

Opérations préliminaires



Créez un nouveau répertoire sur le disque dur local en utilisant votre nom et prénom pour le nommer. Créez trois sous-répertoires pour les trois exercices proposés. Nommez ces sous-répertoires LISTES, RECTANGLE et JEU. Veillez à sauvegarder vos programmes, fichiers d'entrée et de sortie ainsi que vos exécutables dans les répertoires correspondants.

Problème I - LISTES

20 points

Soient deux listes de lettres L_1 et L_2 . La liste L_1 est incluse dans L_2 si toutes les lettres de L_1 sont comprises dans L_2 avec les spécifications suivantes:

- les lettres de L_1 dans L_2 doivent apparaître soit dans l'ordre, soit dans l'ordre inverse;
- les lettres de L_1 dans L_2 n'ont pas besoin d'être séquentielles.

Dans l'exemple suivant, la liste L_1

P	a	s	c	a	l
---	---	---	---	---	---

est incluse dans toutes les listes L_2 suivantes

O	u	R	d	F	F	F	a	P	a	s	c	a	l	B	z
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

f	P	a	e	Q	s	a	R	R	e	c	a	l	D	p	x	x	W
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

l	e	F	L	Q	q	Y	c	c	c	S	C	k	a	c	s	a	t	E	P	j	R
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

l	a	c	s	a	P
---	---	---	---	---	---

Problème

Ecrivez un programme qui détermine si une liste de lettres L_1 est incluse dans une liste L_2 .

Entrée

Le programme lit les deux listes dans le fichier LISTES.TXT. Ce fichier contient deux lignes avec les deux listes, d'abord L_1 et ensuite L_2 .

Sortie

Le programme affiche simplement sur l'écran si oui ou non il y a inclusion à l'aide des messages "Inclusion!" ou "Pas d'inclusion!".

Contraintes

$$1 \leq \text{long}(L_1) \leq 255$$

$$1 \leq \text{long}(L_2) \leq 255$$

$$\text{long}(L_1) \leq \text{long}(L_2)$$

Votre programme doit faire la différence entre lettres minuscules et majuscules.

 Remettez le programme sous le nom LISTES.xxx, avec xxx=PAS ou C(PP). Remettez également le fichier binaire exécutable LISTES.EXE correspondant au programme.

Soit un tableau d'une taille $m \times n$ (m colonnes et n lignes), appelé rectangle, contenant comme éléments des entiers relatifs.

Le terme de sous-rectangle définit une portion quelconque du rectangle. Le sous-rectangle le plus petit possède ainsi une taille de 1×1 et contient un seul élément et le plus grand sous-rectangle possède une taille de $m \times n$ et contient $m \times n$ éléments.

On appelle sous-rectangle maximal le sous-rectangle dont la somme des éléments est la plus grande.

Exemple

Le sous-rectangle maximal du rectangle

0	-2	-7	0
9	2	-6	2
-4	1	-4	1
-1	8	0	-2

se situe en bas à gauche.

9	2
-4	1
-1	8

et a une somme de 15.

Problème

Ecrivez un programme qui trouve le sous-rectangle maximal pour un rectangle donné et qui détermine sa somme.

Entrée

L'entrée consiste dans un fichier texte RECTIN.TXT.

La première ligne du fichier contient deux entiers naturels non nuls séparés par un espace, indiquant la taille $m \times n$ du rectangle donné.

Les n lignes suivantes contiennent chacune m entiers relatifs, indiquant les $m \times n$ éléments du rectangle donné.

Sortie

La sortie consiste dans un fichier texte RECTOUT.TXT.

La seule ligne contient un entier relatif, indiquant la somme des éléments du sous-rectangle maximal.

Exemple d'entrée et de sortie

Pour l'exemple donné plus haut, on a:

RECTIN.TXT

4	4		
0	-2	-7	0
9	2	-6	2
-4	1	-4	1
-1	8	0	-2

RECTOUT.TXT

15

Restrictions

$1 \leq m \leq 100$ et $1 \leq n \leq 100$

Les éléments du rectangle sont compris dans l'intervalle entier $[-128 ; 127]$

 Remettez le programme sous le nom RECTANGLE.xxx, avec xxx=PAS ou C(PP). Remettez également le fichier binaire exécutable RECTANGLE.EXE correspondant au programme.

On donne une liste de N nombres entiers compris entre 1 et 1000. N est un nombre pair.

Vous devez programmer un jeu qui se déroule entre votre programme et un utilisateur qui choisissent tour à tour un nombre de la liste - soit au début soit à la fin de la liste. Votre programme commence le jeu en choisissant le premier nombre. A la fin du jeu, donc si la liste est vide, on compare les sommes des nombres choisis. Votre programme a gagné le jeu si sa somme est supérieure ou égale à celle de l'utilisateur.

Votre programme doit gagner toutes les parties!

Le déroulement du jeu est affiché à l'écran tour à tour, où un tour comporte les 6 pas que voici:

afficher la liste
afficher le choix de votre programme
afficher la somme du programme obtenue jusqu'ici
afficher la liste
demander le choix de l'utilisateur
afficher la somme de l'utilisateur obtenue jusqu'ici

Exemple d'un déroulement du jeu

Voici un exemple pour N = 6.

```
Liste: 2 43 14 15 16 1
Je prends à droite.
Ma somme est 1
Liste: 2 43 14 15 16
Où prenez-vous (g/d)? d
Votre somme est 16
Liste: 2 43 14 15
Je prends à droite.
Ma somme est 16
Liste: 2 43 14
Où prenez-vous (g/d)? d
Votre somme est 30
Liste: 2 43
Je prends à droite.
Ma somme est 59
Liste: 2
Où prenez-vous (g/d)? g
Votre somme est 32
Je gagne!
```

Entrée

L'entrée consiste dans un fichier texte JEUN.TXT.

La seule ligne contient d'abord la taille N de la liste et ensuite les N nombres de la liste. Tous les nombres sont séparés par un espace.

Exemple d'entrée

Pour l'exemple donné plus haut, on a:

JEUN.TXT

6 2 43 14 15 16 1

Restrictions

$$2 \leq N \leq 10000$$

 Remettez le programme sous le nom JEU.xxx, avec xxx=PAS ou C(PP). Remettez également le fichier binaire exécutable JEU.EXE correspondant au programme.