International Olympiad in Informatics
2013
6-13 July 2013
Brisbane, Australia
Day 2 tasks

robots

Japanese — 1.0

Marita の弟がリビングルーム全体におもちゃを置きっぱなしにしている！ 幸運にも，Marita はおもちゃを片付ける特殊なロボットを開発していた．彼女はどのロボットがおもちゃを片付けるかを決めるのにあなたの助けを必要としている．

おもちゃが `T` 個ある．各おもちゃの重さは整数 `W[i]` であり，大きさは整数 `S[i]` である． ロボットには **弱い (weak)** ロボットと **小さい (small)** ロボットの 2 種類がある．

- `A` 台の弱いロボットがある．それぞれの弱いロボットには重さの制限 `X[i]` が定まっている．弱いロボットは，おもちゃの大きさに関わらず，重さが `X[i]` 未満のどのおもちゃも運ぶことができる．
- `B` 台の小さいロボットがある．それぞれの小さいロボットには大きさの制限 `Y[i]` が定まっている．小さいロボットは，おもちゃの重さに関わらず，大きさが `Y[i]` 未満のどのおもちゃも運ぶことができる．

Marita のロボットは 1 つのおもちゃを片付けるのに 1 分かかる．異なるロボットは同時に異なるおもちゃを片付けることができる．

あなたの課題は，Marita のロボットたちがすべてのおもちゃを片づけることが可能かを判定し，また可能ならばそれにかかる時間の最小値を求めることである．

例 (Examples)

1 つ目の例として，弱いロボットが `A = 3` 台と小さいロボットが `B = 2` 台あるとする．弱いロボットの重さの制限は `X = [6, 2, 9]` であり，小さいロボットの大きさの制限は `Y = [4, 7]` であるとする．また，次に述べるような `T = 10` 個のおもちゃがあるとする：

おもちゃの番号	0	1	2	3	4	5	6	7	8	9
重さ	4	8	2	7	1	5	3	8	7	10
大きさ	6	5	3	9	8	1	3	7	6	5

すべてのおもちゃを片付けるのにかかる時間の最小値は 3 分である：

	弱いロボット 0	弱いロボット 1	弱いロボット 2	小さいロボット 0	小さいロボット 1
1 分後	おもちゃ 0	おもちゃ 4	おもちゃ 1	おもちゃ 6	おもちゃ 2
2 分後	おもちゃ 5		おもちゃ 3		おもちゃ 8
3 分後			おもちゃ 7		おもちゃ 9

2 つ目の例として，弱いロボットが `A = 2` 台と小さいロボットが `B = 1` 台あるとする．弱いロボットの重さの制限は `X = [2, 5]` であり，小さいロボットの大きさの制限は `Y = [2]` であるとする．また，次に述べるような `T = 3` 個のおもちゃがあるとする：

おもちゃの番号	0	1	2
重さ	3	5	2
大きさ	1	3	2

どのロボットも重さが 5 で大きさが 3 のおもちゃを片付けることができないので，ロボットたちがすべてのおもちゃを片付けることはできない．

実装 (Implementation)

あなたは，次のような関数 `putaway()` を実装した 1 つのファイルを提出しなければならない．

あなたの関数 (Your Function) : `putaway()`

C/C++

```
int putaway(int A, int B, int T,
        int X[], int Y[], int W[], int S[]);
```

Pascal

```
function putaway(A, B, T : LongInt;
        var X, Y, W, S : array of LongInt) : LongInt;
```

説明 (Description)

この関数は，ロボットたちがすべてのおもちゃを片付けるのに必要な時間の最小値を計算して返さなければならない．ただし，それが不可能ならば `-1` を返さなければならない．

引数と戻り値 (Parameters)

- `A`：弱いロボットの数．
- `B`：小さいロボットの数．

- `T`：おもちゃの数.
- `X`：弱いロボットそれぞれの重さの制限を表す整数からなる長さ `A` の配列.
- `Y`：小さいロボットそれぞれの大きさの制限を表す整数からなる長さ `B` の配列.
- `W`：各おもちゃの重さを表す整数からなる長さ `T` の配列.
- `S`：各おもちゃの大きさを表す整数からなる長さ `T` の配列.
- **Returns** (戻り値)：ロボットたちがすべてのおもちゃを片付けるのに必要な時間の最小値．ただし，それが不可能ならば `-1` を返すこと.

やりとりの例 (Sample Session)

次のやりとりは，上に述べた 1 つ目の例を表す.

引数	値
A	`3`
B	`2`
T	`10`
X	`[6, 2, 9]`
Y	`[4, 7]`
W	`[4, 8, 2, 7, 1, 5, 3, 8, 7, 10]`
S	`[6, 5, 3, 9, 8, 1, 3, 7, 6, 5]`
戻り値	`3`

次のやりとりは，上に述べた 2 つ目の例を表す.

引数	値
A	`2`
B	`1`
T	`3`
X	`[2, 5]`
Y	`[2]`
W	`[3, 5, 2]`
S	`[1, 3, 2]`
戻り値	`-1`

制限 (Constraints)

- 時間制限：3 秒
- メモリ制限：64 MiB
- $1 \le T \le 1{,}000{,}000$
- $0 \le A, B \le 50{,}000$ かつ $1 \le A + B$
- $1 \le X[i], Y[i], W[i], S[i] \le 2{,}000{,}000{,}000$

小課題 (Subtasks)

小課題	得点	入力に関する追加の制約
1	14	$T = 2$ かつ $A + B = 2$ (ちょうど 2 個のおもちゃと，ちょうど 2 台のロボットがある)
2	14	$B = 0$ (すべてのロボットは弱い)
3	25	$T \le 50$ かつ $A + B \le 50$
4	37	$T \le 10{,}000$ かつ $A + B \le 1{,}000$
5	10	(なし)

試行 (Experimentation)

与えられる採点プログラムのサンプルは，ファイル `robots.in` から入力を読み込む．このファイルは次のフォーマットで書かれていなければならない：

- 1 行目：`A B T`
- 2 行目：`X[0] … X[A-1]`
- 3 行目: `Y[0] … Y[B-1]`
- 続く `T` 行: `W[i] S[i]`

例えば，上に述べた 1 つ目の例は次のフォーマットで与えられなければならない：

```
3 2 10
6 2 9
4 7
4 6
8 5
2 3
7 9
1 8
5 1
3 3
8 7
7 6
10 5
```

もし `A = 0` または `B = 0` ならば，対応する行 (行 2 または行 3) は空行でなければならない．

言語に関する注意 (Language Notes)

C/C++ `#include "robots.h"` を行うこと．

Pascal `unit Robots` を定義すること．すべての配列は `0` から始まる (`1` からではない)．

実装例については，あなたの計算機上に与えられている解法テンプレートを参照すること．