



CONCOURS D'ENTRÉE

CYCLE INGENIEUR

OPTION : INFORMATIQUE

MP / PC / PSI / PT / TSI

Samedi 15 Avril 2017

Durée : 2 Heures

Condition(s) particulière(s)

Calculatrice interdite
Remettre le QCM avec vos copies d'examen

A) Piles et garages ...

Imaginons un garage possédant deux voies d'entrée et une seule de sortie (voir Figure 1). Pour garer sa voiture il existe 2 voies d'accès (*entrée e1* et *entrée e2*) et pour la ressortir une seule voie (*sortie s*).

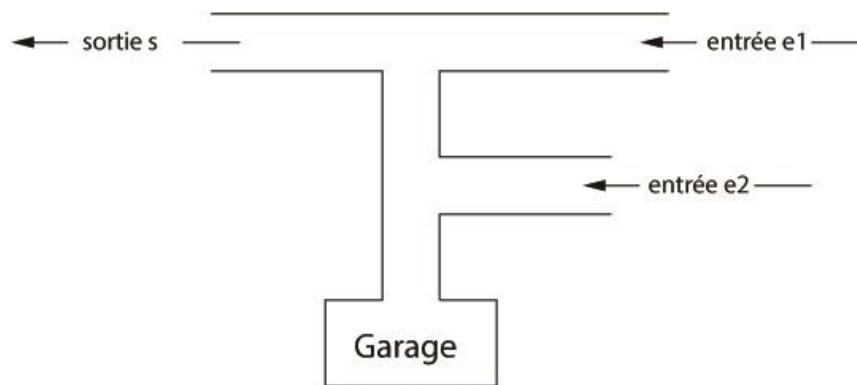


Figure 1 - Garage à deux entrées et une sortie

Les mouvements possibles des voitures sont :

- une voiture peut entrer par l'*entrée e1*
- une voiture peut entrer par l'*entrée e2*
- une voiture ne peut sortir que par la *sortie s*
- une voiture ne peut ressortir que si elle est la dernière entrée (peu importe l'entrée)
- une voiture ne peut pas sauter par dessus une autre

Remarque : Il n'y a pas de problème de voitures se retrouvant face à face.

Le garage fonctionne comme une pile à double entrée.

Symbolisons une entrée à l'aide du couple ($v_{\text{numéro du véhicule}}$, $e_{\text{numéro de l'entrée empruntée}}$) et la sortie simplement par la lettre s .

On réalise la suite d'actions suivante:

$(v_1, e_1), (v_2, e_1), (v_3, e_2), (v_4, e_1), s, s, (v_5, e_2), (v_6, e_2), s, s, s, (v_7, e_1), (v_8, e_2), (v_9, e_2), s, s, s, s$

L'ordre de sortie des voitures est donc:

$v_4, v_3, v_6, v_5, v_2, v_9, v_8, v_7, v_1$

- 1) A partir d'un garage vide, est-ce que les séquences d'entrées/sorties suivantes de véhicules sont valides ?

a) $(v_1, e_1), (v_2, e_1), (v_3, e_1), s, s, (v_4, e_2), (v_5, e_1), s, s, s, (v_6, e_2), s$

b) $(v_1, e_1), (v_2, e_2), s, (v_3, e_2), s, s, s, (v_4, e_1), (v_5, e_2), s, (v_6, e_1), (v_7, e_2), s, s$

Dans les cas où la séquence n'est pas valide, indiquer brièvement pourquoi.

- 2) On peut symboliser l'action "entrer une voiture" par les symboles *E1* ou *E2* (empiler) suivant l'entrée empruntée et l'action "sortir une voiture" par le symbole *D* (dépiler).

La suite d'actions présentée au début peut alors être symbolisée par la séquence suivante :

E1E1E2E1DDE2E2DDDE1E2E2DDDD

Toujours à partir d'un garage vide, donner une règle qui caractériserait les séquences admissibles.

B) Dichotomie : « Chemin » de recherche ...

Supposons des listes d'entiers triées en ordre croissant. Si on effectuait dans celles-ci une recherche dichotomique de la valeur 66: Parmi les séquences suivantes, lesquelles ne pourraient pas correspondre à la suite des valeurs rencontrées lors de la recherche?

- a) 46 - 65 - 81 - 73 - 70 - 66
- b) 31 - 62 - 90 - 72 - 61 - 66
- c) 36 - 70 - 53 - 50 - 61 - 66
- d) 35 - 51 - 55 - 58 - 61 - 66

C) Codage RLE simplifié ...

Le but de cet exercice est d'écrire les fonctions de compression/décompression en utilisant une version simplifiée de l'algorithme RLE (Run Length Encoding).

La compression RLE standard permet de compresser des éléments par factorisation, mais uniquement lorsque ceci permet de gagner de la place. Votre algorithme RLE simplifié effectuera cette factorisation dans tous les cas, y compris quand cela prend plus de place que l'objet non compressé.

Le flux que l'on encode est une liste d'éléments; le flux encodé est une liste de couples, chaque couple étant composé du nombre d'éléments consécutifs identiques, puis de cet élément (Voir les exemples associés aux questions suivantes).

- 1) Écrire la fonction python `decodeRLE` qui décompresse une liste compressée en RLE.

Exemples d'application:

```
>>> decodeRLE([(6, 'grr')])
['grr', 'grr', 'grr', 'grr', 'grr', 'grr']
```

```
>>> decodeRLE([(5, 'a'), (1, 'b'), (3, 'c'), (2, 'd'), (1, 'e')])
['a', 'a', 'a', 'a', 'a', 'b', 'c', 'c', 'c', 'd', 'd', 'e']
```

- 2) Écrire la fonction python encodeRLE ci-dessous qui compresse une liste en utilisant l'algorithme RLE.

Exemples d'application:

```
>>> encodeRLE(['grr', 'grr', 'grr', 'grr', 'grr', 'grr'])  
[(6, 'grr')]
```

```
>>> encodeRLE(['a','a','a','a','a','b','c','c','c','d','d','e'])  
[(5, 'a'), (1, 'b'), (3, 'c'), (2, 'd'), (1, 'e')]
```

D) Des matrices ...

1	10	3	0	3	10	1
1	0	1	8	1	0	1
10	9	14	1	14	9	10
10	3	7	11	7	3	10
7	8	5	1	5	8	7

Figure 2 – Mat1

1	10	3	3	10	1
1	0	1	1	0	1
10	9	4	4	9	10
10	3	7	7	3	10
7	8	15	15	8	7

Figure 3 – Mat2

1	24	12	18	4
10	15	15	0	18
8	14	0	16	2
22	4	8	14	22
19	7	23	5	5

Figure 4 – Mat3

Pour les questions suivantes, les matrices sont supposées non vides.

- 1) Écrire la fonction python posMinimax qui cherche la valeur minimale parmi les maximums de chaque ligne d'une matrice d'entiers et retourne la position de la valeur cherchée. En cas de réponses multiples, vous conserverez la première rencontrée.

Exemples d'applications sur les matrices des figures 2,3 et 4:

```
>>> posMinimax(Mat1)  
(1, 3)  
  
>>> posMinimax(Mat2)  
(1, 0)  
  
>>> posMinimax(Mat3)  
(2, 3)
```

- 2) Écrire la fonction python symmetric qui vérifie si une matrice est symétrique selon un axe vertical (symétrie horizontale).

Exemples d'applications sur les matrices des figures 2, 3 et 4:

```
>>> symmetric(Mat1)  
True  
  
>>> symmetric(Mat2)  
True  
  
>>> symmetric(Mat3)  
False
```