Exhibition

You are the curator of a prestigious art exhibition. You have N paintings, each with two attributes: painting size A_i and artistic value B_i. You also have M available frames, each with frame size S_j.

You want to select and arrange k paintings $i_1, i_2, \ldots, i_k$ and frames $j_1, j_2, \ldots, j_k$ for display such that:

- Each selected painting i_t is placed in frame j_t where the painting size does not exceed the frame size: $A_{i_t} \leq S_{j_t}$
- The painting sizes of selected paintings are non-decreasing in display order: $A_{i_1} \leq A_{i_2} \leq \ldots \leq A_{i_k}$
- The artistic values of selected paintings are non-decreasing in display order: $B_{i_1} \leq B_{i_2} \leq \ldots \leq B_{i_k}$

Find the maximum value of k for which a valid arrangement exists.

Implementation Details

You need to implement the following function:

```
int32 max_paintings(int32 N, int32 M, int32[] A, int32[] B, int32[] S)
```

- N: the number of paintings
- M: the number of frames
- A: array of length N, where $A[i]$ is the size of painting i
- B: array of length N, where $B[i]$ is the artistic value of painting i
- S: array of length M, where $S[j]$ is the size of frame j
- The function should return the maximum number of paintings that can be displayed

Constraints

- $1 \leq N, M \leq 10^5$
- $1 \leq A_i, B_i, S_j \leq 10^9$ for all valid indices

Scoring

- **Subtask 1** (10 points): $N, M \leq 10$

- **Subtask 2** (20 points): All frame sizes are larger than all painting sizes ($S_j > A_i$ for all i, j)
- **Subtask 3** (20 points): All artistic values are equal ($B_i = B_j$ for all i, j)
- **Subtask 4** (20 points): $N, M \leq 2000$
- **Subtask 5** (30 points): No additional constraints

Examples

The following call `max_paintings(3, 3, [1, 2, 3], [1, 2, 4], [2, 3, 5])` should return `3`

- We have 3 paintings with sizes $[1, 2, 3]$ and artistic values $[1, 2, 4]$.
- We have 3 frames with sizes $[2, 3, 5]$.
- We can select all 3 paintings: painting 1 (size 1, value 1) in frame 1 (size 2), painting 2 (size 2, value 2) in frame 2 (size 3), and painting 3 (size 3, value 4) in frame 3 (size 5).
- The sizes are non-decreasing: $1 \leq 2 \leq 3$ and the artistic values are non-decreasing: $1 \leq 2 \leq 4$.

The following call `max_paintings(4, 3, [1, 3, 2, 4], [3, 2, 3, 5], [3, 6, 4])` should return `3`

- We have 4 paintings with sizes $[1, 3, 2, 4]$ and artistic values $[3, 2, 3, 5]$.
- We have 3 frames with sizes $[3, 6, 4]$.
- We can select paintings with indices 1, 3, and 4: painting 1 (size 1, value 3) in frame 1 (size 3), painting 3 (size 2, value 3) in frame 3 (size 4), and painting 4 (size 4, value 5) in frame 2 (size 6).
- The sizes are non-decreasing: $1 \leq 2 \leq 4$ and the artistic values are non-decreasing: $3 \leq 3 \leq 5$.

The following call `max_paintings(4, 3, [1, 3, 2, 4], [3, 2, 3, 5], [1, 1, 4])` should return `2`

- We have 4 paintings with sizes $[1, 3, 2, 4]$ and artistic values $[3, 2, 3, 5]$.
- We have 3 frames with sizes $[1, 1, 4]$.
- We can select painting 1 (size 1, value 3) in frame 1 or 2 (size 1), and painting 4 (size 4, value 5) in frame 3 (size 4).
- The sizes are non-decreasing: $1 \leq 4$ and the artistic values are non-decreasing: $3 \leq 5$.

Sample Grader

The sample grader reads the input in the following format:

- Line 1: Two integers N and M
- Line 2: N integers $A_1, A_2, \ldots, A_N$ (painting sizes)
- Line 3: N integers $B_1, B_2, \ldots, B_N$ (artistic values)
- Line 4: M integers $S_1, S_2, \ldots, S_M$ (frame sizes)

The sample grader calls `max_paintings(N, M, A, B, S)` and prints the returned value.

Note: The sample grader provided with this problem is just for testing your solution locally. The actual grader used during the contest may be different.