implement the function `findClosest(L,x)`, which given a list of numbers L *sorted* in nondecreasing order, and a number x, finds the element e in L which is closest to x. That is, the desired e is an element of L such that $|e - x| \leq |e' - x|$ for all elements e' of L. If the list is empty, your function should raise an assertion with an error message "Empty list!".

For example, for $x = 7$ and $L = [2, 4, 8, 10]$, `findClosest([2,4,8,10],7)`=8 because: $|2 - 7| = 5$; $|4 - 7| = 3$; $|8 - 7| = 1$; and $|10 - 7| = 3$. Hence, 8 is the closest element of L to 7.

Any correct solution is worth 23 points. For full grade, solve it in $O(\log(n))$ time.

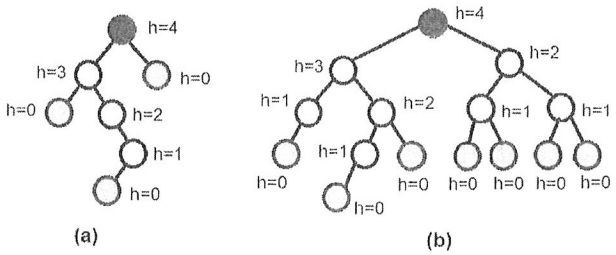


Figure 1: This figure shows two binary trees with the height of each node. Note that the root is black and leaves are gray. In (a), the tree is not balanced: for example the node with height 2 does not have a left child; the children of the root have heights 3 and 0 so they differ by more than one; etc. In (b), the tree is balanced: the height condition is satisfied at all nodes with height ≥ 2.

Test program/output:

```
print(findClosest([1,5,7],4))                                    | 5
print(findClosest([3,9,10,11],10))                               | 10
print(findClosest([3,8,14],11)) # Both 8 and 14 are valid answers | 14
print(findClosest([-2,7,10,18],2))                               | -2
print(findClosest([-2,7,10,18],20))                              | 18
print(findClosest([-2,7,10,18],-3))                              | -2
```

Submit your solution in a file called Prob3.py. Inside the file, include your name and ID number (as comments).

Problem 4 (45 points). Minimum distance between two lists

Implement the function `minimumDistanceL(A,B)`, which given two lists of numbers A and B, returns the *minimum distance* between any pair of elements where one element is chosen from A and the other from B. If either of the lists is empty, the functions raises an assertion error.

For example, if $A = [10, 7, 25]$ and $B = [13, 18, 9]$, then `minimumDistanceL(A,B)`$=1$ since: $|A[0] - B[0]| = |10 - 13| = 3$; $|A[0] - B[1]| = |10 - 18| = 8$; $|A[0] - B[2]| = |10 - 9| = 1$; $|A[1] - B[0]| = 6$; $|A[1] - B[1]| = 11$; $|A[1] - B[2]| = 2$; $|A[2] - B[0]| = 12$; $|A[2] - B[1]| = 7$; and $|A[2] - B[2]| = 16$. Hence, the shortest distance is 1.

Any correct solution is worth 27 points. For full grade, solve it efficiently.

Test program/output:

```
print(minimumDistanceL([10,7,25],[13,18,9])) # A[0]=10 and B[2]=9          | 1
print(minimumDistanceL([56,67,22,113],[33,47,12,77,-2])) # A[0]=56 and B[1]=47 | 9
print(minimumDistanceL([11.3,19.4,-3.5,0,15,10],[6,-7,-9,25])) # A[2]=-3.5 and B[1]=-7 | 3.5
```

Submit your solution in a file called Prob4.py. Inside the file, include your name and ID number (as comments).

Problem 5 (20 points). Number of balanced trees of a given height

In this problem, by a tree we mean a tree with a given root. The height of a node v in a tree is the number of edges from v to the farthest descendant leaf. The height of the tree is the height of the root.

A binary tree is a tree where each node has at most two children. Additionally, the children have left and right labels (that is, the first two trees in Figure 2.b are distinct because the first one consists of a root with a *left* child, and the second is a root with a *right* child). See Figure 1.

A balanced binary tree is a binary tree T such that for each node v of T of height at least 2, v must have a left and right child whose heights differ by at most one. See Figure 1 for an example.

Implement the function `numberOfBalancedBinTrees(h)`, which given h, returns the number of all possible balanced binary trees of height h. See Figure 2.

Any correct solution is worth 15 points. For full grade, solve it efficiently. Note that we are not interested in finding all balanced trees of height h; we are only interested in computing their number.

(Hint: Find a recurrence)

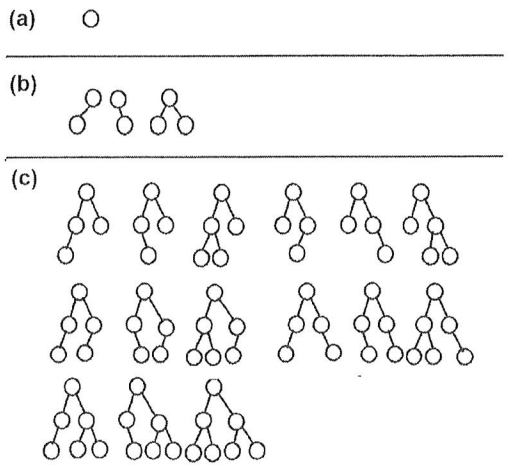


Figure 2: This figure shows the only balanced binary tree of height $h = 0$ in (a), all 3 balanced binary trees of height $h = 1$ in (b), and all 15 balanced binary trees of height $h = 2$ in (c).

Test program:

```
for h in [0,1,2,6,9,12]:
    print("numberOfBalancedBinTrees("+str(h)+"):",numberOfBalancedBinTrees(h))
```

Output:

```
numberOfBalancedBinTrees(0): 1
numberOfBalancedBinTrees(1): 3
numberOfBalancedBinTrees(2): 15
numberOfBalancedBinTrees(6): 141106591466142946875
numberOfBalancedBinTrees(9): 157173169311826015719601810646022104985769563616688776984
347668596538511693043974144618633302848944843830885084323887267270084212867423967889 64
5644664764404296875
numberOfBalancedBinTrees(12): 372415102498722032705526658376999059377413908588911479930
270176183249991420320993333416563465715307530243436568982960582093563925821771444901064
763712998158628865388617285325292370914778767587691713736853053156051497911154236003517
308630975412458620660768065698712001683749368689689334660195441274710488106920038822849
969617554649396939125238300675287550342890914483676776311668560614985115749152245368304
702851738333826162027629395780490439320331833224760141354038989634813477703350141674578 3
473962183748610061876349310924339579045659798498004132963954553314444860900411802078295
384259140823218882142524479448794749622034179533146012275089132563784539394794808206564
605959988987055820867205672309852532668001537260262162597783052796532505099880555684552
934855413094665408367476462466617029678586211161521772380310838522377006237513697040514
492539170480780825251440488499754789024785033820539336784903869225919818547645802139117
975833377549202085586411349881461230425616823658148271428369743351006899566289389390281
712744325492957842796907630558740263040869518458489306966799003964126751133444130816392
241243944527810152357749357111548529192086548241605742863584556040563936892968323276161
635936123204557980712387970221953043886742238739172759394153826295603693097291397862136
363983154296875
```