15-110 Practice Exam 2

Show work when needed, it can be used for partial credit! Also note that these questions are a rough estimate and are compiled by TA's who have not seen the exam. Topics covered in class are fair game even if they are not on this exam.

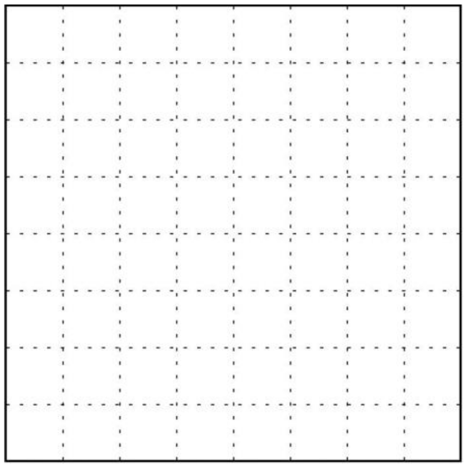
Short Answer/Multiple Choice/Fill in the blank/Code Tracing:

Graphics

The box below is the canvas, with a width and height of 400px. Each small box is 50px by 50px. You may assume a tkinter canvas has already been run. Draw the graphics made by this code.

```
def drawCT1(canvas, w, h):
      a = 300
      canvas.create_rectangle(w/4, h/4, 50, 50)
      canvas.create_rectangle(w/5, h/5, 200, 200)
      canvas.create_oval(w/2, h/2, a, a)
      canvas.create_text(w/2, h/2, text = "15-110", anchor = "ne")

drawCT1(canvas, 400, 400)
```



Possibly the Worst Hash Function in Existence (That's Not Just a Constant Function)

Someone has the brilliant idea of using the length of a string as its hash function. For example, hash("a") -> 1, while hash("aaaa") -> 4. Why might this be a bad idea?

How Fast Does It Run?

Compute the Big-O runtime of each of the following functions. Tip: listing the runtime of each line in the function may help.

```
def bigO1(s):                                                   Big-O
     x = 0                                                      _____
     for c in s:                                                _____
          nextC = chr(ord(c) + 1) # chr() and ord() are O(1)    _____
          x += s.count(nextC) # Note that s.count() is O(n)     _____
     return x                                                   _____
```

Total Big-O:__________

```
def bigO2(L):                                                   Big-O
     n = len(L)                                                 _____
     newL = []                                                  _____
     for i in range(n**3):                                      _____
          if i % 5 == 0:                                        _____
               newL.append(i)                                   _____
     return newL                                                _____
```

Total Big-O:__________

I See Trees Of Green… And Other Ones Too

Indicate what the following function will print for the given binary tree input below.

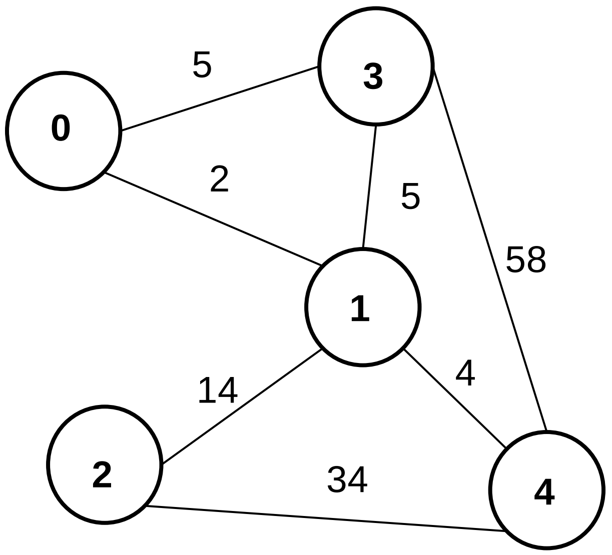
```
def greenTrees(tree):
    numGreen = 0
    if tree["left"] != None:
        numGreen += greenTrees(tree["left"])
    if tree["value"] == "green":
        numGreen += 1
    if tree["right"] != None:
        numGreen += greenTrees(tree["right"])
    print(numGreen)
    return numGreen

tree = {"left" : {
                    "left" : {
                                "left" : None, "value" : "red",
                                "right": None
                             },
                    "value" : "green",
                    "right" : {
                                "left" : None, "value" : "green",
                                "right": None
                              }
                  },
         "value" : "green",
         "right" : {
                    "left" : None,
                    "value" : "blue",
                    "right" : {
                                "left" : None, "value" : "green",
                                "right" : None
                              }
                   }
        }

greenTrees(tree)
```

Graphs and All That

Represent the given graph as an adjacency matrix. Then, output the order in which all the nodes will be visited with BFS and DFS starting at node 1, assuming you search for a node not in the graph.



Adjacency Matrix:

BFS Output:

DFS Output:

Polynomial or Nah

True or False: Some problems in P are intractable.

True or False: NP means “not P” and therefore a problem is in NP if it is not in P.

What is the significance of the complexity class NP-complete?

Free Response (Code Writing):

Find My Ascii, Recursively

Write the function findMyAscii such that, given an input string s, it recursively calculates the sum of the ascii codes of each character in the input string.

Run An Election!

Given a list of votes, return an announcement of the winner or that there was no winner.

```
# runElection(["Red", "Blue", "Red", "Blue", "Blue"]) -> "Blue wins!"
# runElection(["Black", "White", "White", "Black"]) -> "No winner..."
```

Your Password is Safe With Me

Write a function called **IsValidLogin** with three arguments:

- **users:** a dictionary of username keys and password values (we will give this to you)
- **username**: a string for a login name
- **password:** a string for a password.

IsValidLogin should return True if the user exists and the password is correct and False otherwise.

For example:

```
users =
{ "gshandar" : "ilovepitbull",
"rtn" : "feelsbadman",
"rzh1" : "grandmavibes",
"ela" : "imawesome",
"krivers": "stellaisthecutestdoggo",
"srosenth": "imsuperwoman"
  }

isValidLogin(users, "ela", "feelsbadman") -> False
isValidLogin(users, "srosenth", "imsuperwoman") -> True
```