Computer Science 220S1C (2019)

Assignment 4 (traversal and optimisation)

Due date June 7, 2019, 10pm
100 Marks in total

This assignment requires you to submit programs in Python that you have written yourself to the automarker, `http://www.cs.auckland.ac.nz/automated-marker`. Your implementation must be from first principles and cannot use an existing library methods that might solve the problem (eg graph search algorithms etc). You can use libraries for standard implementations of data structures such as queues, stacks and priority queues.

The automarker uses the PyPy compiler on a Linux box. Read the automarker help and FAQ for more details.

Please submit only Python source code.

1. **Arithmetic trees** *50 marks*

   You are given an input file with multiple pairs of input lines. The first line of each pair is a tree given as a predecessor array. The second line is the value at the corresponding node. Values at leaf nodes (nodes with no children) are integers. At non-leaf nodes, the two possible values are $+$ or $*$.

   The tree represents an arithmetic expression where the value at a non-leaf node u is the sum of values at the children of u in the case of $+$, or the product of values at the children of u in the case of $*$.

   You need to calculate the value at each node and output the calculated value at the root. The tree is not constrained to be binary.

   Input format: Input consists of m pairs of lines of comma separated values, so $2m$ lines in total. The first line is each pair is a comma separated list of integers representing a tree in predecessor array format where -1 represents null.

   The second line in each pair is a comma separated list of integers and the symbols $+$ and $*$. The ith item on the list is the value or operator at the ith node in the tree.

   For example:

   ```
   -1,0,0,0,1,1
   +,*,2,3,0,7
   2,0,-1,0
   +,3,*,3
   ```

   Output format: For each pair of input lines, output a line containing the value calculated at the root of the tree.

   For the example input above, output would be:

   ```
   5
   6
   ```

2. **Optimisation** *50 Marks*

Suppose you are flying a drone that can fly exactly 100 km before needing a battery recharge. Recharging is only possible at towns for which you are given locations. Your aim is to find the length of the shortest possible path to get the drone from an origin town to a destination town given the range restriction.

Assume that the landscape is a $n \times n$ two dimensional square (units are km) with coordinates for towns described by (x, y) where $0 <= x, y <= n$. Use Euclidean distance to calculate the distance between towns.

Input format: You are given multiple lines of input. Each line has $2k + 1$ comma separated numbers where $k \geq 2$. The first number in each line is the size of the landscape, n.

The following $2k$ numbers locations of k towns so that that the ith town is located at $(2i, 2i + 1)$.

In each line the first town listed is the origin, the final town listed in the destination.

For example:

```
100,0,0,0,100,100,100
1000,20.892,986,602,138.97,206.2,10.44
200,25,25,10,1,50,25,140,30
```

Output format: For each line of input, output a single number which is the length of the shortest path from the origin town to the desitination town. Use `str.format` to give this value to 2 decimal places. Precisely, format `x` using `'{:.2f}'.format(x)`. Do not use any other rounding throughout your algorithm. If the destination town is unreachable from the origin, output -1.

For the example input above, output would be:

```
200.00
-1
115.14
```

Marking

The maximum number of submissions for each problem is fixed at 10.

Each problem has five test cases associated with it worth one fifth of the marks for that problem. Some of the test cases will be large to test for speed. You get full marks if you pass all test cases.