축제

Nayra는 볼리비아의 유명한 호수인 라구나 콜로라다 여행권이 걸린 게임 축제에 참가 중이다. 이 게임은 토큰을 이용하여 쿠폰을 사는 것이다. 쿠폰을 사는 것은 추가적인 토큰을 얻을 수 있다. 목표는 가능한 많은 쿠폰을 갖는 것이다.

그녀는 A개의 토큰을 갖고 게임을 시작한다. 쿠폰은 N가지가 있고 0부터 $N-1$까지로 구분된다. Nayra는 쿠폰 i ($0 \leq i < N$)를 사기 위해 토큰 $P[i]$개를 지불해야 한다(그리고 구매 전에 최소한 $P[i]$개의 토큰을 갖고 있어야 한다). 그녀는 각 쿠폰을 최대 한 번만 살 수 있다.

게다가, 각 쿠폰 i ($0 \leq i < N$)는 몇 가지 **종류** 중 하나인데, 이 종류는 $T[i]$로 표시되며 **1부터 4까지** 정수 중 하나이다. Nayra가 쿠폰 i를 산 후, 그녀가 가진 남은 토큰의 개수는 $T[i]$와 곱해진다. 정확히 표현하면, 게임 중 어떤 순간에 그녀가 토큰 X개를 갖고 있고 ($X \geq P[i]$를 만족하는) 쿠폰 i를 산다면, 구매 후 그녀는 토큰 $(X - P[i]) \cdot T[i]$개를 갖게 된다.

당신이 할 일은 게임이 끝났을 때 Nayra가 갖는 **쿠폰** 개수가 최대가 되도록 어떤 쿠폰을 어떤 순서로 사야하는지 결정하는 것이다. 만약 이런 구매 순서가 여러개 존재한다면, 당신은 그 중 한가지만 구하면 된다.

Implementation Details

다음 함수를 구현해야 한다.

```
std::vector<int> max_coupons(int A, std::vector<int> P,
    std::vector<int> T)
```

- A: Nayra가 처음에 가진 토큰 개수.
- P: 쿠폰 가격을 나타내는 길이 N인 배열.
- T: 쿠폰 종류를 나타내는 길이 N인 배열.
- 이 함수는 각 테스트 케이스에 대해 정확히 한번 호출된다.

이 함수는 다음과 같이 Nayra의 구매 정보를 나타내는 배열 R을 리턴해야 한다:

- R의 길이는 그녀가 살 수 있는 쿠폰의 최대 개수여야 한다.
- 배열의 원소는 그녀가 사는 쿠폰의 번호이며 구매 순서대로 저장되어 있다. 즉, 그녀는 쿠폰 $R[0]$을 처음에 사고, 쿠폰 $R[1]$을 두 번째로 사며, 이후 같은 방식이 된다.
- R의 모든 원소는 유일해야 한다.

만약 아무 쿠폰도 살 수 없다면, R은 빈 배열이어야 한다.

Constraints

- $1 \le N \le 200\,000$
- $1 \le A \le 10^9$
- $1 \le P[i] \le 10^9$ ($0 \le i < N$인 각 i에 대해).
- $1 \le T[i] \le 4$ ($0 \le i < N$인 각 i에 대해).

Subtasks

Subtask	Score	Additional Constraints
1	5	$T[i] = 1$ ($0 \le i < N$인 각 i에 대해).
2	7	$N \le 3000$; $T[i] \le 2$ ($0 \le i < N$인 각 i에 대해).
3	12	$T[i] \le 2$ ($0 \le i < N$인 각 i에 대해).
4	15	$N \le 70$
5	27	Nayra는 (어떤 순서로) 모든 N가지 쿠폰을 살 수 있다.
6	16	$(A - P[i]) \cdot T[i] < A$ ($0 \le i < N$인 각 i에 대해).
7	18	추가적인 제한이 없다.

Examples

Example 1

다음 호출을 생각해보자.

```
max_coupons(13, [4, 500, 8, 14], [1, 3, 3, 4])
```

Nayra는 처음에 토큰 $A = 13$개를 갖고 있다. 그녀는 아래와 같은 순서로 3개의 쿠폰을 살 수 있다:

산 쿠폰	쿠폰 가격	쿠폰 종류	구매 후 토큰 개수
2	8	3	$(13 - 8) \cdot 3 = 15$
3	14	4	$(15 - 14) \cdot 4 = 4$
0	4	1	$(4 - 4) \cdot 1 = 0$

이 예에서, Nayra가 3개의 쿠폰보다 더 많이 사는 것은 불가능하며, 위에 나타낸 구매 순서가 이들 3개를 살 수 있는 유일한 방법이다. 따라서, 함수는 $[2, 3, 0]$을 리턴해야 한다.

Example 2

다음 호출을 생각해보자.

```
max_coupons(9, [6, 5], [2, 3])
```

이 예에서, Nayra를 두 쿠폰을 어떤 순서로든 살 수 있다. 따라서, 함수는 $[0, 1]$ 또는 $[1, 0]$을 리턴해야 한다.

Example 3

다음 호출을 생각해보자.

```
max_coupons(1, [2, 5, 7], [4, 3, 1])
```

이 예에서, Nayra는 토큰 한 개를 갖고 있고 이걸로는 어떤 쿠폰도 살 수 없다. 따라서, 함수는 [ ] (빈 배열)을 리턴해야 한다.

Sample Grader

입력 형식:

```
N A
P[0]  T[0]
P[1]  T[1]
...
P[N-1]  T[N-1]
```

출력 형식:

```
S
R[0]  R[1]  ...  R[S-1]
```

여기에서, S는 `max_coupons`가 리턴하는 배열 R의 길이이다.