Algorithms and Complexity

Second Series of Programming Exercises

Exercise 1: New Year's Eve Banquet

Context: You are a member of the organizing committee for the New Year's Eve Banquet at the Presidential Palace in the Country of Algorithms, tasked with seating the guests at tables. The protocol is strict: the N guests enter the grand dining room one by one, in the order they received their invitations. Each table can accommodate up to K guests, with no limit on the number of tables you can use. Each new guest either joins the last seated guest at a table (let's call it table t, only if there are available seats at t) or is the first to sit at a new table (let's call it table t+1). In the latter case (which can occur even if table t is not full), table t is considered "closed" permanently, and subsequent guests cannot sit at it (nor at any previous ones).

Joy Multiplication: You know well that joy, love, and smiles are contagious! Thus, the degree of joy for guests sitting at the same table is equal to the highest degree of joy among them. From your research, you know the degree of joy h(1), h(2), ..., h(N) of the guests at the New Year's Eve Banquet. If now guests i, ..., j, for some $i \le j$ with $j - i + 1 \le K$, sit at the same table, the total degree of joy for that table is $(j - i + 1) \cdot \max\{i \le \ell \le j\}\{h(\ell)\}$.

Objective: As New Year's approaches and there are many guests, you need to write a program that calculates how the guests should be seated to maximize the sum of the total degree of joy for all tables.

Input Data:

- Initially, the program reads from the standard input two positive integers corresponding to the number of guests N and the capacity K of the tables.
- On the next line, there will be N natural numbers (separated by a space) corresponding to the degree of joy h(1), h(2), ..., h(N) of the guests at the New Year's Eve Banquet.

Output Data:

- The program must print to the standard output (on the first line) the maximum possible sum of the total degree of joy for all tables.

Constraints:

- $3 \le N \le 50{,}000$
- $2 \le K \le 1{,}000$
- $0 \le h(i) \le 100{,}000$
- $N \cdot \max\{h(i)\} \le 1{,}000{,}000{,}000$
- Execution time limit: 1 sec.
- Memory limit: 64 MB.

Example 1:

- Input:

```
7 3
9 15 8 9 4 7 10
```

- Output:

```
84
```

Example 2:

- Input:

```
10 4
4 14 22 6 19 14 18 20 8 16
```

- Output:

```
206
```

Example 3:

- Input:

```
10 3
4 14 22 6 19 14 18 20 8 16
```

- Output:

```
201
```

Exercise 2: Chemical Waste

Context: In a chemical laboratory, there are N different substances that constitute hazardous experimental waste and must be placed into K metal containers for safe transport to a special facility outside the laboratory. The substances are numbered from 1 to N and, for safety reasons, must be placed in the containers in this order, with the entire quantity of each substance in only one container. The containers are large enough, and the total quantity of each substance small enough, so that there are no capacity issues (i.e., even all substances could fit into the same container). However, there is a risk of chemical reaction between the substances in the same container, releasing significant amounts of energy. Specifically, for each pair of substances i and j in the same container, the chemical reaction between them produces energy equal to A[i, j] units.

Objective: The laboratory personnel follow this procedure for packaging the substances: the first t1 substances are placed in the first container, the next t2 substances in the second container, and so on, until all substances are placed in K containers. The total energy that can be released from the chemical reactions of the substances in all K containers is the sum of the above amounts. For safety reasons during transport, the laboratory personnel want to determine the indices t1, t2, ..., tK−1 where the container changes occur, such that the total energy that can be released from all the containers is minimized. A program is needed for this purpose.

Input Data:

- Initially, the program reads two positive integers from standard input, N and K, representing the number of substances and the number of containers, respectively.
- The program then reads N - 1 lines, where the ith line contains N - i integers separated by spaces. The jth integer of the ith line corresponds to the energy A[i, j + i] (the matrix A is symmetric with respect to the diagonal, i.e., A[i, j] = A[j, i] for every 1 ≤ i < j ≤ N, and the diagonal has zero elements, i.e., A[i, i] = 0 for every 1 ≤ i ≤ N).

Output Data:

- The program must print an integer to the standard output (on the first line) that corresponds to the minimum amount of energy that can be released.

Constraints:

- 0 ≤ A[i, j] ≤ 99

- $1 \leq K \leq 500$
- $K \leq N \leq 1500$
- Execution time limit: 1 sec.
- Memory limit: 64 MB.

Example Input:

3 2
3 2
4

Example Output:

3

Bonus:

- There will be input files with $1 \leq K \leq 700$ and $K \leq N \leq 2500$.