CME 2204 Assignment-2
Dynamic Programming and Greedy Approach

11.06.2021 23:55 (Sharp deadline – No extension)

Rules:

- The submissions will be checked for the code similarity. Plagiarism will be graded as zero.
- Other rules are explained at the end of this document.

Scenario of the Assignment:

A large family of lions is preparing for a hunt. Each lion in the family has hunting ability and this ability is scored between 0 and 100. In addition, this family has a criterion for those who will be chosen to go hunting; "No lion can go hunting with his/her immediate parent". The reason for this is to ensure that there will be a lion (s) who will stay at home to protect their house. Lions will be selected so that the *'total hunting ability'* should be maximum.

Two criteria:

1- No lion can go hunting with his/her immediate parent
2- Provide maximum '*'total hunting ability'* for the selected lion group

You are expected to design a ***dynamic programming*** approach that selects lions to **maximize** *'total hunting ability'* with the given criteria. In order to evaluate the dynamic programming method, you are also expected to develop a ***greedy approach*** for this scenario.

The family hierarchy of lions is given in the "lions_hierarchy.txt". This file contains three columns (tab delimited) which are "first-node-name", "second-node-name" and "relationship" in accordance with **"Left-Child, Right-Sibling Representation"**. Hunting abilities of each lion is given in the "hunting_abilities.txt". This file contains two columns (tab delimited) which are "node-name" and "hunting ability".

Left-Child, Right-Sibling Representation: Each node holds pointers of parent, left-child, right-sibling (See Figure-1).

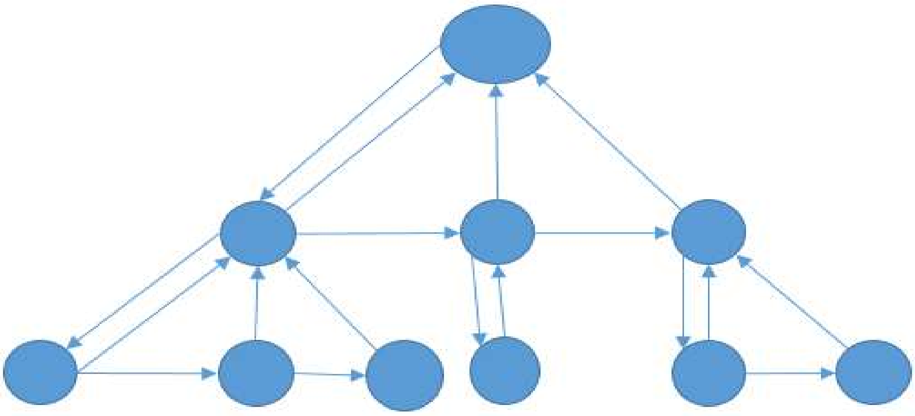

Figure-1: The left-child, right-sibling representation of a tree. Each node x has attributes x:p (top), x:left-child (lower left), and x:right-sibling (lower right).

lions_hierarchy.txt:

Table-1: Example for interpretation of the "lions_hierarchy.txt" file.

first-node-name	second-node-name	relationship
X	Y	Left-Child
Y	Z	Right-Sibling

In Table 1, the first row represents that Y is the left child of X, which means X is the parent of Y. Second row expresses Z is the right sibling of Y. Hence, Y's parent is also Z's parent which is X.

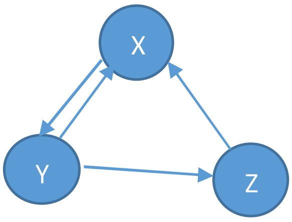

Figure-2: Graph representation of example given in Table-1.

Example scenarios for lions who are eligible (green colored) for going hunting:

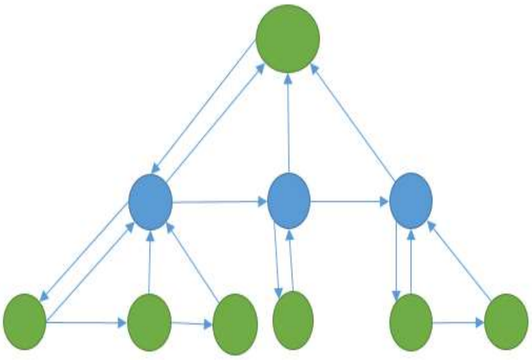

Figure-3: Possible Scenario-1

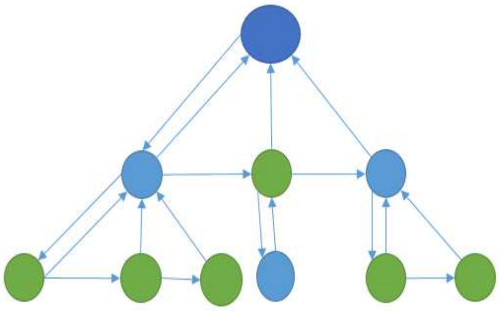

Figure-4: Possible Scenario-2

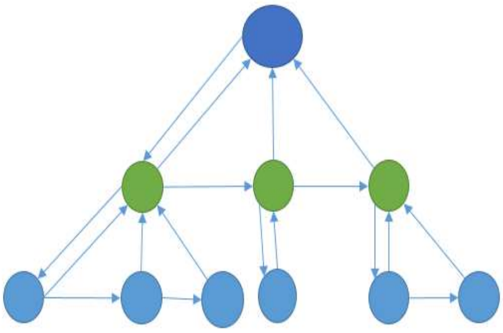

Figure-5: Possible Scenario-3

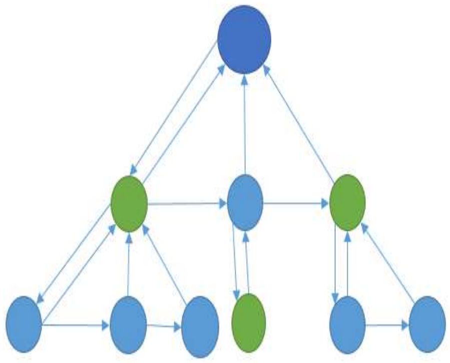

Figure-6: Possible Scenario-4

Your dynamic programming and greedy approach should identify the optimal lion team, who should maximize the total hunting ability. You need to return two outputs for both approaches:

1. Total hunting ability value

2. The lions who should be selected to guarantee the maximum hunting ability value

You are expected to write a ***Scientific Report*** that explains completed and missing parts of your code. Your report should include comparison of run time complexity and space complexity for both dynamic programming approach and greedy approach that you developed.

How to upload the documents (report and java file)?

You must create only ***one*** java file named as "Name_Surname_ StudentID.java" that includes the main, and some functions for dynamic programming approach and greedy approach as seen in Table-2. Your report must be named as "Name_Surname_ StudentID.pdf". Then, put these two files in a folder and archive it as .rar or .zip. And upload the archived file such as "Name_Surname_ StudentID.rar" or "Name_Surname_ StudentID.zip" to the course page.

Table-2: “What must be in your code?”

Seymur_Grace_2020510999.java

```
public class Seymur_Grace_2020510999 {

    public int DP(...) {

    ...

    }

    public int Greedy(...) {

    ...

    }

    //You can add different functions here...

    public static void main(String[] args)  {

        //read files

        ..........

        //Your code

        System.out.println(“DP Results:” + ( Total Hunting Ability )   );

        System.out.println(“DP Results- Selected Lions:”);

        System.out.println(“Greedy Results:” + (Total Hunting Ability)   );

        System.out.println(“Greedy Results- Selected Lions:” );


    }
}
```