CS102 – Algorithms and Programming II
Lab Programming Assignment 5
Spring 2022

ATTENTION:

- Feel free to ask questions on Moodle on the Lab Assignment Forum.
- Compress all of the Java program source files (.java) files into a single zip file.
- The name of the zip file should follow the below convention:
  CS102_SecX_Asgn5_YourSurname_YourName.zip
- Replace the variables "YourSurname" and "YourName" with your actual surname and name and X with your Section id (1, 2 or 3).
- Upload the above zip file to Moodle by April 12, 23:59 with at least Part 1 completed. Otherwise, significant points will be reduced. You will get a chance to update and improve your solution (Part 1 and Part 2) by consulting the TA during the lab. You will resubmit your code (Part 1 and Part 2 together) once you demo your work to the TA.

The work must be done individually. Codesharing and copying code from any source are strictly forbidden. We are using sophisticated tools to check the code similarities. We can even compare your code against online sources. The Honor Code specifies what you can and cannot do. Breaking the rules will result in a disciplinary action.

This lab is about Recursion. You can use the Test Cases at the end of the document to check your solution.

Part 1 +

Question 1

Assume that Sarah purchases **N** apples on day 0. She eats either **2** or **3** apples each day starting from day 1. Write a **recursive** program that checks if she can have **k** apples at day **a**.

Example:

N = 12, **k** = 4, **a** = 3

Day 0 -> has 9 apples

Day 1 -> eat 3 apples (9 left)

Day 2 -> eat 3 apples (6 left)

Day 3 -> eat 2 apples (4 left) **=>** 4 apples left, so, done.

Question 2 + knapsack problem 01

In this question, you are going to help Mike Wazowski. He wants to download video games but his computer has limited storage of **N** GB. Each game **G_i** takes up **A_i** GB of space and has **B_i** Metacritic score. Metacritic score is between 0 and 100. A high score signals high-quality games. Mike's aim is to maximize the Metacritic score of the games on his computer. Write a **recursive** function that takes an array of games as input and returns the scores of the games he should download to achieve the maximum score. Note that he can either download or not download, partially downloading a game is not possible. In addition, he can download each game at most once.

Example 1:
G = [[‘Game A’, 100GB, 50], [‘Game B’, 50GB, 10], [‘Game C’, 60GB, 45]]
N = 110GB
In the above case, since the storage is 110GB, to maximize the Metacritic scores, he should download ‘Game B’ and ‘Game C’ whose scores are 10 and 45. The maximum achievable total score will be 55.

Example 2:
G = [[‘Game A’, 100GB, 50], [‘Game B’, 50GB, 10], [‘Game C’, 60GB, 45]]
N = 109GB
This time the storage is more limited, so, he should only download Game A with score 50 to achieve a maximum score of 50.

Part 2

+edit distance problem

Question 3
Carl Fredricksen is interested to solve the following problem while he is in his flying house. Assume there are two words; **A** and **B**. He wants to know how to convert word **A** to word **B** by only using the following operations a minimum number of times:

1. Add a new character to **A**
2. Remove character from **A**
3. Substitute character in **A**

Write a **recursive** program that, given **A** and **B,** calculates the smallest number of operations Carl needs to do to convert **A** to **B**.

Example:
A = exclamation, **B** = excavation
exclamation -> excamation (remove l)
excamation -> excavation (replace m with v)
Minimum number of operations = **2**

Question 4

not done

Renowned scientific research company Aperture Science is trying to develop a portal technology. Help them by extending the recursive definition of Maze traversal code that we provided (MazeSearch.java and Maze.java by Lewis/Loftus) such that the agent can jump between portals. Each portal has two sides. If the agent steps into one side of the portal it will be teleported to the other side. That way, even if there is no road between two locations, it can reach its destination by jumping between portals. Note that there can be multiple portals. Also, the agent always starts from the top-left corner and its destination is bottom-right corner. At the end of the day, you should create a **recursive** program that traverses the Maze and tries to find if reaching the destination is possible by jumping through portals and walking.

Example:
1, 0, 0, 0, 0, 0
1, 1, 1, 0, 0, 0
0, 0, 0, 0, 1, 1
Assume that above matrix is a top-down view of the maze. The agent starts at top-left location (row=0, column=0) and its destination is bottom-right location (row=2, column=5). 1 signifies walkable areas and 0 means walls. So, in this case, it cannot reach its destination since there is no path of 1s between them.
Let's add portals,
1, 0, 0, 0, 0, 0
1, 1, 1, 0, 0, 0
0, 0, 0, 0, 1, 1
Now, blue locations (row=1, column=2 and row=2, column=4) are connected through portal. Therefore, it can jump between them to reach the destination.

Test Cases

You can use the below test cases to check the correctness of your solutions. We may give additional test cases during the lab session.

Question 1
N = 29, **k** = 5, **a** = 10 => **true**
N = 75, **k** = 25, **a** = 7 => **false**
N = 74, **k** = 30, **a** = 16 => **true**
N = 75, **k** = 25, **a** = 0 => **false**
N = 75, **k** = 75, **a** = 0 => **true**

Question 2
Note: The order of returned scores might be different in your case. That's okay!

G = [[45GB, 50], [10GB, 85], [15GB, 45], [20GB, 100], [25GB, 6], [100GB, 100]]
N = 150GB
=> [85, 45, 100, 100]

G = [[45GB, 50], [10GB, 85], [15GB, 45], [20GB, 100], [25GB, 6], [100GB, 100]]
N = 0GB
=> []

G = [[45GB, 50], [10GB, 85], [15GB, 45], [20GB, 100], [25GB, 6], [100GB, 100]]
N = 1000GB
=> [50, 85, 45, 100, 6, 100]

G = [[60GB, 75], [50GB, 85], [200GB, 65], [15GB, 100], [40GB, 55]]
N = 50GB
=> [100]

G = [[60GB, 75], [50GB, 85], [200GB, 65], [15GB, 100], [40GB, 55]]
N = 115GB
=> [85, 100, 55]

Question 3
A = "sunday", B = "saturday" => **3**
A = "exclamation", B = "excavation" => **2**
A = "inquire"; B = "ensure" => **3**
A = "car"; B = "race" => **3**
A = "man"; B = "men" => **1**
A = "plane"; B = "plane" => **0**

Question 4
Color code:
red, green, brown = different portals (i.e., two green locations mean they are connected through a portal)

1, 0, 0, 0, 0, 0
1, 1, 1, 0, 0, 0
0, 0, 0, 0, 1, 1
=> Can reach end

1, 1, 0, 0, 0, 0
1, 1, 1, 1, 1, 0
1, 1, 1, 1, 1, 0
0, 0, 0, 1, 1, 1
=> Can reach end

1, 1, 1, 1, 1, 1
0, 1, 1, 1, 1, 1
0, 1, 0, 0, 0, 0
0, 0, 0, 1, 0, 1
=> Cannot reach end

1, 1, 0, 1, 1, 1
0, 0, 0, 0, 0, 1
0, 0, 0, 1, 0, 0
0, 0, 0, 0, 0, 0
1, 1, 0, 0, 1, 1
=> Can reach end

IMPORTANT NOTES:

1. Please comment your code according to the documentation and commenting conventions used in the textbook.