CS 602
Algorithm Design and Implementation

Assignment 2

[Question 1] Mirror a Binary Tree into a Binary Search Tree

In this question, you are asked to convert a binary tree to the binary search tree with the mirrored structure. A mirrored binary tree of a binary tree is defined as follow: (1) the mirrored binary tree of empty trees are empty trees; (2) the left subtree of a node in the mirrored binary tree is the mirrored binary tree of its right subtree in the original tree; and its right subtree in the mirrored binary tree is the mirrored binary tree of its left subtree in the original tree. The mirrored binary search tree is then defined as the mirrored binary tree and it is a binary search tree.

Please implement the conversion from a binary tree to the mirrored binary search tree. Test inputs begin with the number of lines below. Each line describes a binary tree by a list of tree nodes with values at itself, its left child and its right child, separated by '`:`'. Letter `x` represents a null node, so a tree node with two `x` is a leaf node. The sequence of tree nodes is arbitrary. For example, '`0:x:x -1:1:-2 -2:0:x 1:x:x`' represents a 4-nodes binary tree rooted at -1, the left child is 1 and the right child is -2, where node 1 is a leaf node. For each binary tree, output the mirrored binary search tree in pre-order traversal separated by space, '`0 -2 -1 1`' for the above example.

Sample code is provided in the file **A2Q1.py**. You need to modify the function `mirror_BST`. The maximum of tree height is 100 and the maximum number of nodes in a binary tree is 1,000,000. Values in the tree nodes are integers between -2^{31} and $2^{31}-1$, and they are distinct.

Please state and justify the time complexity of your algorithm in comments.

Sample Input

Please refer to the file A2Q1.in .

Sample Output

Please refer to the file A2Q1.out .

[Question 2] Project Selection

As the chief project manager of the team, one task of yours is to select a set of projects for the team. Suppose each project i is associated to a pair of values c_i and r_i, where c_i is the cost to work on project i and r_i is the revenue of project i after completion. Assume your team start with a capital value c, and you can only select one project with its cost not exceeding your team's capital. Upon completion of the project i, your team's capital will increase by the profit of that project, which is $r_i - c_i$. You may also assume your team only work on one project at a time, and projects are all valid with $r_i \geq c_i \geq 1$.

Given the set of projects, an initial capital value c, and k the number of projects to be selected. You are to work out a selection plan such that the final capital is maximized.

Implement an algorithm with appropriate data structure (You may use external libraries without implementing this data structure: to test if you can use library xyz, please do a mock submission on the online judge by adding the line "import xyz" to the code skeleton. "Wrong answer" means the library is available on the online judge, and can be used for your implementation) for the function `project_selection`. Please also state and justify the time complexity of your algorithm as comments in your code.

Test inputs begin with the number of projects n ($n \leq 100,000$) and the number of scenarios. The second line contains a list of integer pairs of cost and revenue separated by ':'. These pairs are all distinct, it is possible for two projects to have the same cost or the same revenue, but not both. From the third line onward, each line describes one scenario with initial capital c and the number of projects to be selected k, where $k \leq n$. Both the capital and the costs of projects are integers and ranging between 1 and 1,000,000.

Sample Input

```
10 3
330:510 9:12 733173:864046 664:907 646346:764956 16006:21763 9804:11465 14655:23652
<continued from the previous line> 37023:44290 18:22
9931 5
29718 7
33 5
```

Sample Output

```
12022
53827
impossible
```

Running python skeleton with sample input:

1. Open "Anaconda Prompt"
2. Go to the directory where you put the file **A2Q1.py** and **A2Q1.in**, using command **`cd`**
3. Run command **python A2Q1.py < A2Q1.in**
4. You may want to create a test input called **my_own_test.in** to design a test case for your own program, the command would then be **python A2Q1.py < my_own_test.in**
5. Same applies to Question 2, so you may run **python A2Q2.py < A2Q2.in**