Membres de la réalisation du projet :
Noman Pajoul
Guy-charles Gasso
Kylian Ponsinet

COMPTE RENDU DU PROJET D'ISN

The race to the Beyond

(« la course vers l'au-delà »)

SOMMAIRE

I. Présentation du projet.....**page 3 à 4**
- a) L'enjeu et problématique, pourquoi le choix de ce projet.
- b) Le principe du type de jeu choisi et historique
- c) Les objectifs et mécaniques de jeu

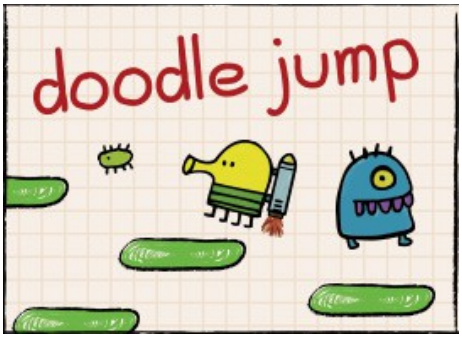
II. Concrétisation du projet/ application des idées.....**page 4 à 5**
- a) Graphismes du jeu
- b) Répartition des tâches de travail

III. Synthèse personnelle.....**page 5 à 6**
- a) Etapes détaillées et difficultés rencontrées
- b) Perspective et évolution du projet
- c) Bilan personnel

IV. ANNEXES.....**page 6 à 7**

LE PROJET :

D'abord lors de l'élaboration de notre projet, on était sûre de faire un jeu de plates-formes plutôt qu'un jeu de hasard, simulation, etc....
Au début on voulais faire un jeu similaire à « doddle jump » car c'est un jeu rapide que beaucoup de personnes connait et apprécie, mais on a vit compris que cela allait être compliqué à finir dans le temps impartie. On a donc décidé de faire un jeu de plateforme simple avec un personnage qui peut se déplacer, sauter et interagir avec des objets.



Les jeux de plates-formes sont apparus au début des années 1980, tout d'abord en 2D, intégrant souvent des systèmes d'échelles à grimper. Le genre a connu un nouvel essor dans les années 1990 avec l'apparition des premiers jeux de plates-formes en 3D.
En générale, dans les jeux de plates-formes, le joueur contrôle un avatar/personnage qui doit sauter sur des plates-formes suspendues dans les air et éviter des obstacles. Les environnements requièrent de devoir sauter ou grimper pour pouvoir être traversés.

Jeu Donkey kong des années 80

Objectifs :

Le but de notre jeu est de récupérer un bonus à un emplacement variable dans le temps impartie, c'est-à-dire que lorsque le joueur aura récupéré le bonus, le bonus changera d'emplacement et le joueur gagnera un bonus de temps, il doit à nouveau récupérer le bonus à son nouvel emplacement, il peut y avoir des pièges. Donc le jeu se termine lorsque le temps est écoulé.

Le joueur contrôle uniquement le personnage à l'aide des touches directionnelles de l'ordinateur, pour jouer à notre jeu.

Les objets dans le jeu sont des images modifiées, à l'aide de « Paint ».



Organisation/Répartition du travail :

Au début du projet il n'y avait pas vraiment de répartition du travail chacun apportait ses idées et/ou codage, ensuite on a listé les tâches à accomplir pour que le jeu soit complet, puis chacun s'est attribué certaines tâches de travail.

Tâches	Guy-charles	Kylian	Noman
Déplacement personnage	**X**	**X**	**X**
Gestion des bonus	**X**		
Gestion du compte à rebours	**X**		
Définition des fonctions	**X**	**X**	
Modification des images inclues dans le jeu			**X**
Menu rapide		**X**	**X**

Durant le projet chacun travaillait de son côté, puis on faisait une mise au point (mise en commun du travail effectué) aux heures de cours, mais on s'est aussi vu en dehors des cours afin d'avancer le projet. On a d'avantage travaillé en classe puisque on était en groupe et le professeur pouvait directement nous assistés lorsqu'on rencontrait des problèmes de codage.

Au début du programme j'ai importé le module/option "random" afin d'utiliser une fonction de Python, la fonction aléatoire "La fonction randrange( ) »

```
import pygame
import random
```

Elle permet de générer un nombre pseudo-aléatoire entre la plage de valeurs donnée .
Par exemple, vous voulez générer un nombre aléatoire compris entre 10 et 50, puis vous pouvez utiliser cette fonction.

```
import random


print("affichage du nombre aléatoire entre 10 et 50 == ", random.randrange (10, 50))
```

L' argument de départ est le numéro de départ dans une plage. c'est-à-dire limite inférieure. Par défaut, commence par 0 si non spécifié. Dans cet exemple l'argument de départ est 10.

L' argument d' arrêt est le dernier numéro d'une plage. L'argument d'arrêt est la limite supérieure, dans notre cas (l'exemple) il est égale à 50.

Le pas est une différence entre chaque numéro de la séquence. L'étape est des paramètres facultatifs. La valeur par défaut de step est 1 si non spécifié, donc si dans cet exemple on avait un step de 10, le résultat sera forcément une dizaine entre 10 et 50.
Il peut y avoir une erreur générée si on utilise des valeurs autres qu'un entier. En d'autres termes, on ne peut pas utiliser de valeurs flottantes. Une erreur peut aussi être générée si la valeur limite est inférieure ou égale au numéro de départ ou si le saut est nulle.

```
*** Console de processus distant Réinitialisée ***
>>>
affichage du nombre aléatoire entre 10 et 50 == 22
>>>
```

Dans le cadre du projet j'ai utilisé cette fonction pour modifier les coordonnées des bonus de façon aléatoire dans la fenêtre de jeu, une fois que l'utilisateur a récupéré les bonus.

Pour le chronomètre, j'avais besoin du module/option time qui était importé au début du programme, puis j'ai simplement pris une variable « secondes » qui exprimait le temps en millisecondes avec une des fonctions du module/option time.

```
# gestion du temps
secondes = clock.tick()/1000.0 #écoulement du temps en millisecondes
temps = 60
```

Puis une seconde variable « temps » qui était une valeur(entier) au choix que je soustrait par la variable précédente.De plus cette opération est affiché en direct sur l'écran de jeu.

```
secondes = clock.tick()/1000.0
temps -= (secondes)
```

Lorsque le temps est nulle un message de fin de jeu s'affiche vers le centre de l'écran, avec l'affichage du score final de l'utilisateur et un petit compte à rebours de 5 secondes avant la fermeture de la fenêtre de jeu.

(en dessous du score le temps restant avant la fermeture de la fenêtre est affiché)

Durant le projet, j'ai rencontrées pas mal de difficultés notamment pour appliquer certaines idées dans le programme puisque il était difficile de les traduire en langage « Python », par exemple les collisions entre le personnage et les plates-formes.

De plus certaines parties du code ne fonctionnent pas comme prévu, donc j'ai du trouvé des solutions alternatives comme les conditions de changement de coordonnées des bonus qui n'ont pas marchée, voir oublier l'idée comme l'idée de l'animation du personnage lors de son mouvement en changeant d'image pendant son mouvement.

Perspectives :

Malheureusement, nous avions pas eu le temps de finaliser notre projet dans les temps à cause des différents problèmes rencontrés. Si on aurait eu plus de temps à notre disposition, le jeu aurait été plus abouti qu'actuellement, par exemple on aurait pu faire l'animation du personnage lors de son déplacement ou créer un saut correct du personnage.

Ce que je proposerai pour améliorer le projet : Au niveau du jeu en lui-même ,on peut encore y apporter des modifications. Comme faire un menu plus complet qu'un simple menu de lancement de jeu, avec une partie indiquant la liste des commandes que l'utilisateur à besoin pour jouer, le fonctionnent des collisions entre le personnage et les blocs (rectangles dessinés à l'aide d'un outil de Pygame), voir créer plusieurs niveaux de jeu.

En travaillant sur ce projet j'ai pu utilisé les connaissances acquises en cours d'ISN à propos du langage python ainsi qu'en acquérant de nouvelles connaissances lors de l'utilisation des fonctions de « Pygame » sur le logiciel « Spyder ». Je n'avais pas d'expérience en programmation avant et ce fut très intéressant de coder à l'aide du langage Python. Je sais également comment traduire virtuellement un programme simple et cette expérience nous a aidées à améliorer nos performances de travaux en groupe.

Sites utilisées :

pygame.org

ANNEXES

```
elif keys[pygame.K_LEFT] and 0 <perso_x:
    perso_x += -k*v
elif keys[pygame.K_RIGHT] and perso_x<longueur-(k*5) :
    perso_x += k*v
elif keys[pygame.K_DOWN] and perso_y<601-75:
    perso_y += k*v
elif keys[pygame.K_UP] and perso_y>0:
   perso_y += -k*v
```

partie du programme correspondant au codage du déplacement du personnage contrôlé par l'utilisateur.

```
for event in pygame.event.get():

        if event.type == pygame.QUIT:
            sortieJeu = True
            time.sleep(1)
```

Partie du programme correspondant au début de la boucle principale dédiée à la fermeture de la fenêtre de jeu.

```
if perso_x == Bonus1_x and perso_y <= Bonus1_y: # idéalement mettre
    A = (255,131,250)
    Bonus1_x = round(random.randrange(200, (longueur-200)-perso_x))
    #Bonus1_y = round(random.randrange(100, (largeur-100)- perso_y))
    temps -= temps/2
    # fonction de placement aléatoire du bonus sous certaines conditi
elif perso_x == Bonus2_x and perso_y <= Bonus2_y: # idéalement mettr
    Bonus2_x = round(random.randrange(0, (longueur-200)-perso_x))
    Bonus2_y = round(random.randrange(0, (largeur-200)- perso_y))
    score += 1
    temps += 1
```

Partie du programme dédiée au changement aléatoire des coordonnées des deux bonus une fois récupéré par l'utilisateur