Algorithms and Complexity

First Series of Programming Exercises

Exercise 1: Outdoor Market

Context: With the holiday season approaching, you are tasked with planning the outdoor Christmas market to be set up along the main pedestrian street in the capital of the country of Algorithms. There are N outdoor stalls to be installed, and M non-overlapping suitable areas have been identified among the decorative flower beds of the walkway.

Area Definition: Each area i of the pedestrian street is defined by an interval of natural numbers [si, fi], where si is the start and fi is the end of the interval. Outdoor stalls can be installed at any integer point within these areas, with the restriction of not allowing more than one stall at the same spot.

Goal: The available space for customers of a stall is proportional to its distance from the nearest neighboring stall. The aim is to maximize this space across all stalls by finding an optimal placement of the stalls such that the minimum distance between any two neighboring stalls is as large as possible.

Input Data:

- First, the program reads two positive integers from standard input: N (number of stalls) and M (number of areas).
- For each of the next M lines, two non-negative integers si and fi are provided, indicating the start and the end of the i-th area. The areas are given in no particular order, are disjoint, and their total length is at least N.

Output Data:

- The program should output a single positive integer to the standard output, representing the minimum distance between two successive stalls in an optimal placement that maximizes this minimum distance.

Constraints:

- $1 \le N \le 10^6$
- $1 \le M \le 10^5$
- $0 \le s_i \le f_i \le 150,000,000$
- Sum of $(f_i - s_i + 1)$ from $i=1$ to $M \ge N$

Execution Time Limit: 1 second

Memory Limit: 64 MB

Example 1:

- Input:

```
5 3
5 9
0 3
12 12
```

- Output:

```
3
```

Example 2:

- Input:

```
6 3
10 12
0 4
5 8
```

- Output:

```
2
```

Exercise 2: Dual-Criteria Spanning Trees

Context: Recently, a controversy has erupted in the country of Algorithms over the usefulness and efficiency of various algorithmic techniques. Among the most adamant are the proponents of Greediness and Binary Search. The President of the country is trying to calm spirits and explain that all techniques are useful and that the efficient resolution of complex algorithmic problems usually requires a combination of algorithmic techniques. As an example, he proposes calculating a spanning tree that maximizes the ratio of the total profit to the total weight of the edges it includes.

Graph Definition: Consider an undirected, connected graph G(V,E) with |V| = N vertices and |E| = M edges. Each edge e contributes a profit p(e) and incurs a weight w(e) if included in the selected spanning tree T of G. The task is to compute a spanning tree T of G that maximizes the ratio Σ(p(e)) over Σ(w(e)) for e in T. The President claims that solving this problem efficiently requires a clever combination of algorithmic techniques and asks you to write a program that confirms this claim.

Input Data:

- Initially, your program will read two positive integers from the standard input: N (the number of vertices) and M (the number of edges) of a connected graph.
- In each of the next M lines, four positive integers u(e), v(e), p(e), w(e) will be provided, representing an edge e. The first two integers denote the vertices u(e) and v(e), with u(e) ≠ v(e), which are the ends of e. The next two integers specify the profit p(e) and the weight w(e) of edge e.

Output Data:

- Your program should print two integers to the standard output, the total profit p(T) = Σ(p(e)) and the total weight w(T) = Σ(w(e)) of the spanning tree T of G that maximizes the ratio p(T)/w(T). Specifically, your program must print the integers p(T)/gcd(p(T),w(T)) and w(T)/gcd(p(T),w(T)), separated by a space (the division by the GCD handles cases where more than one optimal spanning tree exists).

Constraints:

- 2 ≤ N ≤ 50,000
- 1 ≤ M ≤ 200,000
- 1 ≤ u(e), v(e) ≤ N
- 1 ≤ p(e), w(e) ≤ 200
- For 60% of the points, w(e) = 1.

Execution Time Limit: 3 seconds

Memory Limit: 64 MB

Example 1:

- Input:

3 3
1 2 1 3
2 3 2 2
3 1 3 1

- Output:

5 3

Example 2:

- Input:

4 5
1 2 2 3
2 3 3 1
3 4 1 2
4 1 2 1
2 4 4 4

- Output:

3 2