© 2019 Sudarshan S. Chawathe

Name: _____

This assignment builds on the previous one (*JJ's Jolly Jumping Journey*, or *J5*). The **goal** is to get more experience in working with algorithms for new problems. In particular, we will practice making connections between new problems and previously studied ones, as well as devising, implementing, and evaluating solutions.

The previous problem (J5) [verbatim from HW01, for reference] In one phase of the side-scroller game *JJ's Jolly Jumping Journey*, called *J5* for short, the objective is to guide the protagonist, *JJ*, to an exit door that is n meters away from JJ's initial position. JJ can move only by using a collection of k pogo sticks: $p_1, p_2, \ldots, p_k$. Each pogo stick p_i is a precision device that will move JJ exactly d_i meters toward the exit (unless it would overshoot, in which case JJ goes splat against a wall—to be avoided). The pogo sticks all work in just one (forward) direction. Each pogo stick may be used any number of times. (JJ carries them in a backpack when not being used.) We would like to enumerate the number of different ways JJ can get to the exit door, as a function of n and the pogo-stick distances $D = (d_1, d_2, \ldots, d_k)$. For simplicity, we assume that n and d_i are all integers and that the d_i are all distinct.

A new twist (T1) Each pogo stick p_i has an associated *cost* c_i, which is a positive integer. Using p_i for one jump (covering distance d_i) incurs a cost of c_i. As before, we would like to enumerate the number of different ways JJ can get to the exit door, as a function of n and the pogo-stick distances $D = (d_1, d_2, \ldots, d_k)$. However, we would also like to keep track of the cost of each option, given the additional data on costs: $C = (c_1, c_2, \ldots, c_k)$. We would also like to determine if a minimum-cost journey can be computed efficiently. A journey is way for JJ to travel from the initial position to the exit door. The cost of a journey is the sum of the costs of the pogo-stick jumps it uses. (Each jump using a pogo stick p_i contributes c_i to the cost.)

Twisting further (T2) The pogo-sticks have gone coin-op. Before JJ can use a pogo-stick p_i to cover its jump distance d_i, coins totaling c_i must be deposited into the pogo stick. Luckily, at some points on JJ's path toward the exit door, there are piles of coins. If a pogo-stick jump lands exactly on such a pile's location, JJ gets to pick up all the coins in that pile. There may also be a pile of coins in JJ's initial position, which are picked up automatically. As in the first twist, we would like to enumerate the journeys and associated costs, and to compute a minimum-cost journey efficiently. The difference is that costs may now be negative (net profit) as a result of coins picked up along the way. The locations and values of the piles of coins are specified as a list of pairs $G = ((l_1, v_1), (l_2, v_2), \ldots, (l_m, v_m))$. JJ always has enough cash on hand to use any pogo stick.

Questions

1. (1 pt.) Write your name in the space provided above.

2. (9 pts.) Describe, in English as precisely as possible, how your algorithm for the original problem (J5) from the previous assignment must be changed to include the cost of each journey that it enumerates, thus solving the enumeration part of J5+T1.

3. (10 pts.) Devise an efficient algorithm for computing a *minimum-cost journey* for J5+T1. There may be multiple minimum-cost journeys; in such a case, the algorithm may return any of them. Describe your algorithm in English as precisely as possible.

4. (10 pts.) Explain why your algorithm of Question 3 is correct.

5. (10 pts.) Provide pseudocode, using the textbook's style as a guide, for your algorithm of Question 3. Include explanatory comments and outline a proof of its correctness.

6. (10 pts.) State and justify the running time of your algorithm of Question 5 as a function of the number n and sequences of numbers D and C.

7. (40 pts.) Repeat Questions 3 through 6 for the problem including both twists (J5+T1+T2).

[additional space for answering the earlier question]

[additional space for answering the earlier question]

[additional space for answering the earlier question]

8. (150 pts.) Implement your algorithms. Test and document your work carefully and submit your packaged source code and supporting documentation.

9. (30 pts.) Conduct a brief experimental study of your implementation, measuring the running time for a suitable collection of inputs. Include your test code in your electronic submission, with suitable documentation.

10. (30 pts.) Summarize your experimental results by making effective use of charts and tables. Comment on how well the experimental results match the predictions based on your answer to Question 6. Highlight any significant differences and explain them the best you can. Include these results, comments, and explanations as a single PDF file in your submission.

IO format Your program should read from *standard input* and write to *standard output*. The first token (whitespace-delimited) of the input is either `E`, indicating that the desired output is an *enumeration* of all journeys, with costs, or `M`, indicating that the desired output is a *minimum-cost journey* (and its cost) only. The next input token is the distance n. The remainder of the input has $2k$ integers, for some nonnegative integer k. Conceptually, these are k pairs of integers. If the first element of a pair is negative, then the pair represents the negated value and location (distance from start) of a pile of coins. Otherwise the pair represents the cost and jump-distance of a pogo stick.

For enumeration, your program's output should be one or more lines: The first line consists of just one integer r, which is the number of possibilities for JJ's journey with the given inputs. It is followed by r lines, where each line lists the cost of a journey followed by the sequence of pogo-stick distances used to make a journey. These r lines should appear in lexicographically sorted order. For minimum-cost computation, your program's output should be a single line for a minimum-cost journey, formatted as for the enumeration case. If there is no feasible journey, the output should be empty.

Example If the input is

```
E 5 3 5 5 10 2 1 2 3
```

it means JJ needs to cover 5 meters using pogo sticks that cover 5, 10, 1, and 3 meters, with jump-costs 3, 5, 2, and 2, respectively. In this case, the 10-meter pogo stick is useless. We can use the 5-meter one once, in just one way. That leaves the 1- and 3-meter pogo sticks. We can use the 3-meter one no more than once. If we use it zero times, then all we have is the 1-meter stick, so there is only one way: $1+1+1+1+1$; if we use it once, we have to have two uses of the 1-stick, giving the possibilities $1+1+3$, $1+3+1$, $3+1+1$. We compute the journey costs in the obvious way. So the output in this case is (note lexicographic ordering):

```
5
3 5
6 1 1 3
6 1 3 1
6 3 1 1
10 1 1 1 1 1
```

If the `E` in the input is replaced by `M`, the desired output is `b 5` denoting a minimum-cost journey with cost 3 and a single 5-meter jump. For the input `M 5 -1 0 2 1 -10 1 2 3`, the desired output is `-5 1 1 3` or `-5 1 3 1` denoting of the two minimum-cost (maximum money making) journeys.

Submission: Follow the submission procedure used for the previous assignment, replacing `hw01` with `hw02` in the obvious places.

Reminders Recall, from the previous assignment, policies on collaboration and the use of external resources. Ask for clarifications if anything is unclear. The suggestions in the previous assignment apply to this one too. Use the newsgroup for all questions and discussions unless the matter is private.