Advanced Programming

Samir Datta*

April 16, 2018

Instructions

---

1. Modify the attached file assignment.5.py as *yourusername.5.py*. For example, I would have used aalok.5.py.
2. The file that you submit should be a modification of the attached assignment.5.py file, i.e, it should contain the functions, classes and methods that have already been defined in this file. You may want to define more, but the given ones must be there.
3. The functions, classes and methods provided will be treated as explained in the problem statements and the comments. You are free to change the names for the arguments, but the code will be tested by running the methods as specified in the comments and the code.
4. The *return* type of each function/method should be as specified in the assignment.5.py file.

Problem 1

Implement `largestSumSubsequence` which finds the largest possible sum of a contiguous subarray of a given array.

For example, if the list was $[-2, -3, 4, -1, -2, 1, 5, -3]$, then the subarray $[4, -1, -2, 1, 5]$ is a subarray whose sum is maximum. So on this input, your program should return 7.

Your program may be checked on a list which is at most $1,000,000$ entries. Your program should terminate in atmost 15 seconds.

*Teaching Assistants - Kishlaya Jaiswal, Agnishom Chattopadhyay

Problem 2

A machine is available from t = `start` to t = `end`, inclusive. Several clients have requested jobs during several intervals to be scheduled. Each interval is represented as a tuple `(si, ei)`, indicating that the job shall be run on the machine from `si` to `ei` inclusive.

Implement `freeTime` as a function of `start`, `end` and `intervals` which determines the number of time units for which the machine is available but not engaged in any job.

For example, say `start = 0`, `end = 0`, and `intervals =` $[(1,2),(2,3),(5,7),(6,7),(10,10)]$. Then the machine is it free only during the times 0, 4, 8 and 9. So, you should return 4.

The difference between `start` and `end` may be at most 10^15. There might be at most 10^6 intervals, and they might not necessarily be in sorted order. Your program should terminate in at most 15 seconds.

Problem 3

In this problem, you are required to implement the `MedianKeeper` class, which supports two methods `update` and `query`.

An instance of the `MedianKeeper` is supposed to keep track of a collection of integers and be able to tell what the median of them is, when `query`-ied. This collection initially starts out empty, and can be added to when `update` is invoked with an integer that is to be added.

If the set currently contains an even number of elements, say $0..(2n-1)$, then the median is defined as the 0.5 times the nth and $n-1$th elements, if the elements were in sorted order. Otherwise, if there are an odd number of elements, say $0..2n$, the median is the nth element, if the elements were in sorted order.

Your program maybe tested against upto 1,000,000 `query`-ies and `update`s combined. The total time taken to process all the queries and the updates should not be any more than 15 seconds.

Hint: A good programmer is aware of the standard library. Have you heard of the heapq module?

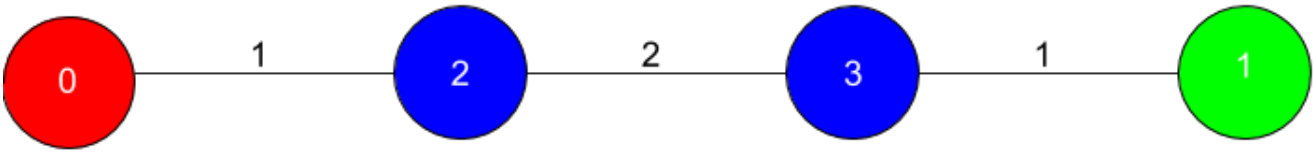
Problem 4

Deeparaj is bored and wants to drive from Chennai (vertex 0) back to Bangalore (vertex 1). However, his car can only hold L liters of fuel at a time, which means that he can only travel at most L kilometers at a stretch without hitting a gas station.

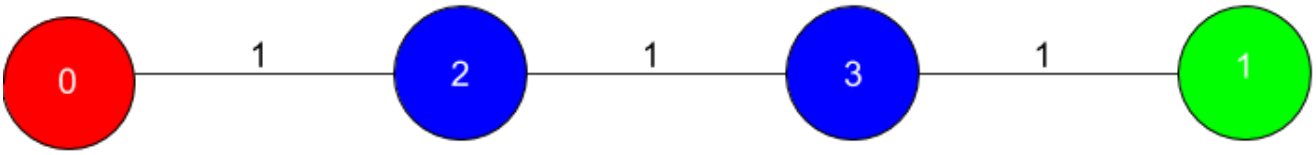
You shall be given the network of roads connecting the gas stations (and the destinations, and the sources) along with the lengths of the roads in the form of a (undirected) graph. You'll have to implement `canTravel` that will return `True` if Deeparaj can travel from Chennai to Bangalore, and return `False` otherwise.

The graph will be encoded in the form of an adjacency list The keys correspond to vertices The values are lists of tuples If `(l, v)` is present in the adjacency list of `u`, there is an edge of length `l` from `u` to `v` Here are two examples:

{`0: [(1, 2)], 1: [(1, 3)], 2: [(1, 0), (2, 3)], 3:[(2, 2), (1, 1)]`} stands for the following graph:



{`0: [(1, 2)], 1: [(1, 3)], 2: [(1, 0), (1, 3)], 3:[(1, 2), (1, 1)]`} stands fr the following graph:



Your program may be tested on a graph containing up to $10,000$ vertices and should terminate in at most 15 seconds.

Deforestation: When adding more branches gives you less trees