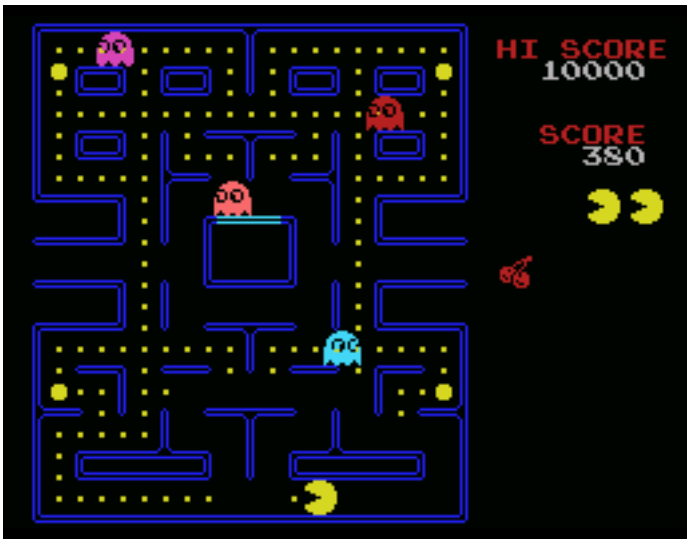
PacMan

Présentation du sujet :

PacMan est un jeu vidéo qui parut en 1980.

Le héros du jeu doit parcourir un labyrinthe et gober toutes les « pac-gommes » qui s'y trouvent, en évitant d'être touché par les fantômes.



Dans ce sujet, on propose une version très simplifiée du jeu dans laquelle il n'y a pas de fantômes.

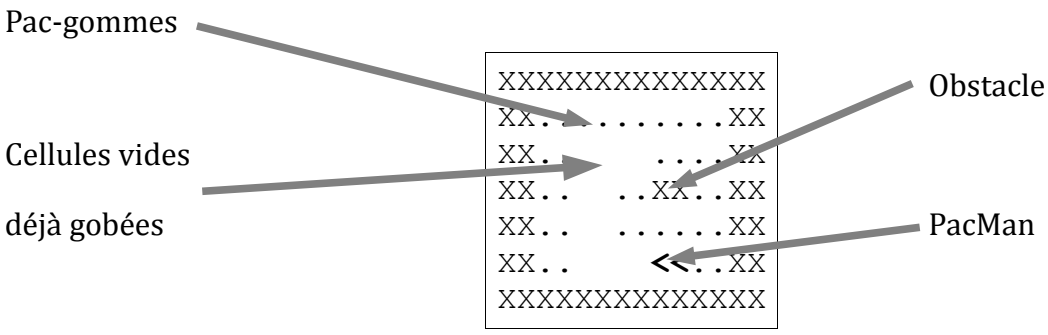
Le plateau de jeu comportera 7 lignes et 7 colonnes, donc 49 cellules.

Le but du jeu sera simplement de déplacer PacMan sur toutes les « pac-gommes », en évitant les obstacles :

La cellule du personnage PacMan sera dessinée par	<<
Une cellule contenant « pac-gomme » sera dessinée	..
Une cellule avec un obstacle sera dessinée	XX

Une cellule vide sera dessinée avec des espaces.

Dès que PacMan se déplace sur un cellule contenant « pac-gomme », il « gobe » le contenu de cette cellule. Lorsque PacMan quitte cette cellule, celle ci devient vide. On peut voir ci-dessous un exemple du jeu simplifié dans lequel PacMan a déjà gobé plusieurs « pac-gommes »



Composants fournis

Dans tout le sujet, on utilisera les variables définies dans l'encadré ci-dessous

```
VARIABLES
plateau  de type tableau à 7 lignes et 7 colonnes
k, lin, col  de type entier
```

L'initialisation de ces variables se fait en plusieurs étapes :

```
# ETAPE 0
plateau ← tableau de dimensions 7 x 7 dont chaque cellule vaut 2
# ETAPE 1
Pour k allant de 1 à 5 :
    plateau[0][k] ←  -1
    plateau[6][k] ←  -1
# ETAPE 2
Pour k allant de 1 à 6 :
    plateau[k][6] ← -1
    plateau[k][0] ← -1
# ETAPE 3
plateau [3][4] ← -1
lin ← 2
col ← 3
plateau [lin][col] = 7
# FIN
```

Dans la variable **plateau**, la valeur –1 représente un obstacle,
la valeur 2 représente « pac-gomme »
la valeur 0 représente une cellule vide
la valeur 7 représente PacMan

PARTIE A :

Utilisation d'un ordinateur interdite. Les réponses sont à rédiger sur la feuille réponse (page 5)

Durée : 30 minutes

Question A1

a) Donner, sur la feuille réponse, le contenu des **cellules** de la variable **plateau** qui sont **modifiées à l'étape 1**.

b) Donner, sur la feuille réponse, le contenu des **cellules** de la variable **plateau** qui sont **modifiées à l'étape 2**.

c) Donner, sur la feuille réponse, le **contenu de la variable plateau** après la fin de l'étape **3**.

Question A2 On définit la fonction `convert_ligne`, et la variable `tab1` par :

```
fonction  convert_ligne(tab de type tableau, i de type entier) :
variables locales :
     ligne de type chaine de caractères
     j     de type entier
     case  de type entier
début
     ligne ← ''   # chaîne de caractères de longueur nulle
     pour j allant de 0 à 6 :
          case ← tab[i][j]
          si case est égal à  2 :   # pac-gommme
               ligne ← ligne + '..'
          fin_si
          si case est égal à   -1 : # obstacle
               ligne ← ligne + 'XX'
          fin_si
          si case est égal à  0 :   # case vide
               ligne ← ligne + '  '
          fin_si
          si case est égal à  7 :   # pacman
               ligne ← ligne + '<<'
          fin_si
     retourner ligne
fin


tab1 ← [[-1, -1, -1, -1, -1, -1, -1],
        [-1,  2,  2,  2,  2,  2, -1],
        [-1,  0,  0,  0,  2,  2, -1],
        [-1,  0,  7,  2, -1,  2, -1],
        [-1,  0,  0,  2,  2,  2, -1],
        [-1,  0,  0,  2,  2,  2, -1],
        [-1, -1, -1, -1, -1, -1, -1]]
```

a) On exécute l'instruction **`convert_ligne(tab1 , 2)`**.

Sur la feuille réponse, compléter le tableau de suivi de variables

b) En déduire la valeur retournée par cet appel de fonction.

Question A3 On souhaite définir une fonction **`nb_cases(tab, n)`** qui retourne le nombre de cellule égales à la valeur de `n` dans le tableau `tab`. Par exemple :

`nb_cases( tab1 , 7 )` doit retourner 1, car une seule cellule de tab1 contient la valeur 7

`nb_cases( tab1 , -1 )` doit retourner 25, car tab1 contient 25 cellules de valeur **-1**.

Compléter, sur la feuille réponse, l'algorithme proposé pour cette fonction

PARTIE B :
Durée : 30 minutes

Pour cette partie, une implémentation de vos solutions algorithmiques est demandée. Pour cela vous travaillerez sur ordinateur. *En cas de difficulté d'implémentation, un algorithme langage naturel sera pris en compte dans l'évaluation. Dans ce cas, vous rédigerez votre algorithme sur une feuille de brouillon.*
Vous enregistrerez votre travail sur la clé USB dans un dossier à votre nom.
Ficher de travail : pacman_A_COMPLETER.py
Le fichier contient les composants fournis dans la partie A.
Vous devrez compléter ce fichier pour traiter la partie B
Pensez à sauvegarder régulièrement votre travail.

Le code fourni sur la clé contient :

- une fonction **`affichage()`** qui permet l'affichage de la variable plateau
- une fonction **`jeu()`** qui permet au joueur d'effectuer 25 déplacements du PacMan, en saisissant l'une des touches du pavé numérique (2, 4, 6, ou 8)
- une fonction **`déplacement(touche)`** qui effectue certaines modifications dans la variable **`plateau`**, lorsque la valeur du paramètre touche est 2, 4, 6, ou 8.

Question B1

La fonction **`déplacement`**(**`touche`**) permet au PacMan d'effectuer un déplacement vers le bas ou vers le haut, mais pas vers la gauche ou vers la droite. **Compléter le code aux endroits indiqué** pour traiter le cas d'un déplacement vers la gauche ou vers la droite

Question B2 : fin du jeu

quelle que soit la stratégie du joueur, le jeu termine après 25 déplacements. On souhaite desormais que le jeu cesse lorsque toutes les pac-gommes ont été gobées, c'est à dire lorsque la variable `plateau` ne contient plus aucune cellule de valeur 2.

Effectuer les modifications nécessaires dans le programme (on pourra utiliser la question A3)

Question B3 Calcul du score

On souhaite faire calculer le score du joueur de la manière suivante :

- Au départ, le score vaut zéro
- Chaque appui sur l'une des touches de déplacement coûte 1 point de score.
- Chaque fois que PacMan atteint une case contenant `..` (« pac-gomme ») le score augmente de deux points.

Ainsi,

- lorsque PacMan se rend sur une case vide, il perd 1 point.
- lorsque PacMan se déplace sur une case contenant '..' : il gagne 2 points mais il dépense 1 point pour son déplacement. Au total, le score augmente d'un point (+2 – 1)

Effectuer les modifications nécessaires pour que le programme calcule et affiche le score.

FEUILLE REPONSE DE LA PARTIE A **NOM : . . . . . .**

Question A1

a) Donner le contenu des cellules de la variable **`plateau`** qui sont **modifiées à l'étape 1**.

b) Donner le contenu des cellules de la variable **`plateau`** qui sont **modifiées à l'étape 2**.

c) Donner le **contenu de la variable `plateau`** après la fin de l'étape **3**.

Question A2

a) On exécute l'instruction convert_ligne (tab1 , 2) . Compléter le tableau de suivi de variables

`i`	`j`	`case`	`ligne`

b) En déduire la valeur retournée par cet appel de fonction.

Question A3

On souhaite définir une fonction **nb_cases(tab, n)** qui retourne le nombre de cellule égales à la valeur de **n** dans le tableau **tab** :

tab est de type tableau à deux dimensions avec 7 lignes et 7 colonnes

n est de type entier

par exemple :

nb_cases(tab1,7) doit retourner 1, car une seule cellule de tab1 contient la valeur 7

nb_cases(tab1, -1) doit retourner 25, car tab1 contient 25 cellules de valeur -1.

```
fonction nb_cases (tab de type tableau à 7 lignes et 7 colonnes,
                   n de type entier) :
variables locales :
     resu  de type entier
     i     de type entier
     j     de type entier
début
     resu ← 0
     pour i allant de 0 à ...
          pour j allant de 0 à ...
               si . . . . . est égal à n :
                    resu ← ...
               fin_si
     retourner resu
fin
```