Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Department Informatik
Markus Püschel David Steurer Peter Widmayer
Chih-Hung Liu
Stefano Leucci

20. November 2017

Datenstrukturen & Algorithmen Blatt P9 HS 17

Hand-in: Bis Sonntag, 3. Dezember 2017, 23:59 Uhr via Online Judge (nur Source Code). Fragen zur Aufgabenstellung oder Übersetzung werden wie üblich im Forum beantwortet.

Exercise P9.1 *Dyno.*

Dyno, the dinosaur of Figure 1, wants to cross a perfectly straight desert. The desert is L meters long and it is split into L segments, indexed from 0 to $L-1$. Each segment can either be *empty* or it can contain one of the C *cacti* that inhabit the desert. When Dyno is in a generic segment i it can either walk or jump forward. Walking allows it to move from segment i to segment $i+1$, provided that segment $i+1$ is empty. Jumping allows it to move from segment i to segment $i+D$, provided that segment $i+D$ is empty (Dyno can jump even if there are cacti between segment $i+1$ and segment $i+D-1$. Dyno starts at segment 0, which is always empty, and your job is to help Dyno reach the farthest possible segment in the desert (i.e., the one with the largest possible index).

Input The input consists of a set of instances, or *test-cases*, of the previous problem. The first line of the input contains the number T of test-cases. The first line of each test case contains integers L, D and C, separated by spaces. The second line of each test case contains the locations of the C cacti as n integers separated by spaces. The segment numbers of the cacti locations appear in increasing order.

The inputs satisfy $10 \leq L \leq 1\,000\,000$, $0 \leq C \leq 1\,000\,000$, and $2 \leq D \leq 10$.

Output The output consists of T lines, each containing a single integer. The i-th line is the answer to the i-th test-case, i.e., it contains the the largest index reachable by Dyno.

Grading You get 3 bonus points if your program works for all inputs. Your program should run in time $O(L \log(C+D))$. Submit your `Main.java` at `https://judge.inf.ethz.ch/team/websubmit.php?cid=18997&problem=DA17P4.5`. The enrollment password is "`asymptotic`".

Example

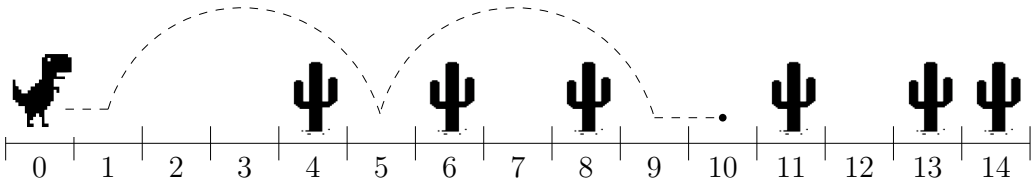

Figure 1: Example input. The dashed line represents the unique optimal solution.[1]

Input (corresponding to Figure 1):

```
1
15 4 6
4 6 8 11 13 14
```

Output:

```
10
```

Notes For this exercise we provide an archive on the lecture website containing a program template that will load the input and write the output for you. The archive also contains additional test cases (which differ from the ones used for grading). Importing or using classes that are not in `java.lang.*` is **not allowed** (with the exception of the already imported `java.util.Scanner` class).

[1] Dyno is not to scale.

Exercise P9.2 *Light Coffee.*

Alice is visiting the city of Algo and wants to buy a cup of Java Coffee that costs $C > 0$ *Flops* (the currency of the city of Algo). Alice's wallet contains n coins. The i-th coins is worth $v_i > 0$ Flops and weighs $w_i > 0$ *Grahams* (the weight unit of the city of Algo). She wants to pay exactly C Flops using an even number of coins while also lightening her wallet as much as possible.

Your task is to design an algorithm that computes the largest number of Grahams that Alice can remove from her wallet. You can assume that there is at least one way to pay for the coffee using an even number of coins.

Input The input consists of a set of instances, or *test-cases*, of the previous problem. The first line of the input contains the number T of test-cases, and each test-case consists of 3 lines. The first line of each test-case contains the two integers C and n. The second line contains n integers, where the i-th integer is the value v_i of the i-th coin. The third line contains n integers, where the i-th integer is the weight w_i of the i-th coin.

Output The output consists of T lines. The i-th line is the answer to the i-th test-case and contains the maximum total weight of an even number of coins whose values sum to exactly C.

Grading You get 3 bonus points if your program works for all inputs. Your algorithm should require $O(C \cdot n)$ time (with reasonable hidden constants). Submit your `Main.java` at `https://judge.inf.ethz.ch/team/websubmit.php?cid=18997&problem=DA17P4.6`. The enrollment password is "`asymptotic`".

Subtask 1: You are encouraged to develop a solution that works for every instance of the problem. However, the judge will award 1 bonus point (out of the 3 maximum points) if your program correctly solves the following (easier) subtask: *compute the largest number of Grahams that Alice can remove from her wallet in the special case in which all coins weigh exactly* 1 *Graham.*[2]

Example

Input:

```
2
10 8
4 3 3 5 1 2 7 12
2 1 3 15 3 5 9 4
26 7
1 2 4 13 8 7 23
1 5 8 1 2 4 30
```

Output:

```
13
18
```

Notes For this exercise we provide an archive on the lecture website containing a program template that will load the input and write the output for you. The archive also contains additional test cases (which differ from the ones used for grading). Importing or using classes that are not in `java.lang.*` is **not allowed** (with the exception of the already imported `java.util.Scanner` class).

[2] Alice still needs to pay exactly C Flops using an even number of coins.