JEU DE MÉMOIRE

Le joueur choisit successivement deux cartes en cliquant dessus. Si la 2eme carte est identique à la première, elles restent définitivement dans cette position, faces visibles.

La première choisie reste visible tant que la 2eme n'a pas été découverte.

Si la 2eme carte est différente de la 1ere, les deux cartes restent visibles environ un demi second puis se replacent automatiquement face cachée.

Code :

La 1ere partie du programme :

```
from tkinter import *
from random import shuffle
```

on importe toutes les fonctionnalités de Tkinter

on importe seulement 'shuffle ' de la bibliothèque 'random'

Partie déclaration :

```
nb_lig=4
nb_col=5
nb_carte=nb_col*nb_lig//2
dim=128 #dimension de l'image
pad=5 #panding
side=dim+pad*2 #une seul coté
x0=y0=side/2
```

nb_lig :nombre de lignes
nb_col : nombre de colonnes
nb_carte : nombre de cartes utilisées
dim :la dimension d'une seule carte
pad :le 'pading' entre le carte (l'espacement)
side : la dimension d'une seule cotée dans le tableau
X0 et Y0 : coordonnées du centre de l'image

La fonction Melange_grille() :

```
def Melange_grille():
    carte=list(range(nb_carte))*2  #[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, 3,
4, 5, 6, 7, 8, 9]

    shuffle(carte) #melanger le contenu de la carte
    p=[]
    i=0
    for k in range(nb_lig):
        l=[]
        for j in range(nb_col):
            l.append(carte[i])
            i=i+1
        p.append(l)
    return p
```

Le plateau contient 20 images qui se regroupent en 10 paires. Dans le tableau ,chaque image sera représentée par un entier entre 0 et 9, ce qui fait 10 entiers, chaque entier apparaissant deux fois.

Carte : contient les nombres qui présentent les images

Shuffle(carte) : pour changer la position des numéros d'une manière aléatoire

P : c'est le tableau qui contient les différentes numéro

La fonction creation_logos() :

```
liste=['achiv','choy','daifuku','gas','kiwi','popsicle','python','sound','thumb','truck']
def creation_logos():
    icones=[]
    for lang in liste:
        fichier="./image/"+lang+".png"
        icon=PhotoImage(file=fichier)
        icones.append(icon)
    return icones
```

Cette fonction crée une liste d'objet (la variable 'icones') en utilisant 'PhotoImage ' . En effet elle boucle dans la liste 'liste' et fait la construction du chemin du fichier image

La fonction remplissage(plat) :

```
def remplissage(plat):
    nb_l=4
    nb_c=5
    ids_imgs=[]
    for ligne in range(nb_l):
        id=[]
        for col in range(nb_c):
            centre=(col*side+x0,ligne*side+y0)
            nr=plat[ligne][col]
            icon=icones[nr]
            cnv.create_image(centre,image=icon)
            id_im= cnv.create_image(centre,image=logo)
            id.append(id_im)
        ids_imgs.append(id)
    return ids_imgs
```

Cette fonction nous permet de remplir le tableau avec les images à partir du plat (plat =Melange_grille()) :

1. Elle récupère le numéro de l'image (variable nr) à partir du (plat)
2. Elle récupère l'image correspondante à 'nr' à partir de la liste des images 'icones '
3. Elle ajoute l'image récupérée à la canvas (cnv)
4. Elle retourne la liste d'identifiants d'images (ids_imgs)

La fonction clic() :

```
def clic(event):
    if move[1] is not None:
        return
    x=event.x
    y=event.y
    col=x//side
    lig=y//side
    if (plat[lig][col]!=-1):
        traite_clic(lig,col)
```

La fonction clic(event) semble gérer les événements de clic de souris dans la canvas (cnv). Comme les cartes sont régulièrement espacées d'une distance SIDE, en supposant que chaque ligne et chaque colonne de cartes soit numérotée à partir de 0, le numéro de la colonne 'col' vaut le quotient entier x par 'side' et la ligne lig vaut le quotient entier y par 'side'.

Si le numéro de l'image est diffèrent de -1 on va passer à la fonction traite_clic(lig,col)

La fonction traite_clic (lig,col) :

```
def traite_clic(lig,col):
    global cpt
    item=ids_imgs[lig][col]
    cnv.delete(item)
    if move[0] is None:
        move[0]=(lig,col)
    else:
        if move[0]==(lig,col):       #meme clic
            return
        cpt=cpt-1           #traitement du compteur
        lbl['text']=cpt     # --
        move[1]=(lig,col)
        i,j=move[0]
        if plat[i][j]==plat[lig][col]:
             plat[i][j]=plat[lig][col]=-1
             move[0]=move[1]=None

        else:
            cnv.after(400,cacher,i,j,lig,col)
```

La fonction doit gérer les paires de clics et lancer l'animation si le joueur découvre des cartes différentes. Donc elle doit tester toutes les clics (clic sur la même carte – clic sur une carte déjà retournée – clic sur une carte identique -clic sur une carte différente)

Aussi elle permet de calculer le score

La fonction cacher () :

```
def cacher(i,j,lig,col):
    centre=(j*side+x0,side*i+y0)
    ids_imgs[i][j]= cnv.create_image(centre,image=logo)
    centre=(col*side+x0,lig*side+y0)
    ids_imgs[lig][col]= cnv.create_image(centre,image=logo)
    move[0]=move[1]=None
```

Cette fonction permet de cacher les 2 cartes si elles sont différentes et retourne l'image initiale

La fonction init () :

```
def init():
    global plat,ids_imgs,move,icones,logo ,cpt,lbl
    icones= creation_logos()
    logo=PhotoImage(file="flag.png")
    plat=Melange_grille()
    ids_imgs=remplissage(plat)
    move=[None,None]
    cpt=500
    lbl['text']=500
```

C’est la fonction initiale qui renforme toutes les autres fonctions

Jlassi Sahar