PROJET

Du

LYCEE FRANCOIS RABELAIS DE CHINON

BACCALAUREAT SCIENTIFIQUE
SPECIALITE INFORMATIQUE SCIENCES DU NUMERIQUE

Présenté par

Prénom NOM

Thème. : ISN
Support : Python

Logo Python

Soutenu le 24/05/2017

Encadré par :
Mr GUIBERT : Professeur de SII
Mr LAMBERT : Professeur de Mathématiques

TABLE DES MATIERES

TABLE DES ILLUSTRATIONS

1. METHODOLOGIE

1.1. Recherche du projet (carte mentale)

Voici la carte mentale que j'avais effectuée en premier lieu de mon côté.

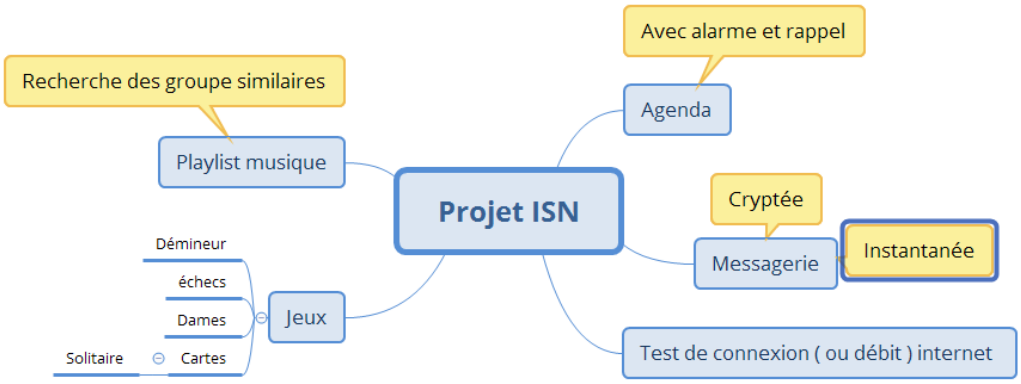


Figure 1 : carte mentale

Finalement aucune des mes idées n'a été retenue. J'ai donc participé à un projet proposé par un des mes camarades qui est le jeu sur plateau.

1.2. Gestion du projet

1.2.1. GANTT

Voici le planning créer au début du projet.

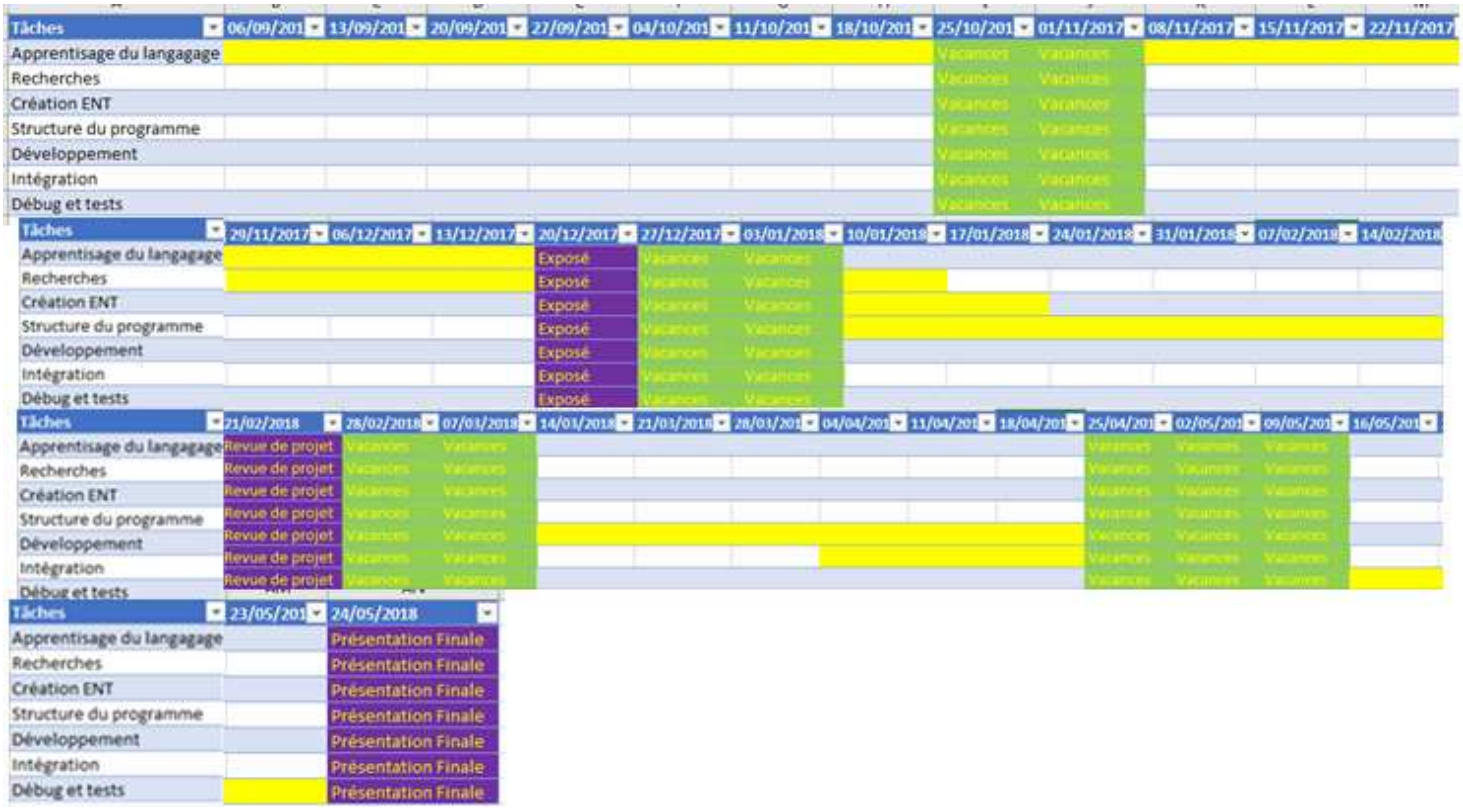


Figure 2 : GANTT

1.2.2. Espace numérique de travail

	Logiciel
Gantt	Excel
SYSML	Draw.io
Carte mentale	Xmind
Stockage en ligne	Google drive + email groupe + discord

Tableau 1 : ENT

1.2.3. Composition du groupe

Exemple qui suit comporte de nombreuses fonctions qui ne sont pas abordées dans votre projet.

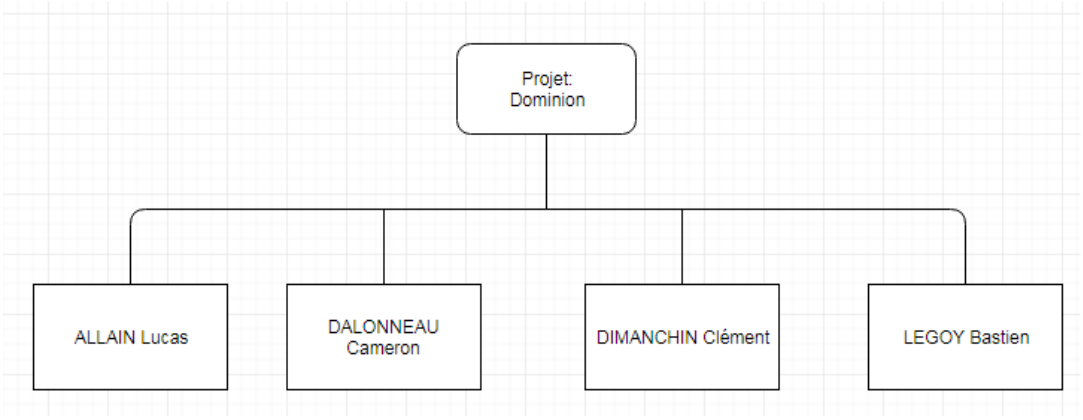


Figure 3 : Organigramme

1.3. Support et librairies

J'ai tout codé avec python sur IDLE, car je me suis habitué à l'interface, mais pour la vérification finale j'ai utilisé Spyder qui propose la numérotation de ligne et l'explorateur de variables. Je n'ai utilisé que la librairie « random ».

1.4. Mes objectifs

Mes objectifs étaient de générer des objets à intervalle de tours réguliers dans la limite de l'inventaire voulu. Puis quand le joueur ou le bot utilisent un objet, de vérifier leur capacité à pouvoir utiliser l'objet, de supprimer l'objet de l'inventaire et de renvoyer les caractéristiques de l'objet au plateau et au main.

2. ANALYSE

Voici l'algorigramme du projet. Il se décompose en 5 fonctions.

1. Le programme principal (main ()) qui gère toutes les autres fonctions.
   Réalisateur : Bastien
2. La fonction Bot () qui gère les mouvements de L'IA.
   Réalisateur : Bastien
3. La fonction Objet () qui gère la génération et l'utilisation de bonus dits « objets ».
   Réalisateur : Moi-même
4. La fonction Plateau () qui gère les modifications opérées par les joueurs sur le plateau.
   Réalisateur : Cameron
5. La fonction GUI () qui gère l'affichage graphique du programme.
   Réalisateur : Clément

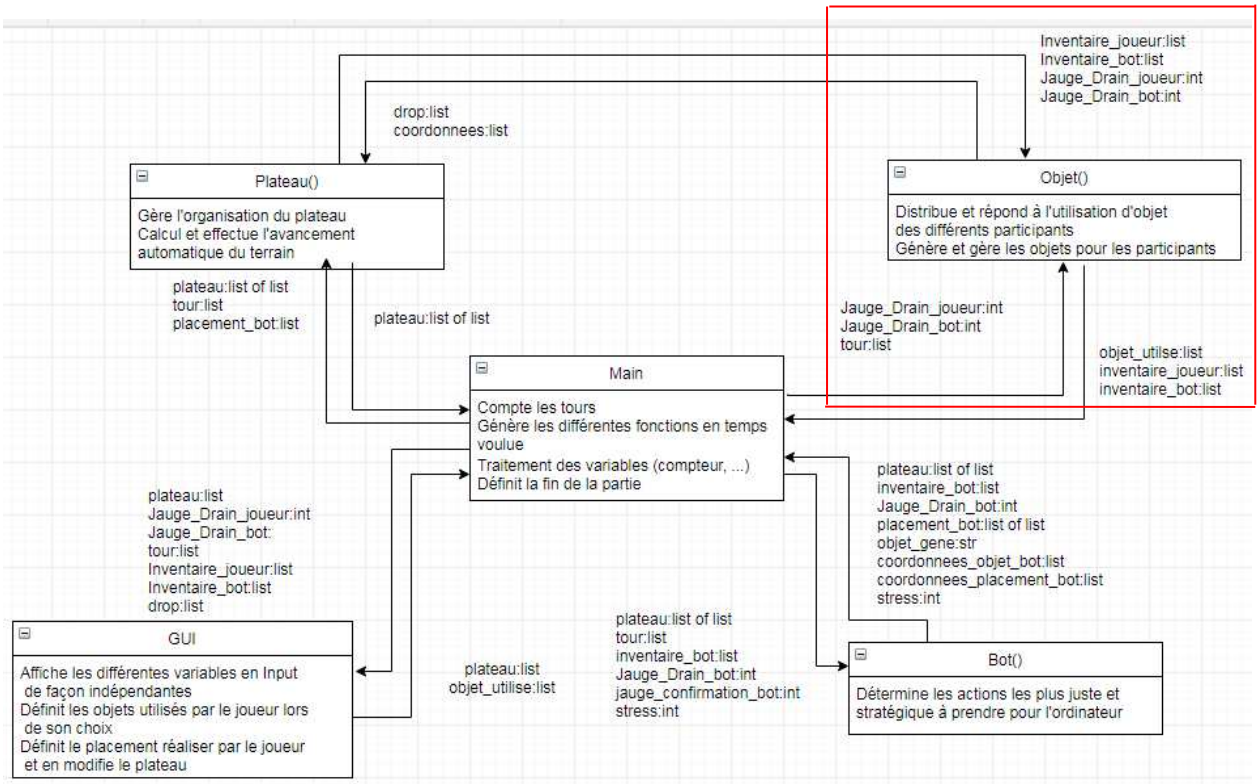


Figure 4 : Découpage fonctionnel

3. TABLEAU DE VARIABLES

Voici le tableau récapitulatif des variables utilisées dans le programme.

Variables globales	Type
tour	List
inventaire_joueur	List
inventaire_bot	List
drop	List
coordonnees	List
objet_utilise	List
Jauge-Drain_joueur	Int
Jauge_Drain_bot	Int
numero_objet	Int
objet_gene	Str
Variables locales	**Type**
caracteristique_BOM	List
caracteristique_REE	List
caracteristique_DRL	List
caracteristique_ADC	List
caracteristique_CAS	List
nombre_objets_generes	Int
droit	Int
proba_classe	Int
proba_objet	Int
numero_objet_bot	Int
objet_genere_aleatoire	Str

Tableau 2 : Variables totales

4. ALGORITHME DU PROGRAMME PRINCIPAL

Voici l'algorithme de mon programme principal.

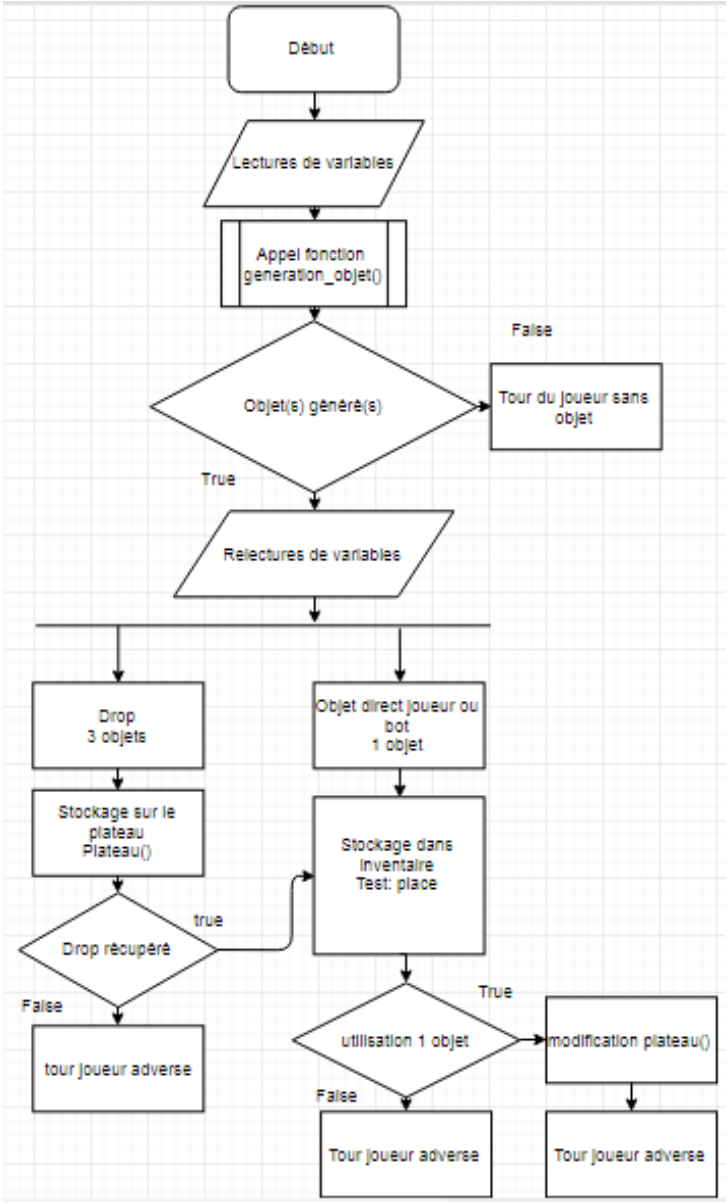


Figure 5 : Algorithme principal

5. CONSTRUCTION DU PROGRAMME

Premièrement les objets apparaissent tous les 3 ou 10 tours, ils dépendent ainsi de la variable tour.

```
if tour[0]%10 == 0 and tour[0] != 0 and tour[1] != 2:
elif tour[0]%3 == 0 and tour[0] != 0 and tour[1] == 1:
elif tour[0]%3 == 0 and tour[0] != 0 and tour[1] == 2:
```

Vérification pour le drop.
Vérification pour le Joueur.
Verification pour le bot.

Puis il génère le nombre d'objets voulus à chaque utilisateur, dans la limite de place de leur stockage.

```
for j in range(nombre_objet_generes):
if len(drop) == 0:
drop.append(choix_objet())

if len(inventaire_joueur) < 5:
inventaire_joueur.append(choix_objet())

if len(inventaire_bot) < 5:
inventaire_bot.append(choix_objet())
```

Il génère les objets selon une probabilité d'obtenir un nombre entre 1 et 100 avec randint.

```
proba_classe = randint(1,100)

if proba_classe <= 50:# On choisit un un objet dans la classe 1
proba_objet = randint(1,2)
if proba_objet == 1:

elif 50 < proba_classe <= 85:# On choisit un un objet dans la classe 2
proba_objet = randint(1,2)
if proba_objet == 1:

else:# On choisit un un objet dans la classe 3
```

Lorsque les utilisateurs utilisent un objet, ils envoient une information, un srt pour le bot et un int pour le joueur. On vérifie aussi leur cappacité à utiliser l'objet. On appelle ensuite la fonction objet_utilise

```
    numero_objet = int(input('numero_objet = '))
    numero_objet = numero_objet - 1 # Pour correspondre à l'indice de la liste
    objet_utilise = objet_choisit(tour, numero_objet, coordonnees)
    if Jauge_Drain_joueur < objet_utilise[1]:

    objet_gene = input('objet_gene = ') # On le transforme en int pour la suite
    objet_utilise = objet_choisit(tour, objet_gene, coordonnees)
    if Jauge_Drain_bot < objet_utilise[1]:
```

On vérifie l'objet utilise, par exmple la bombe pour le joueur.

```
if tour[1] == 1:
if inventaire_joueur[numero_objet] == 'BOM':
return bombes(coordonnees)
```

Pour la bombe, la fonction return les caractéristiques suivantes.

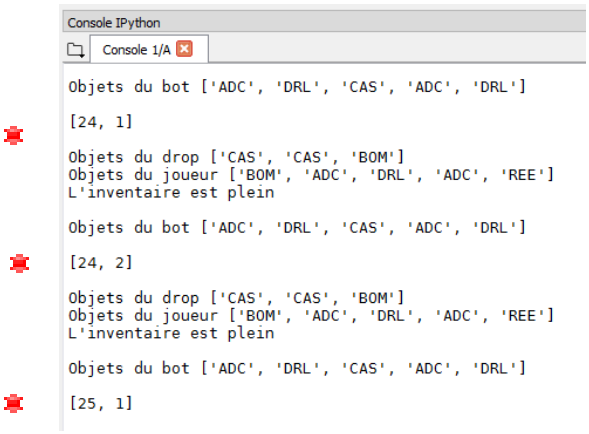
```
x = randint(1, 16)
y = randint(1, 16)
global coordonnees
coordonnees = [x, y]

def bombes(coordonnees):
caracteristique_BOM =[1, 20]
caracteristique_BOM = caracteristique_BOM + coordonnees
return caracteristique_BOM
```

Enfin on suprimme l'objet utilisé.

```
del inventaire_joueur[numero_objet]
```

6. FONCTIONNEMENT PROGRAMME



Ici le programme à générer 25 tours. Le tour est une liste de 2 éléments, le premier est un compteur et le deuxième désigne s'il s'agit du joueur (1) ou du bot (2).

* : variable tour

On constate qu'il a bien généré 5 objets maximum pour le bot est le joueur et 3 pour le drop.

Il indique également que l'inventaire du joueur est plein.

Il génère des objets au joueur et au bot tours les 3 tours. Et au drop tous les 10 tours qu'il renouvelle s'il n'est pas pris.

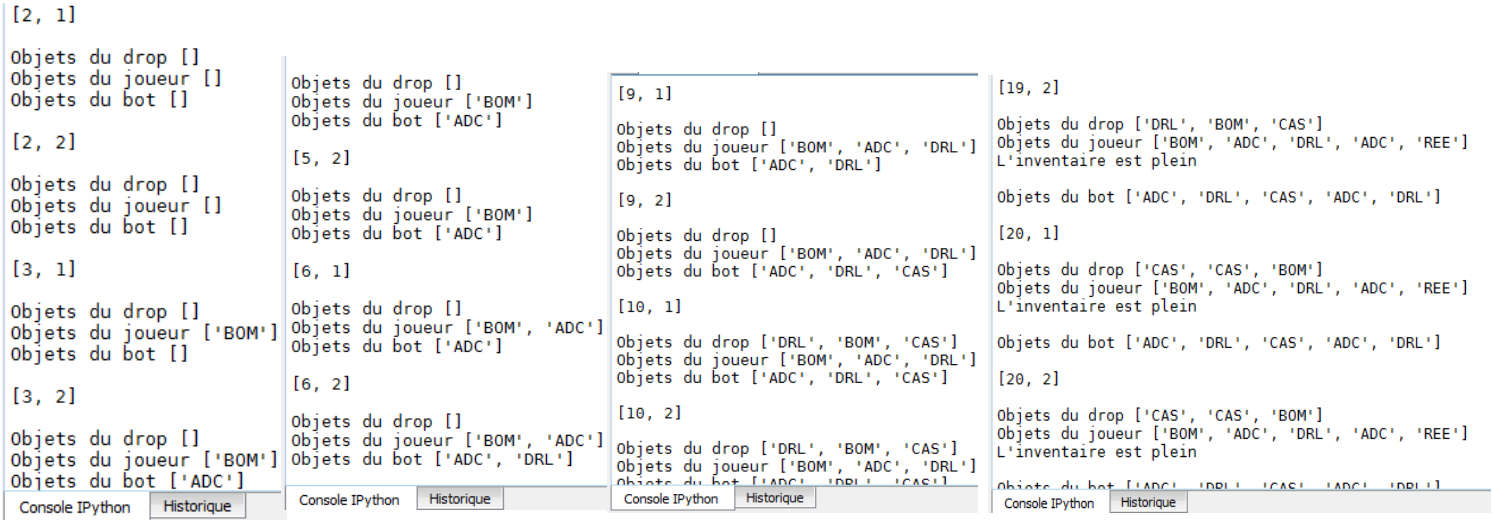


Figure 6 : Fonctionnement du programme

Le joueur choisit un objet, il reçoit ses caractéristiques et l'objet est supprimé de son inventaire. Même chose pour le bot.

```
[24, 2]

Objets du drop ['CAS', 'CAS', 'BOM']
Objets du joueur ['BOM', 'ADC', 'DRL', 'ADC', 'REE']
L'inventaire est plein

Objets du bot ['ADC', 'DRL', 'CAS', 'ADC', 'DRL']

[25, 1]


numero_objet = 3
[3, 50, 15, 7]
['BOM', 'ADC', 'ADC', 'REE']
[25, 2]


objet_gene = DRL
[3, 50, 15, 7]
['ADC', 'CAS', 'ADC', 'DRL']
```

7. AMELIORATION

Tout d'abord, je pense que mon programme peut être facilement optimisé en utilisant des fonctions de Python que je ne connais pas encore, en raccourcissant des blocs d'instructions ou en gérant mieux le nombres de variables.

Puis on peut facilement ajouter des instructions qui permettraient au programme d'accomplir plus de tâches. Comme un plus grand nombre d'objet, des conditions plus importantes sur les objets. Par exemple intégrer un système qui diminue la puissance de la bombe lorsque celle-ci est lancée loin.

8. CONCLUSION

Ainsi mon programme est sujet à amélioration, mais je pense avoir atteint les objectifs fixés.

La réalisation du programme en lui-même e m'as pas semblée compliquée contrairement à ce que je pensais au départ, mais c'est toute la préparation qui fut fastidieuse car je n'avais jamais mis en œuvre lors de mes précédents programmes.

Auparavant j'avais déjà fait des petits programmes, notamment pour m'amuser, que je développais directement sans travail au préalable. Lorsque j'ai commencé le projet j'ai découvert une nouvelle manière de créer un programme et la rigueur qui l'accompagne.

J'ai choisi l'option ISN car j'aspire à des études en informatique et je souhaitais voir à quoi ressemblais le travail dans cette discipline. J'ai appris un nouveau langage qui est Python, dont je ne connaissais rien et que je compte approfondir par la suite. Ce fut une expérience enrichissante qui m'a confortée dans mes choix de poursuites d'études.

9. WEBOGRAPHIE

8.1 Liens images

Logo Python : Document de présentation du projet
file:///F:/Lyc%C3%A9e/TS1/ISN/Projet/s%C3%A9quence%200%20projet/index.html?Rponselaproblmatique.html

Toutes les images proviennent de documents que j'ai créés.

8.2 Autres liens

http://apprendre-python.com/

https://openclassrooms.com/courses/apprenez-a-programmer-en-python

10.ANNEXE

9.1 Programme

Figure 7 : Programme

```
            ##########################################################
                  # Première Fonction: Génération d'un objet
            ##########################################################


##############################################
#  Génération des objets: Fonction Secondaire
##############################################

def choix_objet():
  proba_classe = randint(1,100) # Générer une classe d'objet "aléatoirement"
  if proba_classe <= 50:        # On choisit un un objet dans la classe 1
    proba_objet = randint(1,2)
    if proba_objet == 1:
      objet_genere_aleatoire = 'BOM'
    else:
      objet_genere_aleatoire = 'REE'
  elif 50 < proba_classe <= 85: # On choisit un un objet dans la classe 2
    proba_objet = randint(1,2)
    if proba_objet == 1:
      objet_genere_aleatoire = 'DRL'
    else:
      objet_genere_aleatoire = 'ADC'
  else:                         # On choisit un un objet dans la classe 3
    objet_genere_aleatoire = 'CAS'
  return objet_genere_aleatoire


##############################################
#   Génération des objets: Fonction Principale
##############################################

def generation_objet(tour):

  # Déclaration des différentes variables
  nombre_objet_generes = 0
  droit = 0 # Désigne qui a le droit de recevoir un objet(drop, joueur ou bot)
  objet_generes_aleatoire = ""

  # On vérifie que les objets du drop apparaisent tout les 10 tours
  if tour[0]%10 == 0 and tour[0] != 0 and tour[1] != 2:
    nombre_objet_generes = 3
    droit = 1
  # On vérifie que les objets du joueur tout les 3 tours(SES tours)
  elif tour[0]%3 == 0 and tour[0] != 0 and tour[1] == 1:
    nombre_objet_generes = 1
```

```
    droit = 2
  # On vérifie que les objets du bot tout les 3 tours(SES tours)
  elif tour[0]%3 == 0 and tour[0] != 0 and tour[1] == 2:
    nombre_objet_generes = 1
    droit = 3

  for j in range(nombre_objet_generes): # On génère le nombre d'objets nécessaire
    if droit == 1:                      # Les objets générés sont pour le drop
      if len(drop) == 3:                # On change le drop au bout de 10 tours, si
        for n in range(len(drop)):
          drop.remove(drop[0])
      drop.append(choix_objet())
    elif droit == 2:                    # Les objets générés sont pour le joueur
      if len(inventaire_joueur) < 5:    # L'inventaire du joueur ne peut contenir pl
        inventaire_joueur.append(choix_objet())
    elif droit == 3:                    # Les objets générés sont pour le bot
      if len(inventaire_bot) < 5:       # L'inventaire du joueur ne peut contenir pl
        inventaire_bot.append(choix_objet())

  return inventaire_bot, inventaire_joueur, drop

            ############################################################
                  # Deuxième Fonction: Utilisation d'un objet
            ############################################################


##################################################
#  Utilisation des objets: Fonctions Secondaires
##################################################

def bombes(coordonnees):                    # Objet qui détruit les cases adverses
  caracteristique_BOM =[1, 20]
  caracteristique_BOM = caracteristique_BOM + coordonnees
  return caracteristique_BOM

def regenerateur_energie():     # Objet qui remet l'énergie au maximum suivant le to
  caracteristique_REE = [2, 20, 100]
  return caracteristique_REE

def drop_leurre(coordonnees):               # Objet qui trompe l'adversaire
  caracteristique_DRL = [3, 50]
  caracteristique_DRL = caracteristique_DRL + coordonnees
  return caracteristique_DRL

def accelerateur_de_case():     # Objet qui accélère l'avancement du jeu
  caracteristique_ADC = [4, 50, 2]
```

```
93      return caracteristique_ADC
94
95  def case_supplementaire(coordonnees):      # Objet qui autorise 2 coups dans le même tour
96      caracteristique_CAS = [5, 70, 1]
97      caracteristique_CAS = caracteristique_CAS + coordonnees
98      return caracteristique_CAS
99
100
101 ################################################
102 #   Utilisation des objets: Fonction Principale
103 ################################################
104
105 def objet_choisit(tour, numero_objet, coordonnees):
106     if tour[1] == 1:      # Le joueur utilise un objet ou récupère le drop
107         if inventaire_joueur[numero_objet] == 'BOM':
108             return bombes(coordonnees)
109         elif inventaire_joueur[numero_objet] == 'REE':
110             return regenerateur_energie()
111         elif inventaire_joueur[numero_objet] == 'DRL':
112             return drop_leurre(coordonnees)
113         elif inventaire_joueur[numero_objet] == 'ADC':
114             return accelerateur_de_case()
115         elif inventaire_joueur[numero_objet] == 'CAS':
116             return case_supplementaire(coordonnees)
117
118
119     elif tour[1] == 2:    # Le bot utilise un objet ou récupère le drop
120         if inventaire_bot[numero_objet_bot] == 'BOM':
121             return bombes(coordonnees)
122         elif inventaire_bot[numero_objet_bot] == 'REE':
123             return regenerateur_energie()
124         elif inventaire_bot[numero_objet_bot] == 'DRL':
125             return drop_leurre(coordonnees)
126         elif inventaire_bot[numero_objet_bot] == 'ADC':
127             return accelerateur_de_case()
128         elif inventaire_bot[numero_objet_bot] == 'CAS':
129             return case_supplementaire(coordonnees)
130
131
132 ############################           TEST            ############################
133
134 from random import randint #*#
135
136
137 x = randint(1, 16)
138 y = randint(1, 16)
```

```
tour = [0,1]
numero_objet_bot = 0 #*#
caracteristique_BOM = [] #*#
caracteristique_REE = [] #*#
caracteristique_DRL = [] #*#
caractéristique_ADC = [] #*#
caracteristique_CAS = [] #*#
objet_generes_aleatoire = ""

global objet_gene #*#
objet_gene = '' #*#

global numero_objet #*#
numero_objet = 0 #*#

global inventaire_joueur #*#
inventaire_joueur = [] #*#

global inventaire_bot #*#
inventaire_bot = [] #*#

global drop #*#
drop = [] #*#

global Jauge_Drain_joueur
Jauge_Drain_joueur = 50

global Jauge_Drain_bot
Jauge_Drain_bot = 50

global objet_utilise #*#
objet_utilise = [] #*#

global coordonnees
coordonnees = [x, y]



for n in range(50):  # Génération sur 25 tours
  print(tour)
  print('')

  generation_objet(tour) #*#
  print("Objets du drop", drop)
  print("Objets du joueur", inventaire_joueur)
  if len(inventaire_joueur) == 5: #*#
```

```
    print("L'inventaire est plein") #*#
    print("")
  print("Objets du bot", inventaire_bot)
  print('')
  if tour[1] == 1:
    tour[1] = 2
  else:
    tour[1] = 1
    tour[0] += 1



# Utilisation des objets

for n in range(2):  #  A retirer pour l'integration

  # Le joueur choisit un objet
  if tour[1] == 1:  #*#
    print(tour)
    print('')
    numero_objet = int(input('numero_objet = ')) # On vérifie de quel objet il s'agi
    numero_objet = numero_objet - 1   #*#
    objet_utilise = objet_choisit(tour, numero_objet, coordonnees) # On trouve les ca
    print(objet_utilise)

    if Jauge_Drain_joueur < objet_utilise[1]: #on vérifie qu'on a la capcité d'utili
      print("Impossible d'utiliser l'objet") #*#
      tour[1] = 2
      # Tour suivant
    else: #*#
      # Envoi de la liste objet-utilise au plateau pour les modifications
      del inventaire_joueur[numero_objet] #*#
      print(inventaire_joueur)
      tour[1] = 2
      # Tour suivant

 # Le bot utilise un objet
  elif tour[1] == 2:  #*#
    print(tour)
    print('')
    objet_gene = input('objet_gene = ') # On vérifie de quel objet il s'agit    #*#
    # Taper le nom de l'objet sans paranthèse, sinon le programe ne trouvera jamais

    while objet_gene != inventaire_bot[numero_objet_bot] or numero_objet_bot > 4:  #
      if objet_gene != inventaire_bot[numero_objet_bot]: #*#
        numero_objet_bot += 1 #*#
      else: #*#
        numero_objet_bot =+ 1 #*#
        print(numero_objet_bot)

    objet_utilise = objet_choisit(tour, objet_gene, coordonnees) # On trouve les cara
    print(objet_utilise)

    if Jauge_Drain_bot < objet_utilise[1]: #on vérifie qu'on a la capcité d'utiliser
      print("Impossible d'utiliser l'objet") #*#
      # Tour suivant
      tour[1] = 1
      tour[0] += 1
    else:   #*#
      # Envoi de la liste objet-utilise au plateau pour les modifications
      del inventaire_bot[numero_objet_bot]   #*#
      # Tour suivant
      print(inventaire_bot)
      tour[1] = 1
      tour[0] += 1


# Intégration

## A l'intégration certaines lignes du test seront à intégrées dans une fonction
## objet() qui sera celle appelée par le main, elles seront annotées de #*#.
## Les lignes que j'aurai signalées ne seront peut-être pas exhaustives
```

9.2 Algorithme génération

Voici l'algorithme correspondant à la génération des objets.

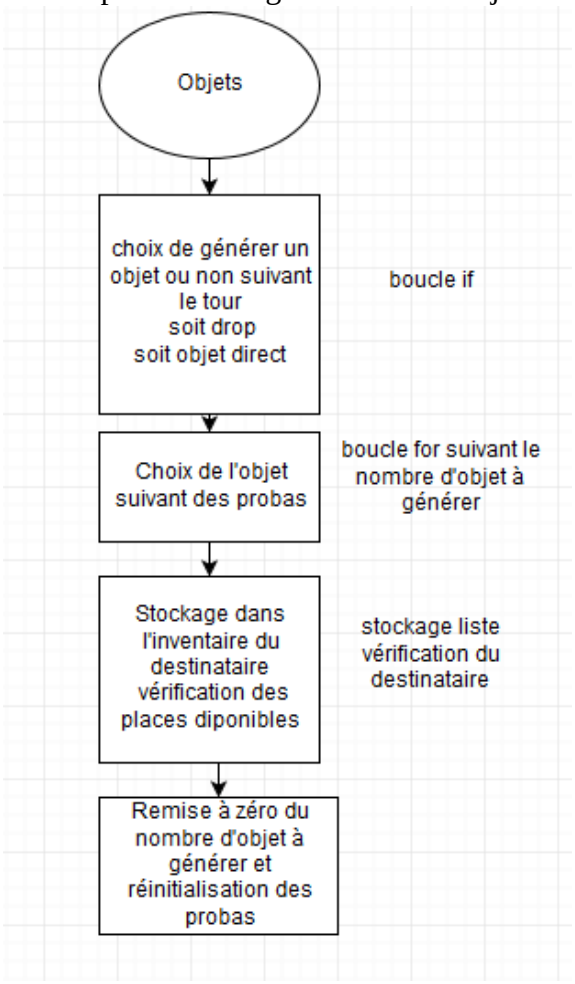


Figure 8 : Algorithme génération

9.2 Algorithme utilisation

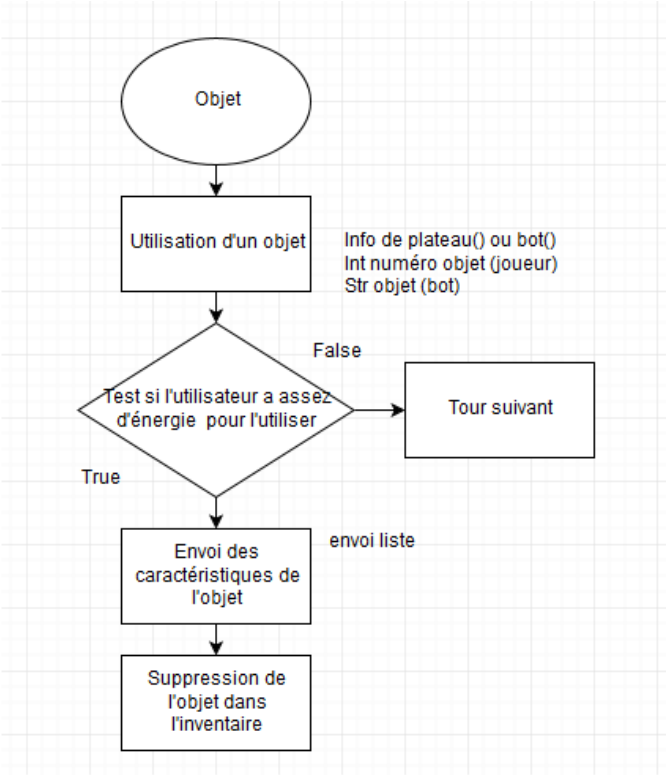


Figure 9 : Algorithme utilisation

9.3 Indice des listes des caractéristiques d'objets

Voici le document explicatif des caractéristiques des objets. Il associe à chaque indice de la liste la « fonction » du nombre s'y trouvant.

Exemple : Bombes : L'indice 0 de la liste renvoyée correspond à la désignation de l'objet ici 1. Pour le Drop Leurre, la liste aura pour indice 0 : 3.

Fonction Bombe () : fini
Indice liste
- 0 - désignation objet
- 1 - coup_drain
- 2 - coordonnées selon x
- 3 - coordonnées selon y

Fonction Régénérateur_énergie(): fini
Indice liste
- 0 - désignation objet
- 1 - coup_drain
- 2 -Jauge_Drain

Fonction Drop Leurre(): fini
Indice liste
- 0 - désignation objet
- 1 - coup_drain
- 2 - coordonnées selon x
- 3 - coordonnées selon y

Fonction Accélérateur_de_case(): fini
Indice liste
- 0 - désignation objet
- 1 - coup_drain
- 2 - accélération

Fonction Case_supplémentaire(): fini
Indice liste
- 0 - désignation objet
- 1 - coup_drain
- 2 - coup supplémentaire
- 3 - coordonnées selon x
- 4 - coordonnées selon y

Figure : Indices liste