Assignment 01- Artificial Intelligence Lab

Contents

1. Lists and Dictionaries 2
2. Numpy Matrix 3
3. String manipulations 6
4. OOP 7

Instructions 9

1. Lists and Dictionaries [5 + 5 + 15]

a) You own a car rental company. You maintain car bookings in an array consisting of start day and end day, e.g. [[1,10], [5, 10], [11,20]], which means that one person has booked a car from day 1 to day 10, second person has booked a car from day 5 to day 10 and a third person has booked a car from day 11 to day 20. Given such as an array, write python code to determine the minimum number of cars required to fulfil the booking.

Example 1

Input: bookings= [[1,10], [5, 10],[11,20]]

Output: 2

Example 1

Input: bookings = [[1,5], [3,8], [7, 10], [11, 15]]

Output: 2

b) You have an array of any type. There is method call **most_frequent_elements_in_lis(elem_list, left, right, frequency)**. This method will take four parameter left, right, frequency and array. Left and right are left and right index of array/subarray and frequency of element. Your task is to find those elements in array which occur frequency time or more than frequency time. Example lis = [2,2,4,2,4,5,6, l,1].

We call most_frequent_elements_in_lis(Arr,0,5,2) it will, return the following elements [2,4].

c) There are two main tasks to complete. (10+5)

i. Write a function with the name `get_student_marks`. This function will be given the following list as its only parameter:

```
[{'roll_no': 'p18-1001', 'marks': {
        'english': (1.4, 2.5, 15, 9.6, 33),
        'calculus': (2.4, 1.5, 12, 1.6, 21),
    }, 'attendance': 88.4
  },
  {'roll_no': 'p18-1002', 'marks': {
        'english': (2.4, 1.5, 12, 1.6, 21),
        'programming fundaments': (2.4, 1.5, 12, 1.6, 21),
    }, 'attendance': 79.4
  },
  {'roll_no': 'p18-1003', 'marks': {
        'calculus': (2.4, 1.5, 12, None, 21),
        'programming fundamentals': (2.4, 1.5, 12, 1.6, 21),
    }, 'attendance': 79.4 }]
```

This is not that complicated. So, pay attention: It's a list of student records – each record being a dictionary. This has three keys: roll_no and attendance are straight-forward but the value of the key 'marks' is itself a dictionary.

This dictionary has subject names as keys and tuples as values. Each of these tuples has the marks the student has obtained in a quiz/assignment. (If the student did not take that quiz, the record will have a `None` there.)

Your mission, should you choose to accept it, is to write `get_student_marks` in a way that it returns the cumulative marks of each student but in a specific structure: So, for the case above, it should return a dictionary as follows:

```
{
 'p18-1001': {'english': 61.5,
              'calculus': 38.5},
 'p18-1002': {'english': 38.5,
              'programming fundaments': 38.5},
 'p18-1003': {'calculus': 36.9,
              'programming fundamentals': 38.5}}
```

So, you now have a dictionary where the key is the student's roll number and the value is a dictionary of subject, total marks pairs.

Your function should be able to handle any number of evaluation marks in the tuple as well as any number of students.

ii. The second function you need to write is `get_grade`. It will be given a float as its first argument and a dictionary as the second argument. For instance, we can send it the marks 65 as its first argument and the dictionary sent to it will be, for example, the following:

```
{'A': 80, 'B': 70, 'C': 60, 'D': 50}
```

The semantics of this dictionary are that if the student gets more than or equal to 80 marks, the grade should be 'A'. With 70, the grade should be B and so on. Anything below a 50 is an F.

So, for the above two parameters, the grade should be C. Notice that the dictionary is only an example and the values of these grade can change (but the actual grade letters will always be A, B, C, D or F). You can assume that marks for A will always be greater than those for B and so on.

2. Numpy Matrix [10 + 5+ 15+5]

a) Create a two dimensional (7x7) numpy array and initialize it with random integer values in between 1- 10. Your task is to do the followings;

i. Write a method "second_Max_RowSum(mat)" that will calculate and return the second maximum sum of rows.

ii. Write a function that check whether matrix is upper triangle or lower triangle?
iii. Write a function that return the row that have minimum sum of the row.
iv. Write a method "**swap_Odd_Rows**" that will swap the values of odd rows.
v. Write a method that will calculate the meam of all rows and return the array of mean.
vi. Write a method "**sortColumns**" that will sort each column of 2-D array into ascending order.

b) You are given a matrix of characters. The matrix has *N* rows and *M* columns. Given a string *s*, you have to tell if it is possible to generate that string from given matrix. Rules for generating string from matrix are:

i. You have to pick first character of string from row *1*, second character from row *2* and so on. The (N+1)th character of string is to be picked from row *1*, that is, you can traverse the rows in a cyclic manner (row *1* comes after row *N*).
ii. If an occurrence of a character is picked from a row, you cannot pick the same occurrence again from that row.

You have to return **Yes** if given string can be generated from matrix using the given rules, else return **No**.

c) An hour Glass Problem:

Given a 6×6 2D Array, arr:

```
1 1 1 0 0 0
0 1 0 0 0 0
1 1 1 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
```

An hourglass in A is a subset of values with indices falling in this pattern in arr's graphical representation:

```
a b c
  d
e f g
```

There are 16 hourglasses in arr. An hourglass sum is the sum of an hourglass' values. Calculate the hourglass sum for every hourglass in arr, then print the maximum hourglass sum. The array will always be 6×6.

A complete Example of hourglasses

```
1 1 1 0 0 0
0 1 0 0 0 0
1 1 1 0 0 0
0 0 2 4 4 0
0 0 0 2 0 0
0 0 1 2 4 0
```

Hourglasses are:

arr contains the following hourglasses:

```
1 1 1   1 1 0   1 0 0   0 0 0
  1       0       0       0
1 1 1   1 1 0   1 0 0   0 0 0

0 1 0   1 0 0   0 0 0   0 0 0
  1       1       0       0
0 0 2   0 2 4   2 4 4   4 4 0

1 1 1   1 1 0   1 0 0   0 0 0
  0       2       4       4
0 0 0   0 0 2   0 2 0   2 0 0

0 0 2   0 2 4   2 4 4   4 4 0
  0       0       2       0
0 0 1   0 1 2   1 2 4   2 4 0
```

The hourglass with the maximum sum (19) is:

```
2 4 4
  2
1 2 4
```

d) Numpy manipulation

i. Replace all odd numbers in arr with -1 without changing `arr.`

```
Input: arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

Out:array([ 0, -1,  2, -1,  4, -1,  6, -1,  8, -1])
```

ii. Get all items between 5 and 10 from `array.` Input should be random array of numbers between 1-17.

3. String manipulations: [2 + 3 + 5 + 5]

a. Given two strings, split the string to character then check whether the given strings are Anagrams or not.

   An anagram is a word formed by rearranging the letters of another given word, for example, “car” and “arc”.

b. Write an efficient program to test if two given String is a rotation of each other or not, e.g. if the given String is "XYZ" and "ZXY" then your function should return true, but if the input is "XYZ" and "YXZ" then return false.

c. A string is said to be "SUPER STRING" if the number of times the character appeared in the string is equal to its ASCII value. Given the conditions that the ASCII value of ‘A’ is 26 and Z is ‘1’. Example: ZYYZ is not SUPER STRING (As Y appear 2 time and ASCII is 2 but Z’s ASCII is 1 but appeared 2 time). ZYY is SUPER STRING

d. Write a program that takes a string as input and calculate the possible number of sub-string that are palindrome. Recall that a palindrome is a non-empty string that reads the same backward as forward. Two sub-strings are considered to be different if they have different lengths or start at different positions in the original string. For example string “ABBA” has possible 4 sub-string palindromes { A, B, BB, ABBA}.

4. OOP [75]

Macau - card game

Macau, also spelled Makaó or Macao, is a Hungarian version of Crazy Eights, where players play a single card in sequence in a manner similar to Uno. Unlike Uno, however, Makaó is played with a standard deck of 52 cards. Makaó also involves bluffing so that the players do not necessarily have to play a card if they wish to save it for higher points later. Cheating is encouraged in the game, and it can make for a lively evening.

Gameplay

2 or more players (up to 10) are dealt 5 cards each; the deck is then cut and the cut card becomes the first card in the discard pile. Play starts to the dealer's right.

The next card played must be an of the same suit or same value as the card on the top of the discard pile. If a 7 of spades was on the top of the discard pile, the player can play a 9 of spades or a 7 of clubs. Alternatively, an Ace or Joker can be played. If the player cannot play a card, he must draw from the deck.

Cards can be played in runs, i.e. 5 of Spades, 6 of Spades, 7 of Spades, 7 of Hearts, 8 of Hearts, 8 of Clubs, 7 of Clubs.

When an action card is played, the player next in sequence must complete the action or add to it to pass in onto the next player in sequence. When down to a single card, a player must say "Macau!". If an opponent calls Macau before the player, the player must draw a card. The winner of the game is the first player to have no cards; in instances of more than 2 players, game play continues until only one player is left with cards.

Action Cards

- If a 2 is played, the next player in sequence must pick up 2 cards unless they have a 2, in which case they can add this to the original 2 and the next player in sequence must pick up 4 cards and so on.
- If a 3 is played, the next player in sequence must pick up 3 cards unless they have a 3, in which case they can add this to the original 3 and the next player in sequence must pick up 6 cards and so on.
- If a 4 is played, the next player in sequence must miss a go, unless they have a 4, in which case they can add this to the original 4 and the next player in sequence miss 2 goes.
- If a Jack is played, the player placing the Jack can call for a non-action card value, which they must hold - if they do not hold the value they are calling, they must call for 'any non-action card', the player in sequence must either play the card value called or place another Jack down and call a different value.
- If a King of Spades is played, the previous player in sequence must pick up 5 cards, unless they have a King of spades or hearts or queen of hearts, in which case they can add this to the original King and the next player in sequence must pick up 10 cards.
- If a King of Hearts is played, the next player in sequence must pick up 5 cards, unless they have a King of spades or hearts or queen of hearts, in which case they can add this to the original King and the next player in sequence must pick up 10 cards.
- If a Joker is played, the sequence is reversed, i.e. clockwise play becomes anti-clockwise.
- If an Ace is played, the player playing the Ace must call a suit different what it is in play. They must hold a card of the suit they are calling, if not, they must call 'free suit' and the next player in sequence can play any suit other than that already in play.

Multiples action cards can be played, i.e. Player 1 plays three 2s and the next player in sequence must

pick up 6 cards unless they have another 2. This is the same for 3's, 4's and Kings.
When a player get down to 1 card they must say Macau or if another player catches them not saying Macau they must draw 5 cards.

To Do

- ➔ Identify the classes and their relationship.
- ➔ Implement the complete game keeping in mind each and every aspect of the game. Display should be readable. And use **input( )** where player want to pick the card.
- ➔ If it is to draw from the deck, then it must be random and update the hands and decks after the card is drawn vice versa.
- ➔ Before distributing the cards shuffle the deck. Most appropriate data structure to store the cards is dictionaries.
- ➔ Cut is also made by the random number.
- ➔ Implement the game keeping in mind the that you are only allowed to use lists, dictionaries and numpy arrays/vectors, strings.
- ➔ Multiple players can play the game.
- ➔ You should have function that shuffle the cards.

National University of Computer and Emerging Sciences

School of Computing **Spring 2022** **Islamabad Campus**

Note: Please carefully read all the instruction.

Instructions: -

1. Read carefully the assignment and understanding the statement is part of the assessment.
2. The assignment is of total of 150 marks.
3. You are allowed to use **numpy** library as well **list**, **dictionaries** and **sets** built in **functions** in assignment.
4. You have to submit the **notebook** file with ***rollNo_labSection.ipynb*** and also submit the **.py** file with named as ***rollNo_labSection.py*** *(*in both cases you have to submit both file).
5. You should number the solution with proper heading in the notebook.
6. Use proper variable names where required
7. Rubrics to evaluate your code is pretty straight forward.
    a. Each and every question has specific function and at least tested on **3** examples.
    b. Code should be optimized means if a solution existed in O(n) or O(nlogn) and you solved in O(n^2) then 30% marks from that specific question will be deducted.
    c. In the **4th** Question create constructors, setters and getters as well as proper relationship between classes also required.
    d. Without heading of question or without proper commenting also penalty of 20% marks in that question.
    e. Bad naming convention also gets you in trouble of 10% penalty of whole assignment marks.
8. If a built in function is available of numpy array, lists, tuple, sets and dictionaries and you built a scratch function then there is no learning frim the assignment. The purpose is minimum line of codes and maximum output.
9. **Plagiarism cases will be dealt strictly. If found plagiarized, both the involved parties will be awarded zero marks in this assignment. Copying from the internet is the easiest way to get caught!**

10. Deadline is: Monday 21st February 2022 at 8:30 AM. Correct and timely submission of assignment is responsibility of every student; hence no relaxation will be given toss anyone.