15-110 Final Exam Practice Problems

List Methods and Recursion

1. **Code Writing:** Write a recursive function that takes in a 2-D list of numbers and returns a 2-D list of all the lists that have a even sum.
   For example `even2D([[1,2,3], [5,5,5], [1], [6,4]])` would return `[[1,2,3], [6,4]]`
   Code:
```
def even2D(L):
    if L == []:
        return []
    else:
        if sum(L[0]) % 2 == 0:
            return [L[0]] + even2D(L[1:])
        else:
            return even2D(L[1:])
```

2. **Code Writing:** Write a recursive function that takes in a list of strings and a subset phrase and returns a list of strings that contain that subset.
   For example `subset(["apple", "cap" , "orange" , "lap"], "ap")` would return `["apple", "cap", "lap"]`
   Code:
```
def subset(L, sub):
    if L == []:
        return []
    else:
        if sub in L[0]:
            return [L[0]] + subset(L[1:], sub)
        else:
            return subset(L[1:], sub)
```

Trees and Recursion

1. **Code Writing:** You are given a tree which is **not** a binary tree, and each node can have an arbitrary number of children. Each node has a numerical value.

   Your task is to find a path through the tree starting from the root to a leaf such that at every single step along the way, you are traversing down the node with the **lowest** value. For example, if a node has children with values of 1, 2, 3, 4, 5, you would choose

to move down the path following the node with value 1. Assume that no node has duplicate child values, and the child values **need not be sorted**. Assume that trees are implemented as dictionaries, and that the children of each node is stored in a **list**, accessed by the key 'children'. The value for each node is accessed through the key 'value'. The function should return the list of nodes traversed in order, starting with the root node and ending with the leaf node.

Code:

```
def minTraversal(t):
      if len(t["children"]) == 0:
            return [t["value"]]
      else:
            minValIndex = t["children"].index(min(t["children"]))
return [t["value"]] + minTraversal(t["children"][minValIndex])
```

2. **Code Writing:** You are given a binary tree. Your task is to reverse the binary tree, which means switching the left and right branches. Assume trees are implemented as nested dictionaries, children are stored in lists, and the value and children can be accessed using the keys 'value' and 'children' respectively. The function should return the inverted binary tree, and the inversion should occur at every level.

   Code:

```
def invertBinaryTree(t):
      if len(t["children"]) == 0:
            return t
      else:
            return {"value" : t["value"], "children" :
      [invertBinaryTree(t["children"][1]),invertBinaryTree(t["ch
      ildren"][0])]}
```

Simulation - Randomness & Monte Carlo

1. **Code Writing:** Write the following Monte Carlo functions:
   a. Write a function runTrial() that takes in 0 inputs and simulates rolling 2 five-sided dice. The function should return True if the sum of the dice is less than 10 AND the first dice roll is strictly greater than the second dice roll. Remember to import the proper libraries when writing this function!

      Code:

```
import random
def runTrial():
      roll1 = random.randint(1, 5)
      roll2 = random.randint(1, 5)
      if roll1 > roll2 and roll1 + roll2 < 10:
            return True
      return False
```

   b. Now, use the function from part (a) to write the Monte Carlo function

getExpectedValue(trials) that takes in an integer number of trials. It should return the probability that with 2 five-sided dice, the first roll is greater than the second and their sum is less than 10.

Code:

```
def getExpectedValue(trials):
    count = 0
    for trial in range(trials):
        if runTrial():
            count+=1
    return count/trials
```

2. **Multiple Choice:** Which is false about randomness?
    a. By the Law of large numbers, the average of the results of 1000 experiments will be the true expected value
    b. Monte Carlo methods use repeated simulations to simulate the law of large numbers
    c. Randomness in Python is not true randomness
    d. The random library in Python includes functions such as random.randint(x,y) that can pick a number between a range.

    Answer: A - The law of large numbers states that the expected value will be *approached* after a larger number of trials. There is no set number of trials, like 100, whose average will be equal to the true expected value.

Simulation - Model/View/Controller

1. **Code Writing:** You are given the following starter file with a model and view built already. The size of the window is 400 and you are given a blue "balloon" with radius 20 positioned in the center of your screen.

    Starter File:
    https://drive.google.com/file/d/1INSmlV_rgJPysun66m748vTqP0Mi5Wm4/view?usp=sharing

    a. Implement the keyPressed function so that pressing the '1' key will 'blow up' the balloon. In other words, pressing the 1 key should increment the radius of the circle by the given pumpValue. Implement functionality that will allow the user to move the balloon around using the up, down, left, and right arrow keys as well (increment the position of the x and y coordinates of the circle by the pumpValue).
    b. Implement the isInBounds function so that the balloon will 'pop' or reset to radius 20 as soon it circumscribes the window. Also add functionality that will prevent the user from moving any portion of the circle out of the window. Instead of allowing them to move outside of the window, have the balloon reposition itself to the middle of the window by manipulating the appropriate coordinates.

    Code:
    https://drive.google.com/file/d/1Mr3hIR0U-7DA1PKGW-eJdUykEaa_l0vi/view?usp=sharing

Runtime/Big O

1. **Short Answer:** Consider the following function:

```
def f(x): #assume x is a positive integer
      #section 1
      count=0
      i=1
      while i=<x:
            i *= 4
            count += 1
      #section 2
      newcount=0
      for i in range(100):
            newcount += 1
      #section 3
      newcount2 = 0
      for i in range(x):
            newcount2 += 1
      return [count, newcount, newcount2]
```

   a. Answer the following questions about the big-O runtime of this function:
      i. What is the runtime of section 1?
         Answer: O(log(x))
      ii. What is the runtime of section 2?
         Answer: O(1)
      iii. What is the runtime of section 3?
         Answer: O(x)
      iv. What is the overall big-O runtime of this function?
         Answer: O(x) - O(log(x)) + O(1) + O(x) = O(x)
   b. What is the runtime of the function if section 2 and section 3 are performed every iteration of the while loop in section 1?
      Answer: O(xlog(x))

2. **Short Answer:** Answer the following questions about the Travelling Salesperson:
   a. What class of problems is this in?
      Answer: All problems (neither P nor NP)

   b. What is the runtime of this problem?
      Answer: O(n!)

   c. What is the tractability of solving the problem? What is the tractability of verifying the solution?
      Answer: Intractable, Intractable

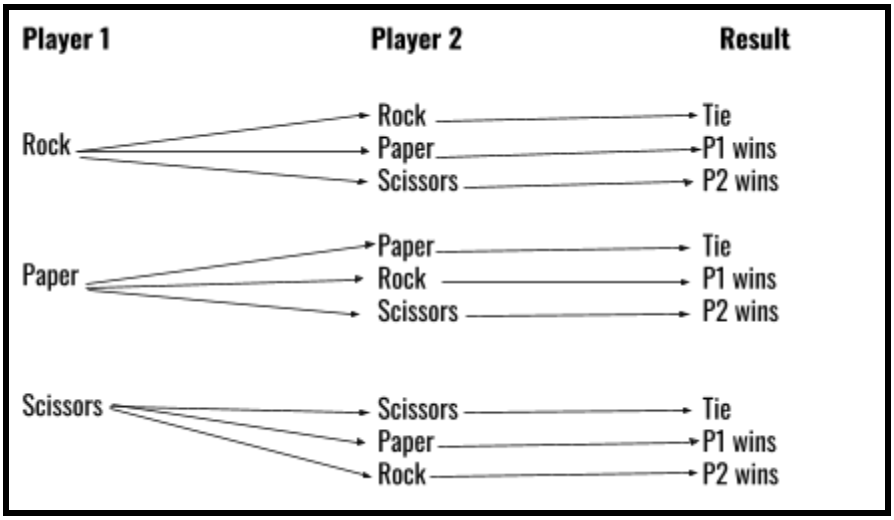
d. Briefly explain how (without using code) you would solve this problem? How is this reflected in the runtime/class that the problem is in?
Answer: Using the brute force method, we would have to try every combination of nodes. Think about a graph...if we have 4 nodes, we initially have 4 nodes to choose from, then 3 once we choose 1, then 2, then 1. This leads to a n! (4!=4*3*2*1) runtime. While 4 nodes aren't that many, if we have even a few more nodes, there are a lot more combinations to test. To find the best solution, we would still have to test all of the combinations. Since we can't find the solution nor verify in a reasonable amount of time (n!), this problem is neither in P or NP.

e. Now consider what happens if we modify the problem to say that we just want a solution that is under 1000 miles in total distance:
    i. How does the class of the problem change?
    Answer: It becomes NP
    ii. Does the time that it takes to solve this problem and/or verify a solution change? Why?
    Answer: While it may take a while to find a solution, we can easily test if a solution works. It either satisfies the problem statement or it doesn't, no need to test all the possible solutions.
    iii. Is this problem tractable or intractable?
    Answer: Tractable for verification, Intractable for solving

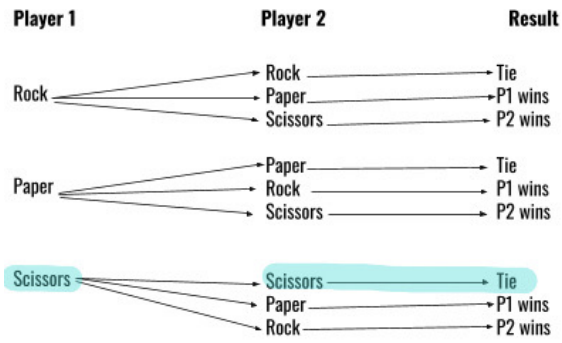
ML and AI

1. **Multiple Choice:** *True or False:* When training an algorithm, it is important to use all of the data you have at your disposal in order to make sure the algorithm can have as much practice as possible.
Answer: False - We never want to train our algorithm on all of our data. Separating your data into training data and testing data ensures that the algorithm can be tested for true accuracy.

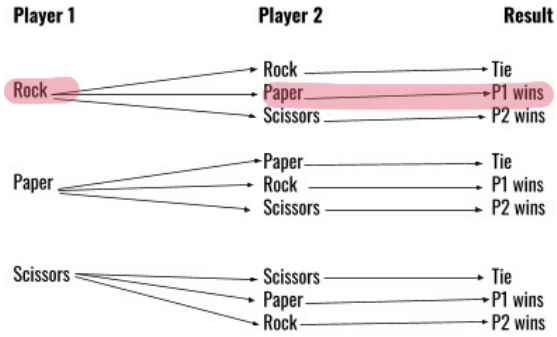
2. **MC:** Based on the game tree for rock, paper, scissors shown below, which of the following options demonstrates a game state score of 1 for Player 2?



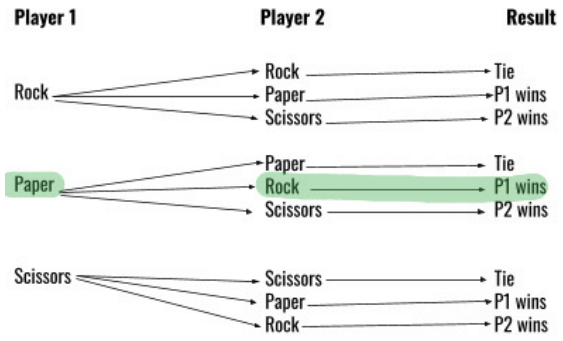
Option A:



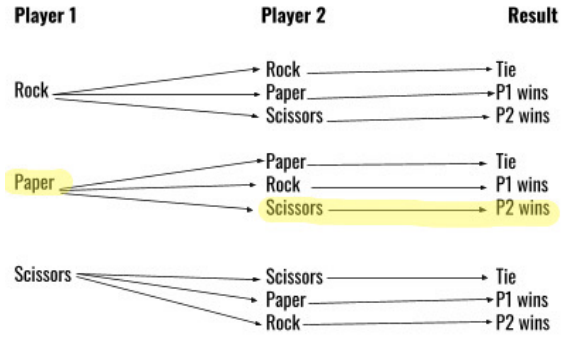
Option B:



Option C:



Option D:



Answer: D - This is the only option where Player 2 wins the game, thus Player 2's game state score in this situation would be 1 as they do not tie (0) and they do not lose (-1)

Graphs and Search

1. **Code Writing:** Write a function `findSmallConnections`, given a weighted graph like the one below, that will return all the pairs of nodes whose weights are less than 5, in a 2D list. Note that repeats do not matter.

```
g = {
"A" : [ ["B", 5], ["E", 2] ],
"B" : [ ["A", 5], ["C", 3] ],
"C" : [ ["B", 3], ["F", 9] ],
"D" : [ ["E", 1], ["F", 7] ],
"E" : [ ["A", 2], ["D", 1], ["F", 2] ],
"F" : [ ["C", 9], ["D", 7], ["E", 2] ]
}
```

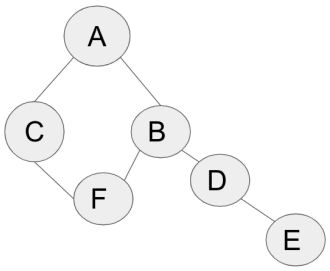
Example: with the graph given above, the intended result would be:

```
[["A", "E"], ["B", "C"], ["C", "B"], ["D", "E"], ["E", "A"],
["E", "D"], ["E", "F"], ["F", "E"]]
```

Code:

```
def findSmallConnections(g):
    result = []
    for node in g:
        for pair in g[node]:
            if pair[1] < 5:
                result.append([node, pair[0]])
    return result
```

2. **Short Answer:** Consider the following graph:



   a. What nodes (in order) would the DFS algorithm visit if you started at node A and searched for node E?
   Answer: A B D E
   b. What nodes (in order) would the BFS algorithm visit if you started at node A and searched for node E?
   Answer: A B C D F E

Hashing & Search/Sort

1. **Code Tracing:** Python has a built in hash function, but recall that we can also make our own! Consider these four functions, which all take in a string and hash it.

```
def h1(s):
    return 0

def h2(s):
    return ord(s[0])

def h3(s):
    x=0
    for i in range(len(s)):
        n=ord(s[i])
        x+=n
    return x

def h4(s):
    x=0
    for i in range(len(s)):
```

```
        n=ord(s[i])
        x=128*x+n
    return x
```

a. Which is the best to use? Why (what term do we want to avoid)? What is the runtime of good and bad hashing function situations?
Answer: h4
The first one results in everything getting stored in the first bucket, the second is fine but will lead to all strings starting with the same character being put in the same bucket, the third is better but can lead to collisions based on if the strings have the same letters, and the fourth one has positional dependence of the characters, leading to the most unique combinations. We want to avoid collisions, where things are stored in the same bucket. Hash tables have expected lookup time of O(1) when using a good hash function that spreads out elements among the buckets. In the worst case, a bad hash function can lead to lookup of O(n) when elements are hashed to the same bucket(s)

b. Use all the functions to hash ‘hi’, ‘bye’, ‘cat’, ‘cool’, and ‘act’ into a hash table with 10 buckets. (You can use this ASCII table if needed http://www.asciitable.com/)
Answer:
1st method:

‘hi’ ‘bye’ ‘cat’ ‘cool’ ‘act’									

2nd method:

				‘hi’			‘act’	‘bye’	‘cat’ ‘cool’

3rd method:

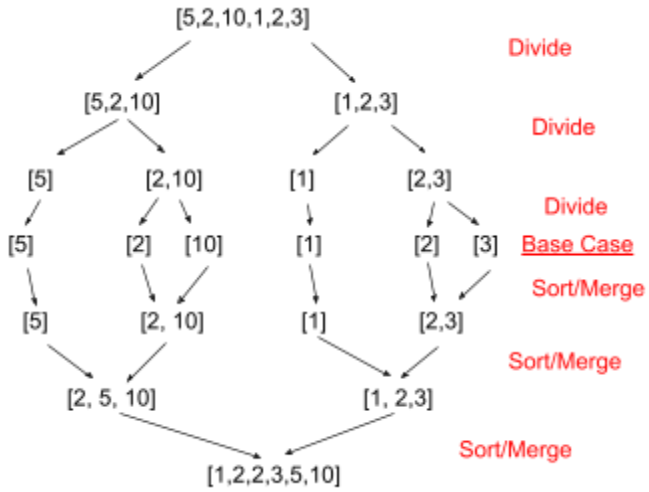
‘bye’		‘cat’ ‘act’							‘hi’ ‘cool’

4th method:

	‘bye’					‘act’	‘hi’	‘cool’ ‘cat’	

2. **Short Answer:** Consider the following questions about sorting algorithms
    a. Merge sort:

i. Draw out merge sort on the list [5,2,10,1,2,3]
Label the steps based on if its dividing, conquer/sort, or combine/merge
Label the step where the function is at its base case.
Answer:



ii. Explain merge sort's runtime
Answer: The n part of the runtime comes from the n copies at each division level, n copies at each merge, and n comparisons when comparing the elements. The depth of the copying depends on $\log_2 n$, since the list is getting cut in half every time and being split concurrently. O(3nlogn)->O(nlogn)

b. Sort the same list using selection sort.
i. Keep count of the number of comparisons and swaps.
Answer:

[5,2,10,1,2,3]
Comparisons: 5 Swaps: 1
[1,2,10,5,2,3]
Comparisons: 4 Swaps: 0
[1,2,10,5,2,3]
Comparisons: 3 Swaps: 1
[1,2,2,5,10,3]
Comparisons: 2 Swaps: 1
[1,2,2,3,10,5]
Comparisons: 1 Swaps: 1
[1,2,2,3,5,10]

ii. Explain selection sort's runtime
Answer: There are (n-1)+(n-2)+(n-3)+...+2+1 comparisons with up to one swap per pass, which breaks down to $O(n^2)$

iii. Is it more or less efficient than merge sort? Why?
Answer: Less efficient $O(n^2) > O(n\log n)$

Authentication & Encryption

1. **Short Answer:** Answer the following questions:
    a. Describe the difference between a man-in-the-middle attack and a DDOS attack
    Answer: In man-in-the-middle attacks, an adversary sets up a router in a network that pretends to be a normal router. That lets this evil router intercept packets that are being sent to different destinations. In DDOS Attacks adversaries send or receive a huge amount of data to/from a server within a short period of time (as a large number of packets). This overwhelms the server and makes it impossible for it to respond to authentic requests, so the site looks like it is down to a normal person.

    b. Describe the difference between symmetric and asymmetric encryption algorithms.
    Answer: Symmetric means that a single key needs to be known by both parties before messages can be exchanged. Others are asymmetric; each person has their own key.

    c. What are 2 of the encryption algorithms we discussed in class, and state whether they are symmetric or asymmetric
    Answer: Caesar Cipher (symmetric) and RSA (asymmetric)

    d. Why is breaking RSA nearly impossible?
    Answer: To break RSA, we would have to do factorization, which is in NP. Thus, it is very hard to break

2. **Multiple Choice:** Tunneling is a process used by which of the following:
    a. RSA Encryption
    b. HTTP Protocol
    c. VPN
    d. Caesar Cipher
    e. Both B and C
    Answer: C

Parallelism, Pipelining, MapReduce

1. **Short Answer:** Consider the following lists of tasks to prepare a hamburger at a fast-food chain and how long each step takes
    1) Cook the patty (10 min)
    2) Toast the bread (3 min)

3) Prepare the vegetables (3 min)
4) Put it together (2 min)

These steps are currently divided among 4 employees using **pipelining** where each worker performs one step.

State whether the following are true or false for the above scenario, and explain:

f. If a new employee is hired the total number of burgers produced will increase
Answer: False - In pipelining each worker has one task that only they are performing. Since there are only 4 tasks, a new employee will not have a task that they can perform and they will not increase the overall burger production

g. If the time taken to prepare the vegetables is reduced to 1 min, the number of burgers produced in an hour will increase
Answer: False - Pipelining's total time is dependent on the time of the longest task. Changing the length of a shorter task will not increase the burger production as it is still limited by how long it takes to cook a patty

h. If the time taken to cook the patty is reduced to 8 min, the number of burgers produced in an hour will increase
Answer: True - Pipelining's total time is dependent on the time of the longest task. By reducing the time required to cook a patty, the time of the longest task has been reduced and more burgers can be produced in an hour.

2. **Short Answer:** You are tasked with providing some analytical information for Spotify Rewind 2020. Your goal is to find the average number of country songs across all Spotify users favorite songs list. Design a MapReduce algorithm to solve this problem. Describe at a high level what values this specific mapper function would input and output, what values this reducer would input and output, and what actions the manager function would take.
Answer:
Mapper: Input - The favorites list of a single user OR a single user's profile
Output - The number of country songs in the user's favorites list

Reducer: Input - A list of numbers of country song
Output - The average number of country songs in the list

Manager: Split up the favorite lists of each user OR the profiles of each users and send one to each mapper; Collect the counts of country songs from each mapper and place them in a list; Send the list to the reducer; Obtain the result from the reducer and output the result

Data Analysis/Visualization

1. **Short Answer:** For each situation, indicate what type of graph would work best
    a. You want to compare the mean SAT scores for several different high schools in the district
    Answer: Bar Graph. 2D data - categorical x numerical (average)

    b. You want to analyze how students did on their last exam by examining the grading distribution
    Answer: Histogram. 1D data - numerical

    c. You want to analyze how temperature and location affect rainfall for several places around the world
    Answer: Colored Scatter Plot. 3D data - numerical x numerical x categorical

2. **Code Writing:** A professor would like to see how the students in his class are doing with their grades. He has a CSV file that has every student's name, grade, attendance, and recitation section in each column respectively.
    Here is the first 21 rows read in from his CSV file as a 2d list:

```
data = [['name', 'grade(out of 100)', 'attendance(out of 100)',
'recitation section'],
 ['Olivia', '83', '83', 'a'],
 ['Emma', '83', '83', 'k'],
 ['Ava', '81', '85', 'e'],
 ['Sophia', '80', '78', 'j'],
 ['Isabella', '93', '93', 'f'],
 ['Charlotte', '84', '85', 'd'],
 ['Amelia', '74', '76', 'c'],
 ['Mia', '84', '86', 'j'],
 ['Harper', '73', '73', 'c'],
 ['Evelyn', '88', '87', 'k'],
 ['Ezra', '91', '92', 'e'],
 ['Hudson', '87', '86', 'g'],
 ['Charles', '90', '87', 'k'],
 ['Caleb', '76', '74', 'b'],
 ['Isaiah', '83', '82', 'e'],
 ['Ryan', '86', '84', 'd'],
 ['Nathan', '83', '84', 'k'],
 ['Adrian', '79', '82', 'j'],
 ['Christian', '77', '81', 'f'],
 ['Maverick', '82', '81', 'c']]
```

    Write the function `makeDataDict(data)` that takes in the 2d list of data above and creates a dictionary that maps student names to a list where the first element is that students grade, and the second element is that students attendance

HINT: Remember that when you read in a CSV file, all of the fields are strings

Create a data dictionary from the 2d list of data and assign it to a variable "dataDict"

First, we want to see the overall grading distribution. Write a function graphGrades(d) that takes in the data dictionary dataDict from above and creates a histogram of all of the grades.

matplotlib functions to use:

```
(import matplotlib.pyplot as plt)
plt.hist(dataName)
plt.show() # Include this right after the above line
```

The professor would like to know if lecture attendance has any correlation with grades. Write a function `graphAttendence(d)` that takes in a dictionary dataDict as above and creates a scatter plot of attendance vs grades.
matplotlib functions to use:

```
(import matplotlib.pyplot as plt)
plt.scatter(xDataName, yDataName)
plt.show()
```

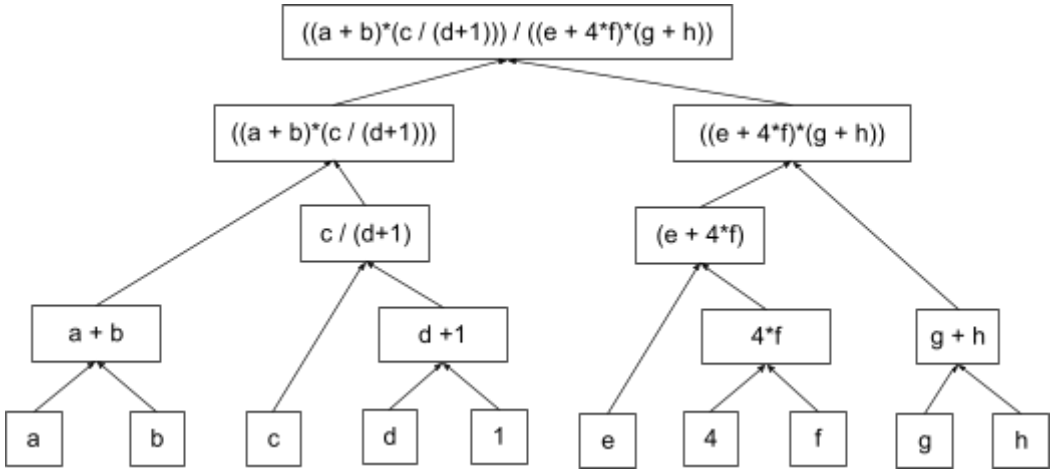
Code:

```
import matplotlib.pyplot as plt
#assume data is the list defined above

def makeDataDict(data):
      datadict = {}
      data = data[1:] #Remove header row
      for row in data:
            name = row[0]
            grade = int(row[1])
            attendance = int(row[2])
            dataDict[name] = [grade, attendance]
      return dataDict

dataDict = makeDataDict(data)

def graphAttendence(d):
      grades = []
      for key in d:
            grades.append(d[key][0])
      plt.hist(grades, color="orange")
      plt.show()

graphAttendence(dataDict)
```

```
def graphGrades(d):
      attendance = []
      grades = []
      for key in d:
            grades.append(d[key][0])
            attendance.append(d[key][1])
      plt.scatter(attendance, grades, color="red")
      plt.show()
graphGrades(dataDict)
```

Concurrency, Internet

1. **Short Answer:**
   a. Create the concurrency tree for the equation
   `((a + b)*(c / (d + 1))) / ((e + 4*f)*(g + h))`
   Answer:



   b. How many total steps are there?
   Answer: 9 total steps

   c. How many time steps are there?
   Answer: 4 time steps

2. **Multiple Choice:** Which of the following is NOT true?
   a. HTTP is a protocol used by browsers that describes how to request information from a website
   b. Packets are not guaranteed to reach their destination in the same order they were sent
   c. Buffering is used to show you part of a website while the rest of it loads
   d. The Cloud is not built to handle failures (i.e not fault-tolerant)

   Answer: D

a. Browsers receive webpages as text and turn that text into visual content using the HTML protocol
b. Packets can take many paths to reach the destination, so there is no guarantee that packets will arrive in the same order, or even at all
c. Some web pages need a lot of packets. For example, a video takes a lot of data to render. Packets may take a long time to reach the browser, which can cause lag. Therefore, the browser uses buffering to show you parts of the website while it waits for the rest of the packets to load.
d. The Cloud is designed to be exceptionally fault-tolerant, to avoid losing any data in case of failure.

Multiple Choice: Which of the following is true about IP addresses?

a. Computers will maintain the same IP address for the entirety of its lifetime
b. IP addresses are a core part of a computer, and are built into the hardware
c. ICANN now has a new system for IP Addresses that contains 16 bytes which can handle 10^38 addresses
d. A request needs to pass through only one server to reach a DNS server

Answer: C

a. Some IP Addresses are static. Many of these are the addresses of specific websites (like Google, or CMU). Other IP Addresses are dynamic. They get assigned to different computers at different times. This is used for computers that go online and offline regularly (like your computer).
b. IP Addresses aren't a core part of a computer; they aren't built into the hardware or software. But they aren't entirely random either.
c. This is correct
d. A request to a website may need to pass through several routers to get to a DNS server