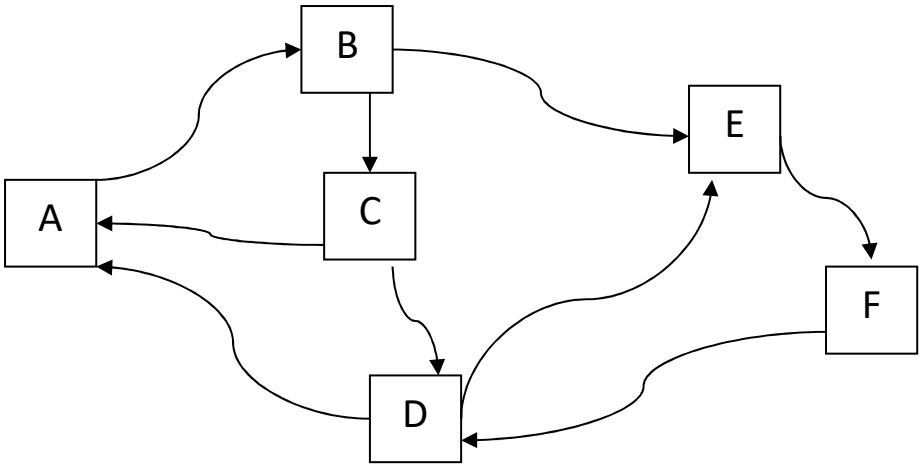
PISTES DE SKI

Présentation du sujet :

On considère un domaine de ski qui est composé de six carrefours que l'on note A, B, C, D, E et F. Le schéma ci-dessous permet de visualiser les 9 pistes ouvertes entre ces six carrefours.



Pour chaque piste un sens de parcours est imposé.

Ces neufs pistes ouvertes sont répertoriées dans un tableau de caractères de deux lignes et neuf colonnes ci-dessous.

'A'	'B'	'B'	'C'	'C'	'D'	'D'	'E'	'F'
'B'	'C'	'E'	'A'	'D'	'A'	'E'	'F'	'D'

Chaque colonne correspond à une piste ouverte avec son début en première ligne et sa fin en deuxième ligne.

Exemple : la deuxième colonne correspond à la piste ouverte entre le carrefour B et le carrefour C.

Ce tableau est stocké dans la variable **piste.**

piste = [['A' , 'B', 'B', 'C', 'C', 'D', 'D', 'E', 'F'] , ['B', 'C', 'E', 'A', 'D', 'A', 'E', 'F', 'D']]

Un parcours est une suite ordonnée de pistes, stocké sous la forme d'une chaine de caractères correspondant aux carrefours parcourus lors de celui-ci. Un parcours est dit valide s'il n'est constitué que de pistes ouvertes.

Exemple : Le parcours entre B et F passant par le carrefour E sera stocké dans la variable **parcours1 = 'BEF'**, ce parcours est un parcours valide (possible compte tenu de l'ouverture et du sens de parcours des pistes)

On note les parcours suivants : **parcours1 = 'BEF'**

parcours2 = 'ABCDFDA'

parcours3 = 'DABEFDA'

On se propose d'écrire un code permettant de déterminer les pistes et les parcours valides entre les différents carrefours.

PARTIE A :

Utilisation d'un ordinateur interdite. Les réponses sont à rédiger sur les feuilles réponses

Durée : 30 minutes

Question A1

On considère la fonction ci-dessous qui prend en argument deux caractères et retourne un booléen.

```
fonction piste_ouverte (debut : de type caractère, fin : de type caractère):
        variables globale :   pistes de type tableau
        variables locales :
                i : de type entier
                valide : de type booléen
        Debut de fonction :
                pour i allant de 0 à 8 :
                        si (pistes[0][i]=debut et pistes[1][i]=fin) alors :
                                valide ← vrai
                        sinon :
                                valide ← faux
                        fin de si
                fin de pour

                retourner valide
        fin de fonction
```

1) On exécute cette fonction avec comme paramètres **debut= 'B'** et **fin ='E'** et la variable globale **pistes** de la page 2. Compléter le jeu d'essais en feuille réponse.
2) Expliquer pourquoi la fonction retourne faux alors que la piste entre B et E est ouverte.
3) Proposer une modification de la fonction **piste_ouverte** pour corriger ce problème.

Question A2

Ecrire l'algorithme d'une fonction ***parcours_valide*** qui prend en argument une chaine de caractères (comme parours1, parcours2 et parcours3) et renvoie en sortie un booléen VRAI si la chaine de caractères saisie correspond à un parcours valide de pistes ouvertes et FAUX dans le cas contraire.

PARTIE B :
Durée : 30 minutes

Pour cette partie, une implémentation de vos solutions algorithmiques est demandée. Pour cela vous devez utiliser un ordinateur.

Toutefois, en cas de difficulté d'implémentation, un algorithme en langage naturel sera pris en compte dans l'évaluation. Dans ce cas seulement, vous rédigerez votre algorithme sur la « feuille réponse » de la page 7.

Vous enregistrerez votre travail sur une clé USB dans un dossier à votre nom.

Pensez à sauvegarder régulièrement votre travail.

Vous trouverez sur le bureau de l'ordinateur qui vous a été désigné un dossier à votre nom dans lequel se trouve un fichier intitulé : **SKI à compléter.py**. Commencez par enregistrer ce fichier sous **la clef USB**

Le code partiel, contient :

la fonction **piste_ouverte** corrigée du problème constaté dans la partie A,

Les variables :

piste = [['A' , 'B', 'B', 'C', 'C', 'D', 'D', 'E', 'F'] , ['B', 'C', 'E', 'A', 'D', 'A', 'E', 'F', 'D']]

parcours1 = 'BEF'

parcours2 = 'ABCDFDA'

parcours3 = 'DABEFDA'

Question 1 :
Ecrire la fonction ***parcours_valide*** définie dans la partie A, qui détermine si le parcours passé en paramètre est valide ou non.

Question 2 :
Modifier le code de la fonction ***parcours_valide*** afin qu'elle affiche en cas de parcours non valide, la ou les pistes qui ne sont pas ouvertes.

Question 3 :
Implémenter un algorithme qui utilise les fonctions vues dans ce sujet et analyse un parcours saisi par un utilisateur. Ce code affichera, en fonction du parcours saisi: 'parcours valide' ou 'parcours non valide', en précisant en cas de parcours non valide, les pistes non ouvertes.

FEUILLE REPONSE DE LA PARTIE A

Question A1

1) On exécute cette fonction avec comme paramètres **debut= 'B'** et **fin ='E'** et la variable globale **pistes** de la page 2. Compléter le jeu d'essais ci-dessous.

i	pistes[0][i]	pistes[1][i]	valide
0			
1			
2			
3			
4			
5			
6			
7			
8			

2) Expliquer pourquoi la fonction retourne faux alors que la piste entre B et E est ouverte.

3) Proposer une modification de la fonction **piste_ouverte** pour corriger ce problème.

```
fonction piste_ouverte (debut : de type caractère, fin : de type caractère):
        variables globale :   pistes de type tableau
        variables locales :
                i de type entier
                valide : de type booléen
        Debut de fonction :

                pour i allant de 0 à 8 :

                        si (pistes[0][i]=debut et pistes[1][i]=fin) alors :

                                valide ← vrai

                        sinon :

                                valide ← faux

                        fin de si

                fin de pour

                retourner valide
        fin de fonction
```

Question A2

Ecrire l'algorithme d'une fonction ***parcours_valide*** qui prend en argument une chaine de caractères (comme parours1, parcours2 et parcours3) et renvoie en sortie un booléen vrai si la chaine de caractères saisie correspond à un parcours valide de pistes ouvertes.

fonction **parcours_valide** (parcours: de type chaine de caractères) :