**1 Principes des solutions techniques**

**1.1 Langage**

Le codage est réalisé avec langage python. Nous choisissons la version 3.10.

**1.2 Architecture du logiciel**

Nous mettons en œuvre le principe de la barrière d'abstraction. Chaque module correspond à un type de donnée et fournit toutes les opérations permettant de le manipuler de manière abstraite.

**1.3 Interface utilisateur**

L'interface utilisateur se fera via un terminal de type linux. Nous reprenons la solution donnée en cours de IPI en utilisant les modules : termios, sys, select.

**1.3.1 Boucle de simulation**

Le programme mettra en œuvre une boucle de simulation qui gèrera l'affichage et les événements clavier.

**1.3.2 Affichage**

L'affichage se fait en communicant directement avec le terminal en envoyant des chaînes de caractères sur la sortie standard de l'application.

**1.3.3 Gestion du clavier**

L'entrée standard est utilisée pour détecter les actions de l'utilisateur. Le module tty permet de rediriger les événements clavier sur l'entrée standard. Pour connaître les actions de l'utilisateur il suffit de lire l'entrée standard.

**2 Analyse**

**2.1 Analyse noms/verbes :**

• Verbes :

Jouer, afficher, déplacer, finir, quitter

• Nom :

Ball

**2.2 Types de donnée**

Type : Ball = dictionnaire

x, y : entier

vx : réel

Type : Arriere = dictionnaire

Grid : liste de liste de caractère

**2.3 Dépendance entre modules**

**2.4 Analyse descendante :**

**2.4.1 Arbre principal :**

Main.main() +-- Main.init()

| +-- Main.Arriere.create ()

| +-- Main.Ball.create()

+-- Main.run()

+-- Main.interact()

+-- Main.live()

+-- Main.show()

**2.4.2 Arbre affichage**

Main.show()

+-- Arriere.show()

**+**--Ball.show()

**2.4.3 Arbre interaction**

Main.interact()

| +-- Main.isData()

| +--Main.quitGame

| +--Ball.move\_up()

| +--Ball.move\_down()

**2.4.4 Arbre live**

Main.live()

+-- Ball.live()

+-- Ball.live1()

**3 Description des fonctions**

**3.1 Programme Principal : main.py**

• Main.**main()**

• Main.**init()**

• Main.**run()**

• Main.**show()**

• Main.**interact()**

• Main.**live()**

+Ball1.live()

• Main.**isData()**

• Main.**quitGame()**

Main.**main()** -> rien

Description : fonction principale du jeu

Parametres : aucun

Valeur de retour : aucune

Main.**init()** -> rien

Description : initialisation du jeu

Paramètres : biblio

Valeur de retour : aucune

Main.**run ()** -> rien

Description : boucle de simulation

Paramètres : biblio

Valeur de retour : aucune

Main.**show()** -> rien

Description : fonction principale du jeu

Paramètres : biblio

Valeur de retour : aucune

Main.**interact()** -> rien

Description : gère les évènements clavier

Paramètres : biblio

Valeur de retour : aucune

Main.**live()** -> rien

Description : fait bouger les éléments dans le terminal

Paramètres : biblio

Valeur de retour : aucune

Main.**IsData() ->** rien

Description : récupère les évènements du clavier

Paramètre : aucune

Valeur de retour : aucune

Main.**quitGame()** -> rien

Description : termine la partie, affiche le visuel de sortie en

Fonction du résultat

Paramètres : biblio

Valeur de retour : aucune

**3.2 Ball.py**

* Ball.**create(x,y,vx)**
* Ball.**get\_x(ball)**
* Ball.**set\_x(ball,x)**
* Ball.**get\_y(ball)**
* Ball.**set\_y(ball,y)**
* Ball.**get\_vx(ball)**
* Ball.**set\_vx(ball,b)**
* Ball.**show(a)**
* Ball.**live(a,dt)**

Ball.**create(x,y,vx) ->** ball

Description: crée une balle

Parametres:

X, y, vx : entier

Valeur de retour : une nouvelle balle

Ball.**show(a) -> rien**

Description: affiche le visuel de a partie en cours

Paramètres:

a : balle

Ball.**live(a) -> rien**

Description: déplace la balle dans le terminal

Paramétres:

a : balle

**3.4 Arriere.py**

Arriere. **create(file)**

Arriere. **show(img)**

Arriere. **get\_width(img)**

Arriere. **get\_height(img)**

Arriere. **set\_height(img,b)**

Arriere. **get\_grid(img)**

Arriere. **set\_grid(img,b)**

Arriere.g**et\_char(img,line\_index,column\_index)**

Arriere. **set\_char(img,line\_index,column\_index,c)**

Arriere. **show(img)**

Arriere. **create(file)**

Description : crée à partir d’un fichier txt une image

Parametres :

File : fichier txt

Valeur de retour : un image

Arriere. **show(img) -> rien**

Description : affiche l’image crée en une liste de liste de caractere

Parametres : image