Problem 2398: Maximum Number of Robots Within Budget

Problem Information

Difficulty: Hard
Acceptance Rate: 37.56%
Paid Only: No
Tags: Array, Binary Search, Queue, Sliding Window, Heap (Priority Queue), Prefix Sum, Monotonic Queue

Problem Description

You have `n` robots. You are given two **0-indexed** integer arrays, `chargeTimes` and `runningCosts`, both of length `n`. The `ith` robot costs `chargeTimes[i]` units to charge and costs `runningCosts[i]` units to run. You are also given an integer `budget`.

The **total cost** of running `k` chosen robots is equal to `max(chargeTimes) + k * sum(runningCosts)`, where `max(chargeTimes)` is the largest charge cost among the `k` robots and `sum(runningCosts)` is the sum of running costs among the `k` robots.

Return _the**maximum** number of **consecutive** robots you can run such that the total cost **does not** exceed _`budget`.

Example 1:

Input: chargeTimes = [3,6,1,3,4], runningCosts = [2,1,3,4,5], budget = 25 **Output:** 3 **Explanation:** It is possible to run all individual and consecutive pairs of robots within budget. To obtain answer 3, consider the first 3 robots. The total cost will be max(3,6,1) + 3 * sum(2,1,3) = 6 + 3 * 6 = 24 which is less than 25. It can be shown that it is not possible to run more than 3 consecutive robots within budget, so we return 3.

Example 2:

Input: chargeTimes = [11,12,19], runningCosts = [10,8,7], budget = 19 **Output:** 0 **Explanation:** No robot can be run that does not exceed the budget, so we return 0.

Constraints:

* `chargeTimes.length == runningCosts.length == n` * `1 <= n <= 5 * 104` * `1 <= chargeTimes[i], runningCosts[i] <= 105` * `1 <= budget <= 1015`

Code Snippets

C++:

```
class Solution {
public:
int maximumRobots(vector<int>& chargeTimes, vector<int>& runningCosts, long
long budget) {

}
};
```

Java:

```
class Solution {
public int maximumRobots(int[] chargeTimes, int[] runningCosts, long budget)
{

}
}
```

Python3:

```
class Solution:
def maximumRobots(self, chargeTimes: List[int], runningCosts: List[int],
budget: int) -> int:
```