Sokoban 3

Dans la partie précédente, nous avons mis en place le jeu pour un niveau donné.

Il ne nous reste plus qu'à permettre de jouer plusieurs niveaux à la suite, si le fichier de description des niveaux, comporte la description de plusieurs niveaux.

Nouvelles commandes pour un niveau donné

Pour un niveau donné, nous rajoutons les commandes :

- P = changement de personnage
- Retour arrière = relancer le niveau actuel à son état initial : bien pratique lorsque l'on ne peut plus finir le niveau.
- S = passer au niveau suivant
- R = revenir au niveau précédent

Nous repartons de la version 2 de Sokoban et nous ajoutons dans la partie traitant les touches :

```
#Si le joueur a appuyé sur la touche p
elif event.key == pygame.K_p:
    # Change l'image du joueur en passant à la suivante
    joueurActuel += 1
    if joueurActuel >= len(ImagesJoueur):
        # Après la dernière image, on revient à la première
        joueurActuel = 0
    redessinerCarte = True
```

Pour les autres touches, nous devons déterminer quel est le niveau à mettre en place puis afficher ce niveau.

Nous créons une fonction chargée de mettre en place le niveau. On lui passe en parametre l'ensemble des niveaux et l'index de celui qu'il faut préparer :

```
def preparerNiveau(niveaux, indexNiveauEnCours):
    global carte, niveau, jeuEtat, niveauTermine, redessinerCarte
    niveau = niveaux[indexNiveauEnCours]
    carte = modifierCarte(niveau['carte'], niveau['etatInitial']['joueur'])
```

```
    jeuEtat = copy.deepcopy(niveau['etatInitial'])
    niveauTermine = False
    redessinerCarte = True
```

Nous pouvons utiliser cette fonction pour traiter les touches agissant sur les niveaux :

```
            #Si le joueur a appuyé sur la touche "retour arrière"
            #On reprend le même niveau à l'état initial
            elif event.key == pygame.K_BACKSPACE:
                preparerNiveau(niveaux, indexNiveauEnCours)
            #Si le joueur a appuyé sur la touche "s"
            #On passe au niveau suivant
            elif event.key == pygame.K_s:
                indexNiveauEnCours = indexNiveauEnCours + 1
                if indexNiveauEnCours >= len(niveaux):
                    # Si on a épuisé tous les niveaux, on revient au premier
                    indexNiveauEnCours = 0
                preparerNiveau(niveaux, indexNiveauEnCours)
            #Si le joueur a appuyé sur la touche "r"
            #On revient un niveau en arrière
            elif event.key == pygame.K_r:
                indexNiveauEnCours = indexNiveauEnCours - 1
                if indexNiveauEnCours < 0:
                    # Si on a dépassé le niveau 0, on passe au dernier
                    indexNiveauEnCours =len(niveaux)-1
                preparerNiveau(niveaux, indexNiveauEnCours)
```

Niveau terminé

Dans la version précédente, lorsque le niveau était terminé, tout s'arrêtait.

Il faut maintenant afficher le panneau résolu, et passer au niveau suivant lorsque le joueur appuie sur une touche.

Nous créons une fonction chargée d'afficher cet écran et d'attendre que le joueur appuie sur une touche.

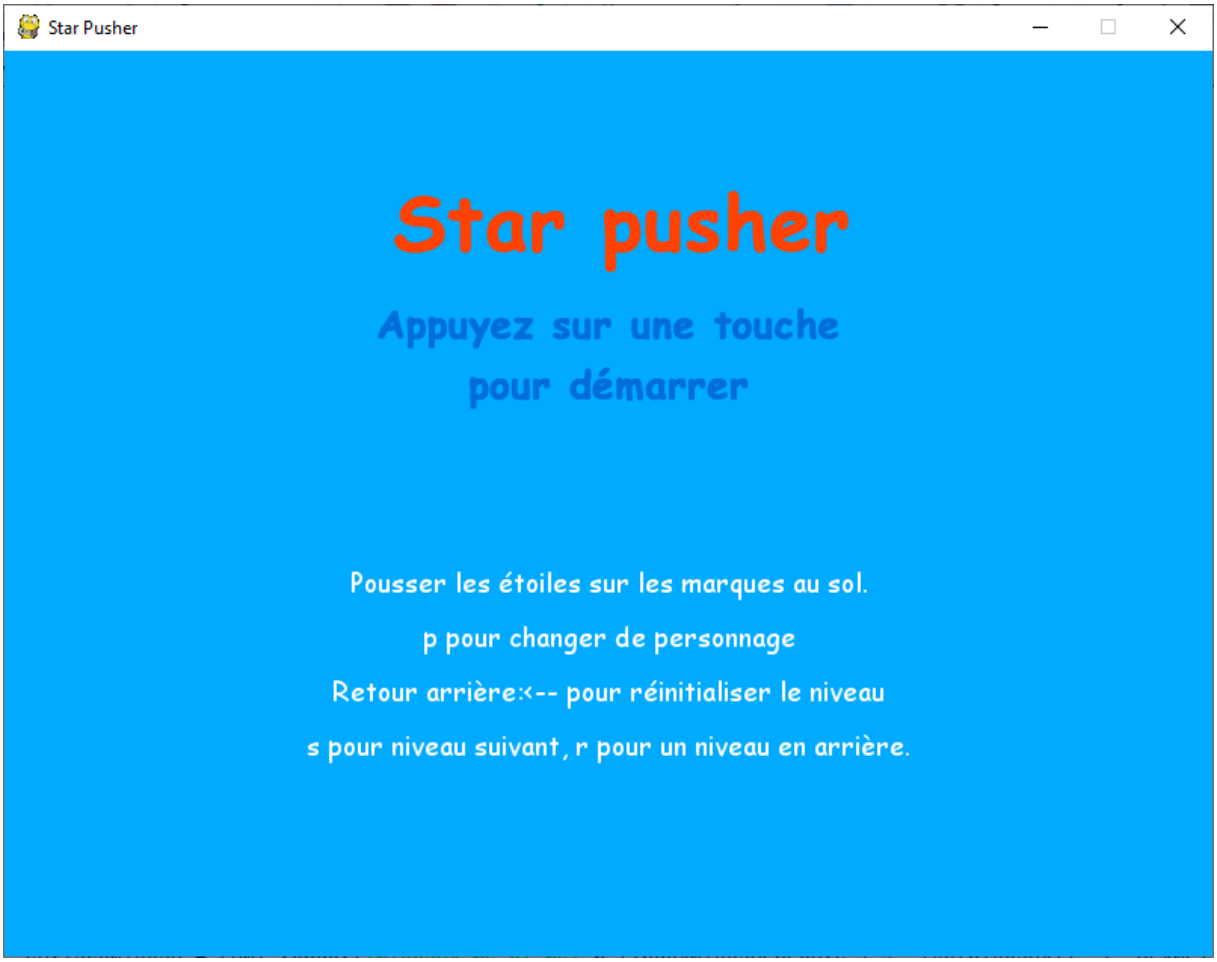
```
def ecranResolu():
    global indexNiveauEnCours
    # Si le niveau est résolu, afficher image "résolu" et
    #attendre que le joueur appuie sur une touche
    resoluRect = DicoImage['resolu'].get_rect()
    resoluRect.center = (MILIEU_LARGEUR_ECRAN, MILIEU_HAUTEUR_ECRAN)
    ecran.blit(DicoImage['resolu'], resoluRect)
    pygame.display.update()
    while True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
```

```
        elif event.type == pygame.KEYDOWN:
            #on passe au niveau suivant lorsque le joueur appuie sur une touche
            indexNiveauEnCours = indexNiveauEnCours + 1
            if indexNiveauEnCours >= len(niveaux):
                # Si on a épuisé tous les niveaux, on revient au premier
                indexNiveauEnCours = 0
            preparerNiveau(niveaux, indexNiveauEnCours)
            return
    FPSCLOCK.tick()
```

A la fin de la boucle de jeu, il suffit d'écrire :

```
    #-----------------------------------------------------
    #Si le niveau est terminé, on affiche l'image "resolu"
    if niveauTermine:
        ecranResolu()
```

Ecran de démarrage




```
def Instructions():
    """Affiche l'écran de départ avec les instructions."""
    # Positionne l'image "titre"
    titreRect = DicoImage['titre'].get_rect()
    topCoord = 50 # topCoord tracks where to position the top of the text
    titreRect.top = topCoord
    titreRect.centerx = MILIEU_LARGEUR_ECRAN
    topCoord += titreRect.height
    instructionText = ['Pousser les étoiles sur les marques au sol.',
                       'p pour changer de personnage',
                       'Retour arrière:<-- pour réinitialiser le niveau',
                       's pour niveau suivant, r pour un niveau en arrière.']
```

```
    ecran.fill(BLEUCLAIR)
    # Affiche l'image titre
    ecran.blit(DicoImage['titre'], titreRect)
    # Positionne et affiche le texte
    for i in range(len(instructionText)):
        instSurf = FONT.render(instructionText[i], 1, BLANC)
        instRect = instSurf.get_rect()
        topCoord += 10 # 10 pixels entre chaque ligne de texte
        instRect.top = topCoord
        instRect.centerx = MILIEU_LARGEUR_ECRAN
        topCoord += instRect.height
        ecran.blit(instSurf, instRect)
    pygame.display.update()
    #On attend que le joueur appuie sur une touche
    while True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
            elif event.type == pygame.KEYDOWN:
                return
        FPSCLOCK.tick()
```

Avant le début de la boucle de jeu

```
#---------------Boucle de jeu----------------------------------------
Instructions()
niveaux = lectureFichierNiveaux('Microban.txt')
indexNiveauEnCours = 0
preparerNiveau(niveaux, indexNiveauEnCours)
```

Dans la boucle de jeu, lors de l'affichage de la scène, nous ajoutons l'affichage du numéro du niveau joué et du nombre de pas fait par le joueur.

```
        #------------------------------------------------------
        #on met l'écran à jour
        ecran.fill(BLEUCLAIR)
        #On ne redessine la carte que si des modifications ont été apportées
        if redessinerCarte:
            surfaceCarte = dessinerScene(carte,jeuEtat,niveau['objectifs'] )
            redessinerCarte = False
        #afficher la scène, et les textes en bas d'écran
        surfaceCarteRect = surfaceCarte.get_rect()
        surfaceCarteRect.center = (MILIEU_LARGEUR_ECRAN,
MILIEU_HAUTEUR_ECRAN)
        ecran.blit(surfaceCarte,surfaceCarteRect)
        surfaceNiveau = FONT.render('Niveaux %s of %s' % (indexNiveauEnCours + 1,
len(niveaux)), 1, BLANC)
        surfaceNiveauRect = surfaceNiveau.get_rect()
        surfaceNiveauRect.bottomleft = (20, HAUTEUR_ECRAN - 35)
        ecran.blit(surfaceNiveau, surfaceNiveauRect)
        surfacePas = FONT.render('Pas: %s' % (jeuEtat['compteur']), 1, BLANC)
        surfacePasRect = surfacePas.get_rect()
        surfacePasRect.bottomleft = (20, HAUTEUR_ECRAN - 10)
        ecran.blit(surfacePas, surfacePasRect)
         pygame.display.update()
```