Annexe

Programme principal

```
# -*- coding: utf-8 -*-
#! /usr/bin/python

import pygame
from pygame.locals import *
pygame.init()
fenetre=pygame.display.set_mode((640,640))
pygame.display.set_caption('Programme Pygame de base')

import Carte.carte as carte
import Carte.script as script
import Carte.inventaire as inventaire
import combat.perso as perso
import Carte.menu as menu

import sys
import Carte.objet as objet
import Carte.scriptpa as script_pa
import combat.combat as combat
import pickle
import random
#sys.path.append()

#definition des couleurs
black=(0,0,0)
white=(0xFF,0xFF,0xFF)
red=(255,0,0)

#initialisation de la fenetre
taille_fenetre=20


#creation d'une liste contenant tous les personnages jouable
joueur =
[perso.AssassinsMagique,perso.AssassinsPhysique,perso.Combattant,perso.Mage,perso.Soigneur,perso.Archer]

#creation des variables pour ecrire un texte
font = pygame.font.SysFont('Calibri', 25, True, False)
ecrire = font.render

#creation de la vatiable permettant de dessiner des rectangles
dessiner=pygame.draw.rect

#initialisations de quelques variables utiles
x=0
y=0


dict_fleche = {K_DOWN: "bas", K_UP:"haut", K_RIGHT:"droite", K_LEFT:"gauche"}

#declanchement de la repetition des touches
pygame.key.set_repeat(300,70)

#declanchement de la musique
pygame.mixer.music.load('data/Linconnue V4 Pont.wav')
pygame.mixer.music.play(-1)
```

```
continuer = True

joueur=menu.start_menu(fenetre,joueur)

def affichercarte(x0,y0, xmvt, ymvt):

    for x in range(-1, taille_fenetre+1):
        for y in range(-1, taille_fenetre+1):
            if xmvt:
                posx = (x+mvt_perso.dict_dir[mvt_perso.direction][0])*32 -mvt_perso.dict_dir[mvt_perso.direction]
[0]*4*mvt_perso.stade_animation
            else:
                posx = x*32
            if ymvt:
                posy = (y+mvt_perso.dict_dir[mvt_perso.direction][1])*32 -mvt_perso.dict_dir[mvt_perso.direction]
[1]*4*mvt_perso.stade_animation
            else:
                posy = y*32
            fenetre.blit(mape.get_image_case(x+x0, y+y0),(posx, posy))

            if x+x0>= len(mape.matrice_objet):
                x=x-1
            if y+y0>= len(mape.matrice_objet[0]):
                y=y-1

            if mape.matrice_objet[x+x0][y+y0]!= None and mape.matrice_objet[x+x0][y+y0] != mvt_perso:
                fenetre.blit(mape.get_image_obj(x+x0, y+y0),(posx, posy))

def afficher_joueur(x0, y0, xmvt, ymvt):
    if mvt_perso.stade_animation!= 0:
        if not xmvt:
            imgx = (32*(mvt_perso.posx-mvt_perso.dict_dir[mvt_perso.direction][0]-x0))
+(4*mvt_perso.dict_dir[mvt_perso.direction][0]*mvt_perso.stade_animation)-16
        else:
            imgx = (32*(mvt_perso.posx-x0))-16
        if not ymvt:
            imgy = (32*(mvt_perso.posy-mvt_perso.dict_dir[mvt_perso.direction][1]-y0))
+(4*mvt_perso.dict_dir[mvt_perso.direction][1]*mvt_perso.stade_animation)-32
        else:
            imgy = (32*(mvt_perso.posy-y0))-32
    else:
        imgx = (32*(mvt_perso.posx-x0))-16
        imgy = (32*(mvt_perso.posy-y0))-32

    fenetre.blit(mvt_perso.dict_images[mvt_perso.direction][mvt_perso.stade_animation],(imgx, imgy))

def affiche_pv(joueur,fenetre):
    prct_pv=(joueur.pv/joueur.pv_max)*100
    dessiner(fenetre,black,(9,9,102,12),1)
    dessiner(fenetre,red,(10,10,prct_pv,10))
    fenetre.blit(ecrire(str(joueur.pv)+"/"+str(joueur.pv_max),True,black),(120,10))

def afficher_ecran():
# affichage
    x_mvt_carte = True
    y_mvt_carte = True

    if mvt_perso.posx<10:
        x0 = 0
    elif mvt_perso.posx>=taille_carte-11:
        x0 = taille_carte-20
        x_mvt_carte = False
    else:
```

```
        x0 = mvt_perso.posx-10

    if mvt_perso.posy<10:
        y0 = 0
        y_mvt_carte = False
    elif mvt_perso.posy>=taille_carte-11:
        y0 = taille_carte-20
        y_mvt_carte = False
    else:
        y0 = mvt_perso.posy-10

    if mvt_perso.stade_animation == 0:
        x_mvt_carte = False
        y_mvt_carte = False

    if mvt_perso.posx<11:
        x_mvt_carte = False
    if mvt_perso.posx>=taille_carte-9:
        x_mvt_carte = False
    if mvt_perso.posy<11:
        y_mvt_carte = False
    if mvt_perso.posy>=taille_carte-9:
        y_mvt_carte = False

    affichercarte(x0,y0, x_mvt_carte, y_mvt_carte)
    afficher_joueur(x0, y0, x_mvt_carte, y_mvt_carte)
    affiche_pv(joueur,fenetre)

#fonction affichant l'ecran de fin de jeu lors d'une defaite
def gameover(fenetre,joueur):
    continuer=True
    perdu = pygame.font.SysFont('Calibri',40,True,False)
    game_over = perdu.render("GameOver",True,red)
    fenetre.fill(black)
    fenetre.blit(game_over,game_over.get_rect(center=(320, 150)))
    dead= joueur.image_complet
    appuie = font.render("Appuiez sur Entrer pour quitter",True,white)
    fenetre.blit(appuie,appuie.get_rect(center=(320,450)))
    for i in range (0,5):
        dead_img = dead.subsurface(i*64,1280,64,64)
        fenetre.fill(black,(288,288,64,64))
        fenetre.blit(dead_img,dead_img.get_rect(center=(320,320)))
        pygame.time.wait(200)
        pygame.display.flip()
    while continuer:
        for event in pygame.event.get():
            if event.type == QUIT:
                continuer = False
            if event.type == KEYDOWN:
                if event.key == K_RETURN:
                    continuer = False

    #creation de la carte et initialisation du deplacement
mape = carte.carte(joueur)
taille_carte = mape.taille_mat[0]


if joueur == "End":
    continuer = False
else:
    mvt_perso = objet.perso(mape, 47, 45, joueur.ls_images)
    script_pa.script_pa(fenetre)
    pygame.mixer.music.load('data/compo 1.wav')
```

```
    pygame.mixer.music.play(-1)

direction_sauvegardee= ""

while continuer:
    # prise en compte des evenements
    for event in pygame.event.get():
        if event.type == QUIT:
            continuer = False

        if event.type == KEYDOWN:
            if mvt_perso.stade_animation == 0:
                if event.key in [K_DOWN, K_UP, K_LEFT, K_RIGHT]:

                    if mvt_perso.direction != dict_fleche[event.key]:
                        mvt_perso.ch_direction(dict_fleche[event.key])
                        mvt_perso.avancer()
                    else:
                        mvt_perso.avancer()

            if event.key == K_ESCAPE:
                info = menu.menupause(fenetre,joueur)
                if info == "End":
                    continuer = False
                if info != None:
                    j=info

            if event.key == K_s:
                script_pa.script_pa(fenetre)

            if event.key == K_i:
                 inventaire.inventaire(fenetre,joueur)

    #Decleclenchement du combat
    if mape.combat[0] == True:
        a = combat.affiche_combat(fenetre,joueur, mape.combat[1].ls_ennemi)
        if a == "fin":
            print mape.combat[1]

            mape.combat[1].effacer()
        elif a == "Fuite":
            pass
        elif a == "Mort joueur":
            continuer = False
            gameover(fenetre, joueur)
        mape.combat = [False]

    if mape.carte_changee:
        mvt_perso = objet.perso(mape, 47, 47, joueur.ls_images)
        mape.carte_changee = False

    afficher_ecran()
    pygame.display.flip()
    mvt_perso.step()
pygame.quit()
```

Catre/carte.py

```
# -*- coding: utf-8 -*-
#! /usr/bin/python
import pygame
from pygame.locals import*

from cases import*
from objet import*

import random
import pickle
import sys
sys.path.append("..")
import combat.perso as perso


class carte:

    def __init__(self,  joueur):
        self.joueur = joueur
        self.mob_niv1 = [[perso.Rats, perso.Rats, perso.Rats, perso.Rats ],[ perso.Gobelins, perso.Gobelins,
perso.Gobelins, perso.Gobelins],[perso.Aigles, perso.Aigles ],[ perso.Slime ], [perso.Carapateur]]
        self.mob_niv2 = [[perso.Centaures, perso.Centaures, perso.Centaures],[perso.Loup_garou],
[perso.Araingnees,perso.Araingnees,perso.Araingnees], [perso.Carapateur],[perso.Golems,perso.Golems]]
        self.mob_niv3 = [[perso.Carapateur],[perso.Golems,perso.Golems],[perso.Treant],[perso.Geant],
[perso.Nains,perso.Nains,perso.Nains],[perso.Elfs,perso.Elfs]]
        self.ls_ennemi = [self.mob_niv1, self.mob_niv2, self.mob_niv3]

        self.niv_carte = 0
        self.cartes = ["carte.mp", "carten2.mp", "carten3.mp"]
        self.combat = [False]
        self.carte_changee = False
        self.fichier = self.cartes[self.niv_carte]
        self.recup_map()
        self.taille_mat = [len(self.matrice_case), len(self.matrice_case[0])]

        # création de la matrice qu sert au cases dde la carte
        self.dict_cases = {
                           "herbe1": herbe1(),
                           "herbe2": herbe2(),
                           "herbe3": herbe3(),
                           "herbe4": herbe4(),
                           "mur": mur(),
                           "murs": murs(),
                           "sols": sols(),
                           "trous":trous()
                          }
        #self.matrice_case = [[[] for a in range(self.taille_mat[1])] for a in range(self.taille_mat[0])]

        # création de la matrice qui sert gérer les objet de la scène
        #self.matrice_objet = [[None for a in range(self.taille_mat[1])] for a in range(self.taille_mat[0])]


    def recup_map(self):
        double_mat = []
        self.fichier = self.cartes[self.niv_carte]
        with open(self.fichier, "r") as fichier:
            pick = pickle.Unpickler(fichier)
            double_mat = pick.load()
        mat_case = double_mat[0]
        self.matrice_case = mat_case
        mat_objet = double_mat[1]
```

```
        for x in range(len(mat_case)):
            for y in range(len(mat_case[0])):
                if mat_case[x][y] == "herbe":
                    self.matrice_case[x][y] = "herbe"+str(random.randrange(1, 5))

        self.matrice_objet = [[None for a in range(len(mat_objet))] for a in range(len(mat_objet[0]))]

        for x in range(len(mat_objet)):
            for y in range(len(mat_objet[0])):
                if mat_objet[x][y] != "":
                    if mat_objet[x][y] == "ennemi":
                        e = random.randrange(len(self.ls_ennemi[self.niv_carte]))
                        ls_en = []
                        for a in self.ls_ennemi[self.niv_carte][e]:
                            ls_en.append(a())
                        self.matrice_objet[x][y] = ennemi(self, x, y, ls_en)

                    elif mat_objet[x][y] == "coffre":
                        contenu = self.potions_aleatoires()
                        self.matrice_objet[x][y] = coffre(self, x, y, contenu)


                    elif mat_objet[x][y] == "boss":
                        e = 0
                        ls_en = []
                        for a in self.ls_ennemi[0][e]:
                            ls_en.append(a())
                        self.matrice_objet[x][y] = ennemi(self, x, y, ls_en)
                    else:
                        exec("self.matrice_objet[x][y] = "+mat_objet[x][y]+"(self, x, y)")

    def changer_carte(self):
        self.niv_carte += 1
        self.recup_map()
        self.carte_changee = True

    def potions_aleatoires(self):
        ls_potions = [
            "potionvie1",
            "potionvie2",
            "potionvie3",
            "potionarmure1",
            "potionarmure2",
            "potionarmure3",
            "potionforce1",
            "potionforce2",
            "potionforce3",
            "potioncritique1",
            "potioncritique2",
            "potioncritique3",
            "potionvitesse",
            "potionprecision"]

        potions_renvoi = []

        for i in range(2):
            potions_renvoi.append(ls_potions[random.randrange(len(ls_potions))])

        return potions_renvoi
    def get_image_case(self, x, y):
        if x>=len(self.matrice_case):
            x = len(self.matrice_case)-1
```

```
        if y>=len(self.matrice_case[0]):
            y = len(self.matrice_case[0])-1
        return self.dict_cases[self.matrice_case[x][y]].image

    def get_image_obj(self, x, y):
        return self.matrice_objet[x][y].image


    def __repr__(self):
        ch = ""
        for x in range(self.taille_mat[0]):
            for y in range(self.taille_mat[1]):
                if self.matrice_objet[x][y] != None:
                    ch += self.matrice_objet[x][y].rep
                else:
                    ch += self.dict_cases[self.matrice_case[x][y]].rep
            ch += "\n"
        return ch

    def placer_obj(self,objet, x, y):
        self.matrice_objet[x][y] = objet

    def effacer_obj(self, x, y):
        self.matrice_objet[x][y] = None
    def est_marchable(self, x, y):
        return self.dict_cases[self.matrice_case[x][y]].marchable

    def deplacer(self, (x1, y1), (x2, y2)):
        self.matrice_objet[x2][y2] = self.matrice_objet[x1][y1]
        self.matrice_objet[x1][y1] = None

    def obj_vide(self, x1, y1):
        if self.matrice_objet[x1][y1] == None:
            return True
        else:
            return False


if __name__ == "__main__":
    # exemple de fonctionnement de la carte et de objets
    print("")
    print("")
    print("")
    #sys.path.append("..")
    print sys.path
    dir(sys)
    a = carte("carte.mp")


    print a
    b = obj_boug(a, 5, 5)
    c = objet(a, 6, 5)
    b.ch_direction("bas")
    print a
```

Catre/cases.py

```
# -*- coding: utf-8 -*-
import pygame
from pygame.locals import*

from carte import*
from objet import*

class case:
    def __init__(self):
        self.type = "case"
        self.marchable = True
        self.ouvrable = False
        self.chemin_image = ""
        self.rep = " "
        self.image = None

    def load_image(self):
        self.image = pygame.image.load(self.chemin_image)

    def __repr__(self):
        if self.dessus == None:
            return self.rep
        else:
            return self.dessus.rep

    def __str__(self):
        return "case de type {} a la position {} \n marchable = {} ouvrable = {} l'image ce trouve en {}".format(
        self.type, self.pos, self.marchable, self.ouvrable, self.image)


class mur(case):
    def __init__(self):
      case.__init__(self)
      self.type = "mur"
      self.marchable = False
      self.rep = "M"
      self.chemin_image = "data/Mur.png"
      self.load_image()

class herbe1(case):
    def __init__(self):
      case.__init__(self)
      self.type = "herbe1"
      self.rep = "H"
      self.chemin_image = "data/Grass17.png"
      self.load_image()

class herbe2(case):
    def __init__(self):
      case.__init__(self)
      self.type = "herbe2"
      self.rep = "H"
      self.chemin_image = "data/Grass04.png"
      self.load_image()

class herbe3(case):
    def __init__(self):
      case.__init__(self)
      self.type = "herbe3"
      self.rep = "H"
      self.chemin_image = "data/Grass03.png"
```

```
        self.load_image()

class herbe4(case):
    def __init__(self):
        case.__init__(self)
        self.type = "herbe4"
        self.rep = "H"
        self.chemin_image = "data/Grass02.png"
        self.load_image()

class murs(case):
    def __init__(self):
        case.__init__(self)
        self.type = "murs"
        self.rep = "M"
        self.chemin_image = "data/Mur.png"
        self.marchable = False
        self.load_image()

class sols(case):
    def __init__(self):
        case.__init__(self)
        self.type = "sols"
        self.rep = "S"
        self.chemin_image = "data/Grass01.png"

        self.load_image()


class trous(case):
    def __init__(self):
        case.__init__(self)
        self.type = "trous"
        self.rep = "T"
        self.chemin_image = "data/trous.png"
        self.marchable = False
        self.load_image()
```

Catre/inventaire.py

```
# -*- coding: utf-8 -*-
"""
Created on Fri Apr 14 11:24:09 2017

@author: nassim.zaki et martin.abelard
"""
import pygame
from pygame.locals import *

#pygame.init()
#fenetre=pygame.display.set_mode((640,640))
#pygame.display.set_caption('Programme Pygame de base')
#Joueur = perso.AssassinsMagique()
#Joueur.potionvie1 += 10
#Joueur.pv -= 10

charger = pygame.image.load
black=(0,0,0)
blanc=(0xFF, 0xFF, 0xFF)
jaune=(255, 255, 153)

def inventaire(fenetre,Joueur):
    continuer = 1
    font = pygame.font.SysFont('Calibri', 25, False, False)
    position_bouton = 0
    dessiner=pygame.draw.rect
    ecrire=font.render
    while continuer == 1:
        fenetre.fill(jaune)

#       pygame.draw.rect(fenetre, blanc, [20, 2+45*(position_bouton-1), 450, 40], 3)
#       pygame.draw.rect(fenetre, blanc, [20, 2+45*(position_bouton+1), 450, 40], 3)
        pygame.draw.rect(fenetre, black, [20, 2+45*position_bouton, 450, 40], 3)

        dessiner(fenetre, black,[20, 2, 450, 40],1)
        potionvie1=ecrire("Petite potion de vie",True, black)
        fenetre.blit(potionvie1,(80,10))
        fenetre.blit(charger("data/potionv2_40.png"),(40,2))
        fenetre.blit(ecrire(str(Joueur.potionvie1),True,black),(410,10))

        dessiner(fenetre, black,[20, 47, 450, 40],1)
        potionvie2=ecrire("Grande potion de vie",True, black)
        fenetre.blit(potionvie2,(80,55))
        fenetre.blit(charger("data/perso.png"),(40,47))
        fenetre.blit(ecrire(str(Joueur.potionvie2),True,black),(410,55))

        dessiner(fenetre, black,[20, 92, 450, 40],1)
        potionvie3=ecrire("Enorme potion de vie",True, black)
        fenetre.blit(potionvie3,(80,100))
        fenetre.blit(charger("data/potionv3_40.png"),(40,92))
        fenetre.blit(ecrire(str(Joueur.potionvie3),True,black),(410,100))

        dessiner(fenetre, black,[20, 137, 450, 40],1)
        potionarmure1=ecrire("Petite potion d'armure",True, black)
        fenetre.blit(potionarmure1,(80,145))
        fenetre.blit(charger("data/potiona2_40.png"),(40,137))
        fenetre.blit(ecrire(str(Joueur.potionarmure1),True,black),(410,145))

        dessiner(fenetre, black,[20, 182, 450, 40],1)
        potionarmure2=ecrire("Grande potion d'armure",True, black)
        fenetre.blit(potionarmure2,(80,190))
```

```
fenetre.blit(charger("data/potiona1_40.png"),(40,182))
fenetre.blit(ecrire(str(Joueur.potionarmure2),True,black),(410,190))

dessiner(fenetre, black,[20, 227, 450, 40],1)
potionarmure3=ecrire("Enorme potion d'armure",True, black)
fenetre.blit(potionarmure3,(80,235))
fenetre.blit(charger("data/potiona3_40.png"),(40,227))
fenetre.blit(ecrire(str(Joueur.potionarmure3),True,black),(410,235))

dessiner(fenetre, black,[20, 272, 450, 40],1)
potionforce1=ecrire("Petite potion de force",True, black)
fenetre.blit(potionforce1,(80,280))
fenetre.blit(charger("data/potionf2_40.png"),(40,272))
fenetre.blit(ecrire(str(Joueur.potionforce1),True,black),(410,280))

dessiner(fenetre, black,[20, 317, 450, 40],1)
potionforce2=ecrire("Grande potion de force",True, black)
fenetre.blit(potionforce2,(80,325))
fenetre.blit(charger("data/potionf1_40.png"),(40,317))
fenetre.blit(ecrire(str(Joueur.potionforce2),True,black),(410,325))

dessiner(fenetre, black,[20, 362, 450, 40],1)
potionforce3=ecrire("Enorme potion de force",True, black)
fenetre.blit(potionforce3,(80,370))
fenetre.blit(charger("data/potionf3_40.png"),(40,362))
fenetre.blit(ecrire(str(Joueur.potionforce3),True,black),(410,370))

dessiner(fenetre, black,[20, 407, 450, 40],1)
potioncritique1=ecrire("Petite potion de critique",True, black)
fenetre.blit(potioncritique1,(80,415))
fenetre.blit(charger("data/potioncrit2_40.png"),(40,407))
fenetre.blit(ecrire(str(Joueur.potioncritique1),True,black),(410,415))

dessiner(fenetre, black,[20, 452, 450, 40],1)
potioncritique2=ecrire("Grande potion de critique",True, black)
fenetre.blit(potioncritique2,(80,460))
fenetre.blit(charger("data/perso.png"),(40,452))
fenetre.blit(ecrire(str(Joueur.potioncritique2),True,black),(410,460))

dessiner(fenetre, black,[20, 497, 450, 40],1)
potioncritique3=ecrire("Enorme potion de critique",True, black)
fenetre.blit(potioncritique3,(80,505))
fenetre.blit(charger("data/perso.png"),(40,497))
fenetre.blit(ecrire(str(Joueur.potioncritique3),True,black),(410,505))

dessiner(fenetre, black,[20, 542, 450, 40],1)
potionvitesse=ecrire("Potion de vitesse",True, black)
fenetre.blit(potionvitesse,(80,550))
fenetre.blit(charger("data/potionvit1_40.png"),(40,542))
fenetre.blit(ecrire(str(Joueur.potionvitesse),True,black),(410,550))

dessiner(fenetre, black,[20, 587, 450, 40],1)
potionprecision=ecrire("Potion de precision",True, black)
fenetre.blit(potionprecision,(80,595))
fenetre.blit(charger("data/perso.png"),(40,587))
fenetre.blit(ecrire(str(Joueur.potionprecision),True,black),(410,595))

        #affichage des stats du personnage
        #affichage de la vie
pvperso=ecrire("Pv: "+str(Joueur.pv),True,black)
pvperso_rect=pvperso.get_rect()
pvperso_rect.right = 630
pvperso_rect.top = 10
```

```
fenetre.blit(pvperso,pvperso_rect)
        #affichage de l'armure
if Joueur.defen > 90:
    armureperso=ecrire("Defense: "+str(90),True,black)
else:
    armureperso=ecrire("Defense: "+str(Joueur.defen),True,black)
armureperso_rect=armureperso.get_rect()
armureperso_rect.right = 630
armureperso_rect.top = 35
fenetre.blit(armureperso,armureperso_rect)
#affichage de l'attaque
atkperso=ecrire("Attaque: "+str(Joueur.atk),True,black)
atkperso_rect=atkperso.get_rect()
atkperso_rect.right = 630
atkperso_rect.top = 60
fenetre.blit(atkperso,atkperso_rect)
#affichage de la magie
magperso=ecrire("Magie: "+str(Joueur.mag),True,black)
magperso_rect=magperso.get_rect()
magperso_rect.right = 630
magperso_rect.top = 85
fenetre.blit(magperso,magperso_rect)
#affichage de la resistance
resperso=ecrire("Resistance: "+str(Joueur.res),True,black)
resperso_rect=resperso.get_rect()
resperso_rect.right = 630
resperso_rect.top = 110
fenetre.blit(resperso,resperso_rect)
#affichage de la vitesse
vitperso=ecrire("Vitesse: "+str(Joueur.vit),True,black)
vitperso_rect=vitperso.get_rect()
vitperso_rect.right = 630
vitperso_rect.top = 135
fenetre.blit(vitperso,vitperso_rect)
#affichage de la precision
precperso=ecrire("Precision: "+str(Joueur.prec),True,black)
precperso_rect=precperso.get_rect()
precperso_rect.right = 630
precperso_rect.top = 160
fenetre.blit(precperso,precperso_rect)
#affichage des critique
critperso=ecrire("Critique: "+str(Joueur.crit),True,black)
critperso_rect=critperso.get_rect()
critperso_rect.right = 630
critperso_rect.top = 185
fenetre.blit(critperso,critperso_rect)

for event in pygame.event.get():
    if event.type == KEYDOWN:
        if event.key == K_i:
            continuer = 0
        if event.key == K_DOWN or event.key == K_RIGHT:
            if position_bouton <13:
                position_bouton += 1
        if event.key == K_UP or event.key == K_LEFT:
            if position_bouton >0:
                position_bouton -= 1
        if event.key == K_RETURN:
            if position_bouton==0:
                Joueur.action="vie1"
            if position_bouton==1:
                Joueur.action="vie2"
            if position_bouton==2:
```

```
                Joueur.action="vie3"
            if position_bouton==3:
                Joueur.action="armure1"
            if position_bouton==4:
                Joueur.action="armure2"
            if position_bouton==5:
                Joueur.action="armure3"
            if position_bouton==6:
                Joueur.action="force1"
            if position_bouton==7:
                Joueur.action="force2"
            if position_bouton==8:
                Joueur.action="force3"
            if position_bouton==9:
                Joueur.action="critique1"
            if position_bouton==10:
                Joueur.action="critique2"
            if position_bouton==11:
                Joueur.action="critique3"
            if position_bouton==12:
                Joueur.action="vitesse"
            if position_bouton==13:
                Joueur.action="precision"
            Joueur.popo_actif()

    pygame.display.flip()
```

Catre/menu.py

```
import pygame
from pygame.locals import *
pygame.init()

#definition des couleurs
black=(0,0,0)
white=(0xFF,0xFF,0xFF)
jaune=(255, 255, 153)

#creation des variables pour ecrire un texte
font = pygame.font.SysFont('Calibri', 25, True, False)
ecrire = font.render

#creation de la vatiable permettant de dessiner des rectangles
dessiner=pygame.draw.rect

def start_menu(fenetre,joueur):

    fenetre.fill(white)
    position_bouton=0
    img_menu = pygame.image.load("data/imagemenu.jpg")
    text1 = ecrire("Sebastien"+" "+"Assassin Magique",True,black)
    text2 = ecrire("Nassim"+" "+"Assassin Physique",True,black)
    text3 = ecrire("Pierre Antoine"+" "+"Combattant",True,black)
    text4 = ecrire("Emeric"+" "+"Mage",True,black)
    text5 = ecrire("Clarisse"+" "+"Soigneur",True,black)
    text6 = ecrire("Martin"+" "+"Archer",True,black)

    while True:
        fenetre.blit(img_menu,[0,0])
        pygame.draw.rect(fenetre, black, [170, 227+64*position_bouton, 300, 64], 3)
        dessiner(fenetre, black, [170, 547, 300, 64], 1)
        dessiner(fenetre, black, [170, 483, 300, 64], 1)
        dessiner(fenetre, black, [170, 419, 300, 64], 1)
        dessiner(fenetre, black, [170, 355, 300, 64], 1)
        dessiner(fenetre, black, [170, 291, 300, 64], 1)
        dessiner(fenetre, black, [170, 227, 300, 64], 1)

        fenetre.blit(text6, text6.get_rect(center=(fenetre.get_width()/2, 579)))
        fenetre.blit(text5, text5.get_rect(center=(fenetre.get_width()/2, 515)))
        fenetre.blit(text4, text4.get_rect(center=(fenetre.get_width()/2, 451)))
        fenetre.blit(text3, text3.get_rect(center=(fenetre.get_width()/2, 387)))
        fenetre.blit(text2, text2.get_rect(center=(fenetre.get_width()/2, 323)))
        fenetre.blit(text1, text1.get_rect(center=(fenetre.get_width()/2, 259)))

        for event in pygame.event.get():
            if event.type == QUIT:
                return "End"
            if event.type == KEYDOWN:
                if event.key == K_DOWN or event.key == K_RIGHT:
                    if position_bouton <5:
                        position_bouton += 1
                if event.key == K_UP or event.key == K_LEFT:
                    if position_bouton >0:
                        position_bouton -= 1
                if event.key == K_RETURN:
                    if position_bouton == 0:
                        pygame.mixer.music.load('data/01 - Chanson Pour l Auvergnat.mp3')
                        #pygame.mixer.music.play(-1)
                        return joueur[0]()
                    if position_bouton == 1:
```

```
                    pygame.mixer.music.load('data/01 - Chanson Pour l Auvergnat.mp3')
                    #pygame.mixer.music.play(-1)
                    return joueur[1]()
                if position_bouton == 2:
                    pygame.mixer.music.load('data/01 - Chanson Pour l Auvergnat.mp3')
                    #pygame.mixer.music.play(-1)
                    return joueur[2]()
                if position_bouton == 3:
                    pygame.mixer.music.load('data/01 - Chanson Pour l Auvergnat.mp3')
                    #pygame.mixer.music.play(-1)
                    return joueur[3]()
                if position_bouton == 4:
                    pygame.mixer.music.load('data/01 - Chanson Pour l Auvergnat.mp3')
                    #pygame.mixer.music.play(-1)
                    return joueur[4]()
                if position_bouton == 5:
                    pygame.mixer.music.load('data/01 - Chanson Pour l Auvergnat.mp3')
                    #pygame.mixer.music.play(-1)
                    return joueur[5]()
        pygame.display.flip()

def menupause(fenetre,joueur):
    pygame.mixer.music.stop()
    position_bouton = 0
    grandtitre=pygame.font.Font('data/fonts/old_london/OldLondon.ttf',55)
    text1 = ecrire("Reprendre",True,black)
    text2 = ecrire("Menu Principal",True,black)
    text3 = ecrire("Retour au Bureau",True,black)
    titre1=grandtitre.render("MENU PAUSE", True, black)
    while True:
        fenetre.fill(jaune)
        dessiner(fenetre, black, [170, 379+84*position_bouton, 300, 64], 3)
        dessiner(fenetre, black, [170, 379, 300, 64], 1)
        dessiner(fenetre, black, [170, 463, 300, 64], 1)
        dessiner(fenetre, black, [170, 547, 300, 64], 1)
        fenetre.blit(titre1, titre1.get_rect(center=(fenetre.get_width()/2, 60)))
        fenetre.blit(text1, text1.get_rect(center=(fenetre.get_width()/2, 411)))
        fenetre.blit(text2, text2.get_rect(center=(fenetre.get_width()/2, 495)))
        fenetre.blit(text3, text3.get_rect(center=(fenetre.get_width()/2, 579)))

        for event in pygame.event.get():
            if event.type == QUIT:
                return "End"
            if event.type == KEYDOWN:
                if event.key == K_RETURN:
                    if position_bouton == 0:
                        return None
                    if position_bouton == 1:
                        return start_menu(fenetre,joueur)
                    if position_bouton == 2:
                        return "End"
                if event.key == K_ESCAPE:
                    pygame.mixer.music.load('data/compo 1.wav')
                    pygame.mixer.music.play(-1)
                    return None
                if event.key == K_DOWN or event.key == K_RIGHT:
                    if position_bouton <2:
                        position_bouton += 1
                if event.key == K_UP or event.key == K_LEFT:
                    if position_bouton >0:
                        position_bouton -= 1
        pygame.display.flip()
```

Catre/objet.py

```
# -*- coding: utf-8 -*-
import pygame
from pygame.locals import*

from cases import*
from carte import*
import sys
sys.path.append("..")
import combat.combat as combat
import subprocess

class objet:
    def __init__(self, carte, x, y):
        if carte.matrice_objet[x][y] == None and carte.matrice_case[x][y]:
            self.carte = carte
            carte.placer_obj(self, x, y)
            self.posx = x
            self.posy = y
            self.rep = "X"
        else:
            self.carte = None
            self.posx = 0
            self.posy = 0
            self.rep = "X"

    def interagir(self):
        pass
    def effacer(self):
        self.carte.effacer_obj(self.posx, self.posy)

class obj_boug(objet):
    def __init__(self, carte, x, y):
        objet.__init__(self, carte, x, y)
        self.direction = "gauche"
        self.dict_dir = {"gauche":(-1, 0), "droite":(1, 0), "haut":(0, -1), "bas":(0, 1)}
        self.image = pygame.image.load("data/perso.png").convert_alpha()
    def ch_direction(self, direction):
        if direction in ["haut", "bas", "gauche", "droite"]:
            self.direction = direction
            return True
        else:
            return False

    def avancer(self):
        if self.carte.est_marchable(self.posx+self.dict_dir[self.direction][0], self.posy+self.dict_dir[self.direction][1]) ==
True:
            if self.carte.obj_vide(self.posx+self.dict_dir[self.direction][0], self.posy+self.dict_dir[self.direction][1]) ==
True:

                self.carte.deplacer((self.posx, self.posy), (self.posx+self.dict_dir[self.direction][0],
self.posy+self.dict_dir[self.direction][1]))
                self.posx = self.posx+self.dict_dir[self.direction][0]
                self.posy = self.posy+self.dict_dir[self.direction][1]



class perso(obj_boug):
    def __init__(self, carte, x, y, images):
        obj_boug.__init__(self, carte , x, y)
        self.dict_images = {"gauche": images[0],"droite": images[1],"haut":images[2],"bas":images[3]}
        self.stade_animation = 0
```

```
        self.inc = 0
    def avancer(self):
        if self.carte.est_marchable(self.posx+self.dict_dir[self.direction][0], self.posy+self.dict_dir[self.direction][1]) ==
True:
            if self.carte.obj_vide(self.posx+self.dict_dir[self.direction][0], self.posy+self.dict_dir[self.direction][1]) ==
True:

                self.carte.deplacer((self.posx, self.posy), (self.posx+self.dict_dir[self.direction][0],
self.posy+self.dict_dir[self.direction][1]))
                self.posx = self.posx+self.dict_dir[self.direction][0]
                self.posy = self.posy+self.dict_dir[self.direction][1]
                self.stade_animation = 1
            else:
                self.carte.matrice_objet[self.posx+self.dict_dir[self.direction][0]][self.posy+self.dict_dir[self.direction]
[1]].interagir()

    def ch_direction(self, direction):
        if direction in ["haut", "bas", "gauche", "droite"]:
            self.direction = direction
            #self.image = self.dict_images[self.direction]
            return True
        else:
            return False

    def step(self):
        if self.inc == 2:
            if self.stade_animation !=0:
                if self.stade_animation == 8:
                    self.stade_animation = 0
                else:
                    self.stade_animation +=1
            self.inc = 0
        self.inc+=1
class arbre(objet):
    def __init__(self, carte, x, y):
        objet.__init__(self, carte, x, y)
        self.image = pygame.image.load("data/sprite_01.png")


class ennemi(obj_boug):
    def __init__(self, carte, x, y, ls_ennemi):
        objet.__init__(self, carte, x, y)
        self.ls_ennemi = ls_ennemi
        self.image = ls_ennemi[0].img

    def interagir(self):
        print("combat")
        self.carte.combat = [True, self]


class coffre(objet):
    def __init__(self, carte, x, y, contenu):
        objet.__init__(self, carte, x, y)
        self.contenu = contenu
        self.image_complet = pygame.image.load("data/chest2.png")
        self.image_fermee = self.image_complet.subsurface((0, 0, 32, 32))
        self.image_ouverte = self.image_complet.subsurface((32, 0, 32, 32))
        self.image = self.image_fermee

    def interagir(self):
        for potion in self.contenu:
            exec("self.carte.joueur."+potion+" += 1")
        self.contenu = []
```

```
        self.image = self.image_ouverte

class porte(objet):
    def __init__(self, carte, x, y):
        objet.__init__(self, carte, x, y)
        self.image = pygame.image.load("data/sprite_07.png")

    def interagir(self):
        self.effacer()

class porte_boss(objet):
    def __init__(self, carte, x, y):
        objet.__init__(self, carte, x, y)
        self.image = pygame.image.load("data/castledoors32.png")

    def interagir(self):
        self.carte.changer_carte()

class boss(obj_boug):
    def __init__(self, carte, x, y, ls_ennemi):
         objet.__init__(self, carte, x, y)
         self.ls_ennemi = ls_ennemi
         self.image = ls_ennemi[0].img

    def interagir(self):
         self.carte.combat = [True, self]


class coffre_boss(objet):
    def __init__(self, carte, x, y):
        objet.__init__(self, carte, x, y)
        self.image_complet = pygame.image.load("data/chest2.png")
        self.image_fermee = self.image_complet.subsurface((0, 0, 32, 32))
        self.image_ouverte = self.image_complet.subsurface((32, 0, 32, 32))
        self.image = self.image_fermee
    def interagir(self):
        self.carte.joueur.upgrade(self.carte.niv_carte+1)
        self.image = self.image_ouverte


class Stalagmites(objet):
     def __init__(self, carte, x, y):
         objet.__init__(self, carte, x, y)
         self.image = pygame.image.load("data/sprite_00.png")

class pnj(objet):
     def __init__(self, carte, x, y):
         objet.__init__(self, carte, x, y)
         self.image = pygame.image.load("data/Sprite_RC.png")

     def interagir(self):
         subprocess.call("start python RC/ElProyectoRubiksCube.py")
```

Catre/script.py

```
# -*- coding: utf-8 -*-
"""
Created on Tue Apr 11 22:38:18 2017

@author: nassim
"""
#import sys
#print u"dépasse".encode( sys.stdout.encoding )

with open("data/script.txt","r") as fichier_script:
    texte = []
    for i in range(5):
        texte.append(fichier_script.readline())

print texte


class script1:
    '''une ligne fait exactement 39 caractères avec les paramètres de bases des caracteres'''
    def __init__(self, texte):
        self.ls_page = texte

if __name__ == "__main__":
    sc = script1(texte)
    for a in range (len(sc.ls_page)):
        print sc.ls_page[a]

#============================================================================
#         self.page2 = u"Vous êtes un ancien membre d’une guilde".encode('utf-8')
#         self.page3 = "bla bla bla bla bla"
#         self.page4 = "bla bla bla bla bla"
#         self.page4 = "bla bla bla bla bla"
#============================================================================

'''
Vous êtes un ancien membre d’une guilde
nommé les Pac’s Men dont les membres ont
été séparés après la défaite de l’armée
de Dénoma face à celle de Brigos. Pour
échapper à l’esclavage ou à l’exécution
des vaincus vous avez du vous enfuir,
blessé durant le combat. Vous avez du
donner l’équipement qu’il vous restait
pour vous faire soigner et survivre.
Depuis vous travaillez en tant que
mercenaire tout en essayant de
retrouver les autres membres de la
guilde.
Vous êtes actuellement dans une taverne
miteuse dans la ville d’Esthar. Ville
corrompue par la cruauté d’Empyrion
célèbre gouverneur de Brigos mis en
place après la prise de la ville qui
torture lui-même les personnes
suspectées d’aider Dénoma.
'''
```

Catre/scriptpa.py

```
import pygame
from pygame import *

pygame.init()
fenetre=pygame.display.set_mode((640,640))

pygame.font.init()
noir=(0,0,0)
bleu=(0,0,160)
blanc=(0xFF, 0xFF, 0xFF)
font = pygame.font.Font("data/Olondon_.otf",25)
font2=pygame.font.SysFont('Calibri',20,True)
font3=pygame.font.SysFont('Calibri',20,True,True)


image=pygame.image.load("data/livreintroisn.png")
image1=pygame.image.load("data/imagepage1isn.jpg")
image3=pygame.image.load("data/taverne.jpg")
image2=pygame.image.load("data/page2isn.jpg")
noire=pygame.image.load("data/noir.png")
image4=pygame.image.load("data/devanttaverne.jpg")
image5=pygame.image.load("data/femmeauberge.jpg")
image6=pygame.image.load("data/hood.jpg")
image7=pygame.image.load("data/blancdialogue.jpg")
image8=pygame.image.load("data/plaine.png")
image9=pygame.image.load("data/clavier.jpg")

def script_pa(fenetre):
    fenetre.fill(noir)
    continuer= True
    page= 1
    while continuer:
        for event in pygame.event.get():
            if event.type == QUIT:
                continuer = False
            if event.type == KEYDOWN:
                if event.key == K_RIGHT:
                    page +=1
                if event.key == K_LEFT:
                    page -=1
                if event.key == K_ESCAPE:
                    continuer = False
        if page>10:
            page=10
        if page<1:
            page=1

        if page== 1:
                fenetre.blit(noire,[0,0])
                fenetre.blit(image,[0,110])
                text1g1 = font.render("Vous etes un ancien membre ",True,noir)
                fenetre.blit(text1g1,[45,145])
                text1g2 = font.render("d'une guilde nomme les Pac's ",True,noir)
                fenetre.blit(text1g2,[45,165])
                text1g3 = font.render("Men dont les membres ont ",True,noir)
                fenetre.blit(text1g3,[45,185])
                text1g4 = font.render("ete separes apres la defaite ",True,noir)
                fenetre.blit(text1g4,[45,205])
                text1g5 = font.render("de l'armee de Denoma face ",True,noir)
                fenetre.blit(text1g5,[45,225])
                text1g6 = font.render("a celle de Brigos. Pour ",True,noir)
```

```
        fenetre.blit(text1g6,[45,245])
        text1g7 = font.render("echapper a l'esclavage ou a ",True,noir)
        fenetre.blit(text1g7,[45,265])
        text1g8 = font.render("l'execution des vaincus vous  ",True,noir)
        fenetre.blit(text1g8,[45,285])
        text1g9 = font.render("avez du vous enfuir, blesse ",True,noir)
        fenetre.blit(text1g9,[45,305])
        text1g10 = font.render("durant le combat. Vous avez",True,noir)
        fenetre.blit(text1g10,[45,325])
        text1g11 = font.render("du donner l'equipement qu'il ",True,noir)
        fenetre.blit(text1g11,[45,345])
        text1g12 = font.render("vous restait pour vous faire",True,noir)
        fenetre.blit(text1g12,[45,365])
        text1g13 = font.render("soigner et survivre. Depuis",True,noir)
        fenetre.blit(text1g13,[45,385])
        text1g14 = font.render("vous travaillez en tant que ",True,noir)
        fenetre.blit(text1g14,[45,405])
        text1g15 = font.render("Page 1 ",True,noir)
        fenetre.blit(text1g15,[150,445])

        text1d1 = font.render("Bataille de Fraktia",True,noir)
        fenetre.blit(text1d1,[375,175])
        text1d2 = font.render("Page 2 ",True,noir)
        fenetre.blit(text1d2,[450,445])
        fenetre.blit(image1,[365,205])



if page== 2:
        fenetre.blit(noire,[0,0])
        fenetre.blit(image,[0,110])

        text2g1 = font.render("Ville d'Esthar",True,noir)
        fenetre.blit(text2g1,[100,175])
        fenetre.blit(image2,[35,205])
        text2g2 = font.render("Page 3 ",True,noir)
        fenetre.blit(text2g2,[150,445])

        text2d1 = font.render("mercenaire tout en essayant ",True,noir)
        fenetre.blit(text2d1,[350,145])
        text2d2 = font.render("de retrouver les autres ",True,noir)
        fenetre.blit(text2d2,[350,165])
        text2d3 = font.render("membres de la guilde. Vous",True,noir)
        fenetre.blit(text2d3,[350,185])
        text2d4 = font.render("etes actuellement dans une ",True,noir)
        fenetre.blit(text2d4,[350,205])
        text2d5 = font.render("taverne miteuse dans la ville",True,noir)
        fenetre.blit(text2d5,[350,225])
        text2d6 = font.render("d'Esthar. Ville corrompue",True,noir)
        fenetre.blit(text2d6,[350,245])
        text2d7 = font.render("par la cruaute d'Empyrion ",True,noir)
        fenetre.blit(text2d7,[350,265])
        text2d8 = font.render("celebre gouverneur de Brigos",True,noir)
        fenetre.blit(text2d8,[350,285])
        text2d9 = font.render("mis en place apres la prise",True,noir)
        fenetre.blit(text2d9,[350,305])
        text2d10 = font.render("de la ville qui torture lui-",True,noir)
        fenetre.blit(text2d10,[350,325])
        text2d11 = font.render("meme les personnes",True,noir)
        fenetre.blit(text2d11,[350,345])
        text2d12 = font.render("suspectees d'aider Denoma.",True,noir)
        fenetre.blit(text2d12,[350,365])
```

```
        text2d13 = font.render("Page 4 ",True,noir)
        fenetre.blit(text2d13,[450,445])


if page== 3:
        fenetre.blit(noire,[0,0])
        fenetre.blit(image,[0,110])
        text3g1 = font.render("Une jeune femme les larmes ",True,noir)
        fenetre.blit(text3g1,[45,145])
        text3g2 = font.render("aux yeux vient demander de",True,noir)
        fenetre.blit(text3g2,[45,165])
        text3g3 = font.render("l'aide dans la taverne, son",True,noir)
        fenetre.blit(text3g3,[45,185])
        text3g4 = font.render("ami Scrap, un petit golem ",True,noir)
        fenetre.blit(text3g4,[45,205])
        text3g5 = font.render("de fer ne grace a la magie et ",True,noir)
        fenetre.blit(text3g5,[45,225])
        text3g6 = font.render("possedant des capacites a ete ",True,noir)
        fenetre.blit(text3g6,[45,245])
        text3g7 = font.render("capture par les sbires de ",True,noir)
        fenetre.blit(text3g7,[45,265])
        text3g8 = font.render("Thunderlord, un savant fou",True,noir)
        fenetre.blit(text3g8,[45,285])
        text3g9 = font.render("connu pour creer des mon-",True,noir)
        fenetre.blit(text3g9,[45,305])
        text3g10 = font.render("truosites. Les autres ",True,noir)
        fenetre.blit(text3g10,[45,325])
        text3g11 = font.render("mercenaires l'ecoute a peine ",True,noir)
        fenetre.blit(text3g11,[45,345])
        text3g12 = font.render("mais a un moment elle parle",True,noir)
        fenetre.blit(text3g12,[45,365])
        text3g13 = font.render("d'un chateau, celui de Gorfath",True,noir)
        fenetre.blit(text3g13,[45,385])
        text3g14 = font.render("dont elle a entendu les sbires ",True,noir)
        fenetre.blit(text3g14,[45,405])
        text3g15 = font.render("Page 5 ",True,noir)
        fenetre.blit(text3g15,[150,445])

        text3d1 = font.render("parler. Entendant cela vous ",True,noir)
        fenetre.blit(text3d1,[350,145])
        text3d2 = font.render("vous levez et allez la voir.",True,noir)
        fenetre.blit(text3d2,[350,165])

        text3d4 = font.render("Taverne des Cavaliers",True,noir)
        fenetre.blit(text3d4,[355,195])
        text3d5 = font.render("Page 6 ",True,noir)
        fenetre.blit(text3d5,[450,445])
        fenetre.blit(image4,[365,225])


if page== 4:
    fenetre.blit(image3,[0,0])
    fenetre.blit(image7,[0,490])
    fenetre.blit(image5,[520,490])
    fenetre.blit(image6,[0,490])
    text4d1 = font2.render("-Savez vous pourquoi il a capture ",True,noir)
    fenetre.blit(text4d1,[150,495])
    text4d2 = font2.render("votre ami ?",True,noir)
    fenetre.blit(text4d2,[150,515])
    text4d3 = font2.render("-Les hommes m'ont dit que",True,bleu)
    fenetre.blit(text4d3,[235,545])
```

```
    text4d4 = font2.render("Thunderlord voulait recuperer les",True,bleu)
    fenetre.blit(text4d4,[235,565])
    text4d5 = font2.render("pouvoirs de Scrap pour pouvoir",True,bleu)
    fenetre.blit(text4d5,[235,585])
    text4d6 = font2.render("donner vie a ses creatures.",True,bleu)
    fenetre.blit(text4d6,[235,605])

if page== 5:
    fenetre.blit(image3,[0,0])
    fenetre.blit(image7,[0,490])
    fenetre.blit(image5,[520,490])
    fenetre.blit(image6,[0,490])
    text5d1 = font2.render("Vous avez parle du chateau de Gorfath ",True,noir)
    fenetre.blit(text5d1,[150,495])
    text4d2 = font2.render("que savez-vous de ce chateau ? ",True,noir)
    fenetre.blit(text4d2,[150,515])
    text4d3 = font2.render("-J'ai entendu les sbires parler",True,bleu)
    fenetre.blit(text4d3,[235,545])
    text4d4 = font2.render("d'une caverne sur le chemin.",True,bleu)
    fenetre.blit(text4d4,[235,565])
    text4d5 = font2.render("Pouvez-vous m'aider ? ",True,bleu)
    fenetre.blit(text4d5,[235,585])

if page== 6:
    fenetre.blit(image3,[0,0])
    fenetre.blit(image7,[0,490])
    fenetre.blit(image5,[520,490])
    fenetre.blit(image6,[0,490])
    text6d1 = font3.render("Vous connaissez ce chateau souterrain",True,noir)
    fenetre.blit(text6d1,[150,495])
    text6d2 = font3.render("vous y etes deja aller avec votre guilde.",True,noir)
    fenetre.blit(text6d2,[150,515])
    text6d3 = font3.render("Peut-etre certains y sont retournes.",True,noir)
    fenetre.blit(text6d3,[150,535])
    text6d4 = font3.render("Vous decidez de tenter l'aventure. ",True,noir)
    fenetre.blit(text6d4,[150,555])

if page== 7:
    fenetre.blit(image3,[0,0])
    fenetre.blit(image7,[0,490])
    fenetre.blit(image5,[520,490])
    fenetre.blit(image6,[0,490])
    text7d1 = font2.render("-Je vais y aller. ",True,noir)
    fenetre.blit(text7d1,[150,495])
    text7d2 = font2.render("-Merci beaucoup ! Je vais vous",True,bleu)
    fenetre.blit(text7d2,[235,525])
    text7d3 = font2.render("vous donner votre argent.",True,bleu)
    fenetre.blit(text7d3,[235,545])
    text7d4 = font2.render("-Vous me paierez si je reviens ",True,noir)
    fenetre.blit(text7d4,[150,575])
    text7d5 = font2.render("avec Scrap.",True,noir)
    fenetre.blit(text7d5,[150,595])

if page== 8:
    fenetre.blit(image3,[0,0])
    fenetre.blit(image7,[0,490])
    fenetre.blit(image5,[520,490])
    fenetre.blit(image6,[0,490])
    text8d1 = font2.render("-D'accord si vous le voulez.",True,bleu)
    fenetre.blit(text8d1,[235,495])
    text8d2 = font2.render("-Je partirai demain a l'aube. ",True,noir)
    fenetre.blit(text8d2,[150,525])
```

```
        if page== 9:
                fenetre.blit(noire,[0,0])
                fenetre.blit(image,[0,110])
                text9g1 = font.render("Vous vous reveillez le",True,noir)
                fenetre.blit(text9g1,[45,145])
                text9g2 = font.render("lendemain et vous allez aux",True,noir)
                fenetre.blit(text9g2,[45,165])
                text9g3 = font.render("portes de la ville. Vous",True,noir)
                fenetre.blit(text9g3,[45,185])
                text9g4 = font.render("partez vers le Nord en",True,noir)
                fenetre.blit(text9g4,[45,205])
                text9g5 = font.render("direction de la caverne par la ",True,noir)
                fenetre.blit(text9g5,[45,225])
                text9g6 = font.render("plaine de Briliance, le chemin",True,noir)
                fenetre.blit(text9g6,[45,245])
                text9g7 = font.render("sera long et probablement",True,noir)
                fenetre.blit(text9g7,[45,265])
                text9g8 = font.render("seme d'ennemis ...",True,noir)
                fenetre.blit(text9g8,[45,285])

                fenetre.blit(image8,[365,245])
                text9d9 = font.render("Plaine de Briliance",True,noir)
                fenetre.blit(text9d9,[375,215])

                text3g15 = font.render("Page 7 ",True,noir)
                fenetre.blit(text3g15,[150,445])

                text3d5 = font.render("Page 8 ",True,noir)
                fenetre.blit(text3d5,[450,445])

        if page== 10:
                fenetre.blit(noire,[0,0])
                fenetre.blit(image,[0,110])

                text10g1 = font.render("Controles",True,noir)
                fenetre.blit(text10g1,[125,145])
                text10g2 = font.render("Deplacement",True,noir)
                fenetre.blit(text10g2,[45,175])
                fenetre.blit(image9,[35,205])


                text10d1 = font.render("Inventaire: I ",True,noir)
                fenetre.blit(text10d1,[350,165])
                text10d2 = font.render("Retour: Tab",True,noir)
                fenetre.blit(text10d2,[350,195])
                text10d3 = font.render("Pause: Echap",True,noir)
                fenetre.blit(text10d3,[350,225])
                text10d4 = font.render("Selectionner: Entree ",True,noir)
                fenetre.blit(text10d4,[350,255])


        pygame.display.flip()
if __name__ == "__main__":
    script_pa(fenetre)
```

Combat/armes.py

```
# -*- coding: utf-8 -*-

class arme:
    def __init__(self):
        self.atk=0
        self.mag=0

class Livre(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Livre Poussiéreux"
        self.atk=0
        self.mag=125

class Parchemin(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Parchemin des Arcanes"
        self.atk=0
        self.mag=150

class Grimoire(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Grimoire de l'Archimage"
        self.atk=0
        self.mag=200

class Dagues(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Dagues fendues"
        self.atk=100
        self.mag=0

class LamesD(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Lames Dansantes"
        self.atk=150
        self.mag=0

class Hachettes(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Hachettes Sanguinaires"
        self.atk=175
        self.mag=0

class Epee(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Épée Rouillée"
        self.atk=75
        self.mag=0

class Hache(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Hache de Guerre"
        self.atk=150
```

```
        self.mag=0

class EpeeRL(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Épée Runique Légendaire"
        self.atk=150
        self.mag=50

class LamesE(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Lames Empoisonnées"
        self.atk=0
        self.mag=100

class Fouet(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Fouet de Soumission"
        self.atk=0
        self.mag=150

class Crescent(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Crescent blade"
        self.atk=0
        self.mag=150

class ArcL(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Arc long"
        self.atk=125
        self.mag=0

class Arbalete(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Arbalète de tueur d'ours"
        self.atk=150
        self.mag=0

class ArcA(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Arc d'Artémis"
        self.atk=200
        self.mag=0

class Baton(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Bâton tordu"
        self.atk=0
        self.mag=75

class Orbe(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Orbe de régénération"
        self.atk=0
```

```
        self.mag=125

class Sceptre(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Sceptre de l'Archange"
        self.atk=25
        self.mag=150

class RubiksCube(arme):
    def __init__(self):
        arme.__init__(self)
        self.nom="Rubik's Cube"
        self.atk=20
        self.mag=0
```

Combat/armure.py

```
# -*- coding: utf-8 -*-

class armure:
    def __init__(self):
        self.defen=0
        self.res=0



class Robe(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Robe en coton"
        self.defen=0
        self.res=20

class TuniqueI(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Tunique d'incantateur"
        self.defen=5
        self.res=25

class TogeI(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Toge de l'Immortel"
        self.defen=10
        self.res=30

class Tenue(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Tenue d'éclaireur"
        self.defen=10
        self.res=0

class Manteau(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Manteau de loup solitaire"
        self.defen=20
        self.res=5

class Armureco(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Armure en cobalt éthérée"
        self.defen=30
        self.res=10

class Armurecu(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Armure en cuir"
        self.defen=20
        self.res=5

class Plastron(armure):
    def __init__(self):
        armure.__init__(self)
```

```
        self.nom="Plastron de chevalier"
        self.defen=30
        self.res=10

class Cuirasse(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Cuirasse en peau de dragon"
        self.defen=40
        self.res=15

class Gilet(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Gilet d'amateur"
        self.defen=0
        self.res=10

class Veste(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Veste de tueur nocturne"
        self.defen=5
        self.res=20

class Armurean(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Armure antique de démon"
        self.defen=10
        self.res=30

class Cape(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Cape en tissu"
        self.defen=10
        self.res=0

class Justaucorps(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Justaucorps en écailles"
        self.defen=15
        self.res=5

class Cote(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Côte de maille en mithril"
        self.defen=20
        self.res=10

class TuniqueH(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="tunique d'Herbaliste"
        self.defen=5
        self.res=10

class TogeA(armure):
    def __init__(self):
        armure.__init__(self)
```

```
        self.nom="Toge d'amis de la nature"
        self.defen=10
        self.res=15

class Corset(armure):
    def __init__(self):
        armure.__init__(self)
        self.nom="Corset de déesse"
        self.defen=15
        self.res=20
```

Combat/combat.py

```
# -*- coding: utf-8 -*-
import pygame,random
from pygame.locals import *
pygame.init()
noir=(0,0,0)
blanc=(0xFF, 0xFF, 0xFF)
font = pygame.font.SysFont('Calibri', 25, True, False)
text_attaque = font.render("attaque",True,noir)
text_potion = font.render("potion",True,noir)
text_fuite = font.render("fuite",True,noir)



def combat_start(joueur,participant):
    '''
        fonction principale du combat
        parametre: class joueur , liste des participants

        appelle la fonction secondaire en ajoutant joueur a la liste des participant
        trie les participants au combat selon leur vitesse
    '''
    participant.append(joueur)
    participant.sort(key=lambda v: v.vit)
    participant.reverse()
    statut = combat_attaque(participant)
    participant.reverse()
    participant.remove(joueur)
    return statut


def combat_attaque(participant_vit):
    '''
        attaque de tous les participants dans l'ordre de vitesse + test de mort
    '''
    for i in range (len(participant_vit)):
        participant_vit[i].passif_def(participant_vit)
        if participant_vit[i].pv == 0:
            if participant_vit[i].jouable==True:
                return "Mort joueur"
            else:
                pass

        else:
            if participant_vit[i].jouable==True:
                if participant_vit[i].type_action == "potion":
                    participant_vit[i].popo_actif()
                elif participant_vit[i].type_action == "attaque":
                    participant_vit[i].attaque()
            else:
                participant_vit[i].cible_def(participant_vit)
                participant_vit[i].attaque()
            participant_vit[i].popo_def()
            participant_vit[i].effet_def()




def affiche_combat(fenetre,joueur,ennemi):
```

```
pygame.mixer.music.load('data/05 - Le Voyage de Basile.mp3')
pygame.mixer.music.play(-1)
nb_ennemis = ennemi[0].nombre
continuer = 1
position_bouton=1
tour=1
text_ennemi = font.render(ennemi[0].nom,True,noir)
background_image = pygame.image.load("data/fondcombatmap1.jpg").convert()

while continuer == 1:
    if joueur.pv == 0:
        return "Mort joueur"

    text_tour = font.render("tour "+str(tour),True,noir)
    text_pv_joueur = font.render("Pv : "+str(joueur.pv)+"/"+str(joueur.pv_max),True,noir)

    fenetre.blit(background_image, [0, 0])
    pygame.draw.rect(fenetre, blanc, [0, 590, 640, 50])
    fenetre.blit(joueur.img, joueur.img.get_rect(center=(100, 300)))

    if nb_ennemis == 1:

        text_pv_ennemi_0 = font.render("Pv : "+str(ennemi[0].pv)+"/"+str(ennemi[0].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_0, [500, 350])
        fenetre.blit(ennemi[0].img_combat, ennemi[0].img_combat.get_rect(center=(500, 300)))

    if nb_ennemis == 2:

        text_pv_ennemi_0 = font.render("Pv : "+str(ennemi[0].pv)+"/"+str(ennemi[0].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_0, [500, 250])
        fenetre.blit(ennemi[1].img_combat, ennemi[1].img_combat.get_rect(center=(500, 200)))

        text_pv_ennemi_1 = font.render("Pv : "+str(ennemi[1].pv)+"/"+str(ennemi[1].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_1, [500, 450])
        fenetre.blit(ennemi[1].img_combat, ennemi[1].img_combat.get_rect(center=(500, 400)))

    if nb_ennemis == 3 :

        text_pv_ennemi_0 = font.render("Pv : "+str(ennemi[0].pv)+"/"+str(ennemi[0].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_0, [500, 200])
        fenetre.blit(ennemi[0].img_combat, ennemi[0].img_combat.get_rect(center=(500, 150)))

        text_pv_ennemi_1 = font.render("Pv : "+str(ennemi[1].pv)+"/"+str(ennemi[1].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_1, [500, 350])
        fenetre.blit(ennemi[1].img_combat, ennemi[1].img_combat.get_rect(center=(500, 300)))

        text_pv_ennemi_2 = font.render("Pv : "+str(ennemi[2].pv)+"/"+str(ennemi[2].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_2, [500, 500])
        fenetre.blit(ennemi[2].img_combat, ennemi[2].img_combat.get_rect(center=(500, 450)))

    if nb_ennemis == 4 :

        text_pv_ennemi_0 = font.render("Pv : "+str(ennemi[0].pv)+"/"+str(ennemi[0].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_0, [500, 170])
        fenetre.blit(ennemi[0].img_combat, ennemi[0].img_combat.get_rect(center=(500, 120)))

        text_pv_ennemi_1 = font.render("Pv : "+str(ennemi[1].pv)+"/"+str(ennemi[1].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_1, [500, 290])
        fenetre.blit(ennemi[1].img_combat, ennemi[1].img_combat.get_rect(center=(500, 240)))

        text_pv_ennemi_2 = font.render("Pv : "+str(ennemi[2].pv)+"/"+str(ennemi[2].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_2, [500, 410])
        fenetre.blit(ennemi[2].img_combat, ennemi[2].img_combat.get_rect(center=(500, 360)))
```

```
    text_pv_ennemi_3 = font.render("Pv : "+str(ennemi[3].pv)+"/"+str(ennemi[3].pv_max),True,noir)
    fenetre.blit(text_pv_ennemi_3, [500, 530])
    fenetre.blit(ennemi[3].img_combat, ennemi[3].img_combat.get_rect(center=(500, 480)))

if position_bouton == 1:
    pygame.draw.rect(fenetre, noir, [0, 590, 100, 50], 3)

if position_bouton == 2:
    pygame.draw.rect(fenetre, noir, [100, 590, 110, 50], 3)

if position_bouton == 3:
    pygame.draw.rect(fenetre, noir, [210, 590, 190, 50], 3)

fenetre.blit(text_attaque, [10, 600])
fenetre.blit(text_potion, [120, 600])
fenetre.blit(text_fuite, [230, 600])
fenetre.blit(text_ennemi, [430, 600])
fenetre.blit(text_tour, [10, 10])
fenetre.blit(text_pv_joueur, [10, 500])


pygame.draw.rect(fenetre, noir, [0, 590, 640, 50], 1)
pygame.draw.line(fenetre, noir, [100, 590], [100, 640], 1)
pygame.draw.line(fenetre, noir, [210, 590], [210, 640], 1)
pygame.draw.line(fenetre, noir, [400, 590], [400, 640], 1)
pygame.display.flip()

pv_toto = 0
for i in range (nb_ennemis):
    pv_toto += ennemi[i].pv
if pv_toto == 0:
    if random.randrange(0,1) == 1:
        joueur.potionvie1+=random.randrange(0,3)
    if random.randrange(0,5) == 1:
        joueur.potionvie2 += 1
    if random.randrange(0,10) == 1:
        joueur.potionvie3 += 1
    if random.randrange(0,5) == 1:
        joueur.potionarmure1 += 1
    if random.randrange(0,10) == 1:
        joueur.potionarmure2 += 1
    if random.randrange(0,20) == 1:
        joueur.potionarmure3 += 1
    if random.randrange(0,5) == 1:
        joueur.potionforce1 += 1
    if random.randrange(0,10) == 1:
        joueur.potionforce2 += 1
    if random.randrange(0,20) == 1:
        joueur.potionforce3 += 1
    if random.randrange(0,5) == 1:
        joueur.potioncritique1 += 1
    if random.randrange(0,10) == 1:
        joueur.potioncritique2 += 1
    if random.randrange(0,20) == 1:
        joueur.potioncritique3 += 1
    if random.randrange(0,10) == 1:
        joueur.potionvitesse += 1
    if random.randrange(0,10) == 1:
        joueur.potionprecision += 1
    pygame.mixer.music.load('data/compo 1.wav')
    pygame.mixer.music.play(-1)
    return "fin"
```

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

            if event.type == QUIT:
                continuer = 0

            if event.type == KEYDOWN:

                if event.key == K_RIGHT:
                    if position_bouton <3:
                        position_bouton += 1

                if event.key == K_LEFT:
                    if position_bouton >1:
                        position_bouton -= 1

                if event.key == K_RETURN:

                    if position_bouton == 1:
                        joueur.type_action = "attaque"
                        info = attaque_type(fenetre,joueur,ennemi,tour)
                        if info == "End":
                            continuer = 0
                        if info == "Mort joueur":
                            return "Mort joueur"
                        if info == "Next":
                            tour+=1

                    if position_bouton == 2:
                        joueur.type_action = "potion"
                        info = potion_type(fenetre,joueur,ennemi)
                        if info == "End":
                            continuer = 0
                        if info == "Next":
                            tour+=1

                    if position_bouton == 3:
                        pygame.mixer.music.load('data/compo 1.wav')
                        pygame.mixer.music.play(-1)
                        return "Fuite"



def select_ennemi(fenetre,joueur,ennemi,tour):
    nb_ennemis = ennemi[0].nombre
    continuer = True
    position_bouton= 0

    text_ennemi = font.render(ennemi[0].nom,True,noir)
    background_image = pygame.image.load("data/fondcombatmap1.jpg").convert()
    fleche = pygame.image.load("data/fleche.png").convert_alpha()

    pygame.draw.rect(fenetre, blanc, [0, 585, 640, 55])
    fenetre.blit(text_ennemi, [430, 600])


    if nb_ennemis == 1:
        while continuer:

            text_tour = font.render("tour "+str(tour),True,noir)
            text_pv_joueur = font.render("Pv : "+str(joueur.pv)+"/"+str(joueur.pv_max),True,noir)
```

```
        fenetre.blit(background_image, [0, 0])
        fenetre.blit(text_tour, [10, 10])
        fenetre.blit(text_pv_joueur, [10, 500])
        fenetre.blit(joueur.img, joueur.img.get_rect(center=(100, 300)))

        text_pv_ennemi_0 = font.render("Pv : "+str(ennemi[0].pv)+"/"+str(ennemi[0].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_0, [500, 350])
        fenetre.blit(ennemi[0].img_combat, ennemi[0].img_combat.get_rect(center=(500, 300)))

        if position_bouton == 0:
            fenetre.blit(fleche, fleche.get_rect(center=(350, 300)))
        pygame.display.flip()
        for event in pygame.event.get():

            if event.type == QUIT:
                return "End"

            if event.type == KEYDOWN:

                if event.key == K_TAB:
                    continuer = False

                if event.key == K_RETURN:
                    joueur.cible = ennemi[position_bouton]
                    anim_joueur(fenetre,joueur)
                    if combat_start(joueur,ennemi) == "Mort joueur":
                        return "Mort joueur"
                    return "next"

    if nb_ennemis == 2:
        while continuer:

            text_tour = font.render("tour "+str(tour),True,noir)
            text_pv_joueur = font.render("Pv : "+str(joueur.pv)+"/"+str(joueur.pv_max),True,noir)

            fenetre.blit(background_image, [0, 0])
            fenetre.blit(text_tour, [10, 10])
            fenetre.blit(text_pv_joueur, [10, 500])
            fenetre.blit(joueur.img, joueur.img.get_rect(center=(100, 300)))

            text_pv_ennemi_0 = font.render("Pv : "+str(ennemi[0].pv)+"/"+str(ennemi[0].pv_max),True,noir)
            fenetre.blit(text_pv_ennemi_0, [500, 250])
            fenetre.blit(ennemi[1].img_combat, ennemi[1].img_combat.get_rect(center=(500, 200)))

            text_pv_ennemi_1 = font.render("Pv : "+str(ennemi[1].pv)+"/"+str(ennemi[1].pv_max),True,noir)
            fenetre.blit(text_pv_ennemi_1, [500, 450])
            fenetre.blit(ennemi[1].img_combat, ennemi[1].img_combat.get_rect(center=(500, 400)))

            if position_bouton == 0:
                fenetre.blit(fleche, fleche.get_rect(center=(350, 200)))
            if position_bouton == 1:
                fenetre.blit(fleche, fleche.get_rect(center=(350, 400)))

            for event in pygame.event.get():

                if event.type == QUIT:
                    return "End"

                if event.type == KEYDOWN:

                    if event.key == K_DOWN:
                        if position_bouton <1:
```

```
                position_bouton += 1

            if event.key == K_UP:
                if position_bouton >0:
                    position_bouton -= 1

            if event.key == K_TAB:
                continuer = False

            if event.key == K_RETURN:
                joueur.cible = ennemi[position_bouton]
                anim_joueur(fenetre,joueur)
                if combat_start(joueur,ennemi) == "Mort joueur":
                    return "Mort joueur"
                return "next"

    pygame.display.flip()
if nb_ennemis == 3:
    while continuer:

        text_tour = font.render("tour "+str(tour),True,noir)
        text_pv_joueur = font.render("Pv : "+str(joueur.pv)+"/"+str(joueur.pv_max),True,noir)

        fenetre.blit(background_image, [0, 0])
        fenetre.blit(text_tour, [10, 10])
        fenetre.blit(text_pv_joueur, [10, 500])
        fenetre.blit(joueur.img, joueur.img.get_rect(center=(100, 300)))

        text_pv_ennemi_0 = font.render("Pv : "+str(ennemi[0].pv)+"/"+str(ennemi[0].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_0, [500, 200])
        fenetre.blit(ennemi[0].img_combat, ennemi[0].img_combat.get_rect(center=(500, 150)))

        text_pv_ennemi_1 = font.render("Pv : "+str(ennemi[1].pv)+"/"+str(ennemi[1].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_1, [500, 350])
        fenetre.blit(ennemi[1].img_combat, ennemi[1].img_combat.get_rect(center=(500, 300)))

        text_pv_ennemi_2 = font.render("Pv : "+str(ennemi[2].pv)+"/"+str(ennemi[2].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_2, [500, 500])
        fenetre.blit(ennemi[2].img_combat, ennemi[2].img_combat.get_rect(center=(500, 450)))

        if position_bouton == 0:
            fenetre.blit(fleche, fleche.get_rect(center=(350, 150)))
        if position_bouton == 1:
            fenetre.blit(fleche, fleche.get_rect(center=(350, 300)))
        if position_bouton == 2:
            fenetre.blit(fleche, fleche.get_rect(center=(350, 450)))

        for event in pygame.event.get():

            if event.type == QUIT:
                return "End"

            if event.type == KEYDOWN:

                if event.key == K_DOWN:
                    if position_bouton <2:
                        position_bouton += 1

                if event.key == K_UP:
                    if position_bouton >0:
                        position_bouton -= 1

                if event.key == K_TAB:
```

```
                continuer = False

            if event.key == K_RETURN:
                joueur.cible = ennemi[position_bouton]
                anim_joueur(fenetre,joueur)
                if combat_start(joueur,ennemi) == "Mort joueur":
                    return "Mort joueur"
                return "next"

        pygame.display.flip()
if nb_ennemis == 4:
    while continuer:

        text_tour = font.render("tour "+str(tour),True,noir)
        text_pv_joueur = font.render("Pv : "+str(joueur.pv)+"/"+str(joueur.pv_max),True,noir)

        fenetre.blit(background_image, [0, 0])
        fenetre.blit(text_tour, [10, 10])
        fenetre.blit(text_pv_joueur, [10, 500])
        fenetre.blit(joueur.img, joueur.img.get_rect(center=(100, 300)))

        text_pv_ennemi_0 = font.render("Pv : "+str(ennemi[0].pv)+"/"+str(ennemi[0].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_0, [500, 170])
        fenetre.blit(ennemi[0].img_combat, ennemi[0].img_combat.get_rect(center=(500, 120)))

        text_pv_ennemi_1 = font.render("Pv : "+str(ennemi[1].pv)+"/"+str(ennemi[1].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_1, [500, 290])
        fenetre.blit(ennemi[1].img_combat, ennemi[1].img_combat.get_rect(center=(500, 240)))

        text_pv_ennemi_2 = font.render("Pv : "+str(ennemi[2].pv)+"/"+str(ennemi[2].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_2, [500, 410])
        fenetre.blit(ennemi[2].img_combat, ennemi[2].img_combat.get_rect(center=(500, 360)))

        text_pv_ennemi_3 = font.render("Pv : "+str(ennemi[3].pv)+"/"+str(ennemi[3].pv_max),True,noir)
        fenetre.blit(text_pv_ennemi_3, [500, 530])
        fenetre.blit(ennemi[3].img_combat, ennemi[3].img_combat.get_rect(center=(500, 480)))

        if position_bouton == 0:
            fenetre.blit(fleche, fleche.get_rect(center=(350, 120)))
        if position_bouton == 1:
            fenetre.blit(fleche, fleche.get_rect(center=(350, 240)))
        if position_bouton == 2:
            fenetre.blit(fleche, fleche.get_rect(center=(350, 360)))
        if position_bouton == 3:
            fenetre.blit(fleche, fleche.get_rect(center=(350, 480)))

        for event in pygame.event.get():

            if event.type == QUIT:
                return "End"

            if event.type == KEYDOWN:

                if event.key == K_DOWN:
                    if position_bouton <3:
                        position_bouton += 1

                if event.key == K_UP:
                    if position_bouton >0:
                        position_bouton -= 1

                if event.key == K_TAB:
                    continuer = False
```

```
                if event.key == K_RETURN:
                    joueur.cible = ennemi[position_bouton]
                    anim_joueur(fenetre,joueur)
                    if combat_start(joueur,ennemi) == "Mort joueur":
                        return "Mort joueur"
                    return "next"

            pygame.display.flip()

    return "Retour"

def attaque_type(fenetre,joueur,ennemi,tour):
    text1 = font.render(ennemi[0].nom,True,noir)
    text2 = font.render("attaque physique",True,noir)
    text3 = font.render("attaque magique",True,noir)
    continuer = 1
    position_bouton = 1
    joueur.action_type = "attaque"
    pygame.draw.rect(fenetre, blanc, [0, 590, 640, 50])



    while continuer == 1:

        if position_bouton == 1:
            pygame.draw.rect(fenetre, blanc, [200, 590, 200, 50], 3)
            pygame.draw.rect(fenetre, noir, [0, 590, 200, 50], 3)

        if position_bouton == 2:
            pygame.draw.rect(fenetre, blanc, [0, 590, 200, 50], 3)
            pygame.draw.rect(fenetre, noir, [200, 590, 200, 50], 3)

        pygame.draw.rect(fenetre, noir, [0, 590, 640, 50], 1)
        pygame.draw.line(fenetre, noir, [400, 590], [400, 640], 1)
        pygame.draw.line(fenetre, noir, [200, 590], [200, 640], 1)

        fenetre.blit(text1, [430, 600])
        fenetre.blit(text2, [5, 600])
        fenetre.blit(text3, [210, 600])
        pygame.display.flip()

        for event in pygame.event.get():

            if event.type == QUIT:
                return "End"

            if event.type == KEYDOWN:

                if event.key == K_TAB:
                    continuer = 0

                if event.key == K_RIGHT:
                    if position_bouton <2:
                        position_bouton += 1

                if event.key == K_LEFT:
                    if position_bouton >1:
                        position_bouton -= 1

                if event.key == K_RETURN:

                    if position_bouton == 1:
```

```
                joueur.action = "Physique"

            if position_bouton == 2:
                joueur.action = "Magique"

            return select_ennemi(fenetre,joueur,ennemi,tour)
    return "Retour"

def potion_type(fenetre,joueur,ennemi):
    joueur.action_type = "potion"
    position_bouton = 0
    dessiner=pygame.draw.rect
    ecrire=font.render
    continuer = 1
    while continuer == 1:

        fenetre.fill(blanc)

        pygame.draw.rect(fenetre, blanc, [20, 2+45*(position_bouton-1), 450, 40], 3)
        pygame.draw.rect(fenetre, blanc, [20, 2+45*(position_bouton+1), 450, 40], 3)
        pygame.draw.rect(fenetre, noir, [20, 2+45*position_bouton, 450, 40], 3)

        dessiner(fenetre, noir,[20, 2, 450, 40],1)
        fenetre.blit(ecrire("Petite potion de vie",True, noir),(30,10))
        fenetre.blit(ecrire(str(joueur.potionvie1),True,noir),(410,10))

        dessiner(fenetre, noir,[20, 47, 450, 40],1)
        fenetre.blit(ecrire("Grande potion de vie",True, noir),(30,55))
        fenetre.blit(ecrire(str(joueur.potionvie2),True,noir),(410,55))

        dessiner(fenetre, noir,[20, 92, 450, 40],1)
        fenetre.blit(ecrire("Enorme potion de vie",True, noir),(30,100))
        fenetre.blit(ecrire(str(joueur.potionvie3),True,noir),(410,100))

        dessiner(fenetre, noir,[20, 137, 450, 40],1)
        fenetre.blit(ecrire("Petite potion d'armure",True, noir),(30,145))
        fenetre.blit(ecrire(str(joueur.potionarmure1),True,noir),(410,145))

        dessiner(fenetre, noir,[20, 182, 450, 40],1)
        fenetre.blit(ecrire("Grande potion d'armure",True, noir),(30,190))
        fenetre.blit(ecrire(str(joueur.potionarmure2),True,noir),(410,190))

        dessiner(fenetre, noir,[20, 227, 450, 40],1)
        fenetre.blit(ecrire("Enorme potion d'armure",True, noir),(30,235))
        fenetre.blit(ecrire(str(joueur.potionarmure3),True,noir),(410,235))

        dessiner(fenetre, noir,[20, 272, 450, 40],1)
        fenetre.blit(ecrire("Petite potion de force",True, noir),(30,280))
        fenetre.blit(ecrire(str(joueur.potionforce1),True,noir),(410,280))

        dessiner(fenetre, noir,[20, 317, 450, 40],1)
        fenetre.blit(ecrire("Grande potion de force",True, noir),(30,325))
        fenetre.blit(ecrire(str(joueur.potionforce2),True,noir),(410,325))

        dessiner(fenetre, noir,[20, 362, 450, 40],1)
        fenetre.blit(ecrire("Enorme potion de force",True, noir),(30,370))
        fenetre.blit(ecrire(str(joueur.potionforce3),True,noir),(410,370))

        dessiner(fenetre, noir,[20, 407, 450, 40],1)
        fenetre.blit(ecrire("Petite potion de critique",True, noir),(30,415))
        fenetre.blit(ecrire(str(joueur.potioncritique1),True,noir),(410,415))

        dessiner(fenetre, noir,[20, 452, 450, 40],1)
```

```
fenetre.blit(ecrire("Grande potion de critique",True, noir),(30,460))
fenetre.blit(ecrire(str(joueur.potioncritique2),True,noir),(410,460))

dessiner(fenetre, noir,[20, 497, 450, 40],1)
fenetre.blit(ecrire("Enorme potion de critique",True, noir),(30,505))
fenetre.blit(ecrire(str(joueur.potioncritique3),True,noir),(410,505))

dessiner(fenetre, noir,[20, 542, 450, 40],1)
fenetre.blit(ecrire("Potion de vitesse",True, noir),(30,550))
fenetre.blit(ecrire(str(joueur.potionvitesse),True,noir),(410,550))

dessiner(fenetre, noir,[20, 587, 450, 40],1)
fenetre.blit(ecrire("Potion de precision",True, noir),(30,595))
fenetre.blit(ecrire(str(joueur.potionprecision),True,noir),(410,595))

for event in pygame.event.get():

    if event.type == QUIT:
        return "End"

    if event.type == KEYDOWN:

        if event.key == K_TAB:
            continuer = 0

        if event.key == K_DOWN:
            if position_bouton <13:
                position_bouton += 1
        if event.key == K_UP:
            if position_bouton >0:
                position_bouton -= 1

        if event.key == K_RETURN:
            if position_bouton==0:
                joueur.action="vie1"
            if position_bouton==1:
                joueur.action="vie2"
            if position_bouton==2:
                joueur.action="vie3"
            if position_bouton==6:
                joueur.action="force1"
            if position_bouton==7:
                joueur.action="force2"
            if position_bouton==8:
                joueur.action="force3"
            if position_bouton==3:
                joueur.action="armure1"
            if position_bouton==4:
                joueur.action="armure2"
            if position_bouton==5:
                joueur.action="armure3"
            if position_bouton==9:
                joueur.action="critique1"
            if position_bouton==10:
                joueur.action="critique2"
            if position_bouton==11:
                joueur.action="critique3"
            if position_bouton==12:
                joueur.action="vitesse"
            if position_bouton==12:
                joueur.action="precision"
            combat_start(joueur,ennemi)
            return "Next"
```

```
        pygame.display.flip()
    return "Retour"

def anim_joueur(fenetre,joueur):
    imganim = joueur.img_combat
    for p in range(0,12):
        fenetre.blit(imganim,(150,250),(64*20,64*p,64,64))
        pygame.display.flip()
# x=20 y il y en a 13
```

Combat/interface.py

```
# -*- coding: utf-8 -*-
"""
Created on Fri Mar 17 11:55:22 2017

@author: pierre.aubinaud
"""

import combat
from random import randint



class action():
    def __init__(self):
        self.enemi=[]


    def quit_d(self):
        print "%s can't find the way back home, and dies of starvation.\nR.I.P." % self.joueur.nom
        self.joueur.pv = 0


    def help_d(self): print Commands.keys()


    def status(self):
        print "%s's health: %d/%d" % (self.joueur.nom, self.joueur.pv, self.joueur.pv_max)
        if self.joueur.state != 'normal':
            print "%s's health: %d/%d" % (self.enemi[0].nom, self.enemi[0].pv, self.enemi[0].pv_max)
            print "%s's health: %d/%d" % (self.enemi[1].nom, self.enemi[1].pv, self.enemi[1].pv_max)
            print "%s's health: %d/%d" % (self.enemi[2].nom, self.enemi[2].pv, self.enemi[2].pv_max)
            print "%s's health: %d/%d" % (self.enemi[3].nom, self.enemi[3].pv, self.enemi[3].pv_max)



    def tired(self):
        print "%s feels tired." % self.joueur.nom
        self.joueur.pv = max(1, self.joueur.pv - 1)


    def rest(self):
        if self.joueur.state != 'normal':
            print "%s can't rest now!" % self.joueur.nom
            self.joueur.type_action = None
            combat.combat_start(self.joueur,self.enemi)
        else:
          print "%s rests." % self.joueur.nom
          if randint(0, 1):
            self.enemi = [combat.ennemi_test(),combat.ennemi_test(),combat.ennemi_test(),combat.ennemi_test()]
            print "%s is rudely awakened by %s!" % (self.joueur.nom, self.enemi[0].nom)
            self.joueur.state = 'fight'
            self.joueur.type_action = None
            combat.combat_start(self.joueur,self.enemi)
          else:
            if self.joueur.pv < self.joueur.pv_max:
              self.joueur.pv = self.joueur.pv + 1
            else: print "%s slept too much." % self.joueur.nom; self.joueur.pv = self.joueur.pv - 1


    def explore(self):
        if self.joueur.state != 'normal':
```

```
          print "%s is too busy right now!" % self.joueur.nom
          self.joueur.type_action = None
          combat.combat_start(self.joueur,self.enemi)
        else:
          print "%s explores a twisty passage." % self.joueur.nom
          if randint(0, 1):
            self.enemi = [combat.ennemi_test(),combat.ennemi_test(),combat.ennemi_test(),combat.ennemi_test()]
            print "%s encounters %s!" % (self.joueur.nom, self.enemi[0].nom)
            self.joueur.state = 'fight'
          else:
            if randint(0, 1): action.tired(action())


    def flee(self):
        if self.joueur.state != 'fight': print "%s runs in circles for a while." % self.joueur.nom; self.action.tired(action())
        else:
          if randint(1, self.joueur.pv + 5) > randint(1, self.enemi[0].pv):
            print "%s flees from %s." % (self.joueur.nom, self.enemi[0].nom)
            self.enemi = None
            self.joueur.state = 'normal'
          else:
             print "%s couldn't escape from %s!" % (self.joueur.nom, self.enemi[0].nom)
             self.joueur.type_action = None
             combat.combat_start(self.joueur,self.enemi)


    def attack(self):
        if self.joueur.state != 'fight': print "%s swats the air, without notable results." % self.joueur.nom;
self.action.tired(action())
        else:
           self.joueur.action = raw_input("What is your attack ?")
           while 1:
              c=raw_input("What is your cible ?")
              c= int(c)
              if c == 1:
                 self.joueur.cible= self.enemi[0]
                 break
              if c == 2:
                 self.joueur.cible= self.enemi[1]
                 break
              if c == 3:
                 self.joueur.cible= self.enemi[2]
                 break
              if c == 4:
                 self.joueur.cible= self.enemi[3]
                 break
           self.joueur.type_action = "attaque"
           enemi=self.enemi
           combat.combat_start(self.joueur,enemi)
           if self.enemi[0].pv+self.enemi[1].pv+self.enemi[2].pv+self.enemi[3].pv <=0:
              print "%s kill all the enemy!" % (self.joueur.nom)
              self.enemi = None
              self.joueur.state = 'normal'
           elif self.joueur.cible.pv <= 0:
              print "%s executes %s!" % (self.joueur.nom, self.joueur.cible.nom)


while 1:
    p = raw_input("What is your character's class? ")

    if p == "Mage":
        action.joueur=combat.Mage()
        break
```

```
    elif p == "Combattant":
        action.joueur=combat.Combattant()
        break
    elif p == "AssassinsMagique":
        action.joueur=combat.AssassinsMagique()
        break
    elif p == "AssassinsPhysique":
        action.joueur=combat.AssassinsPhysique()
        break
    elif p == "Archer":
        action.joueur=combat.Archer()
        break
    elif p == "Soigneur":
        action.joueur=combat.Soigneur()
        break

action_class=action()
action.joueur.state = 'normal'



Commands = {
  'quit': action_class.quit_d,
  'help': action_class.help_d,
  'status': action_class.status,
  'rest': action_class.rest,
  'explore': action_class.explore,
  'flee': action_class.flee,
  'attack': action_class.attack,
  }



print "(type help to get a list of actions)\n"
print "%s enters a dark cave, searching for adventure." % action.joueur.nom

while(action.joueur.pv > 0):
  line = raw_input("> ")
  args = line.split()
  if len(args) > 0:
    commandFound = False
    for c in Commands:
      if args[0] == c[:len(args[0])]:
        print c
        Commands[c]()
        commandFound = True
        break
    if not commandFound:
      print "%s doesn't understand the suggestion." % action.joueur.nom
```

Combat/passif.py

```
# -*- coding: utf-8 -*-
import combat

    #mage
    def AttaquePhysique(self,adversaire):
        if random.randrange(100)<self.prec:
            if random.randrange(100)>self.crit:
                adversaire.pv -= self.atk*(1-adversaire.defen/100)
            else:
                adversaire.pv -= self.atk*(1-adversaire.defen/100)*2

        if random.randrange(100) < 10:
            return true
        else:
            return false

    def AttaqueMagique(self,adversaire):
        if random.randrange(100)<self.prec:
            if random.randrange(100)>self.crit:
                adversaire.pv -= self.mag*(1-adversaire.res/100)
            else:
                adversaire.pv -= self.atk*(1-adversaire.defen/100)*2

        if random.randrange(100) < 10:
            return true
        else:
            return false
#combat
def combat(joueur,arme,armure,enemi_1,enemi_2,enemi_3,enemi_4):

    passif=0
    while joueur.pv > 0 and (enemi_1.pv > 0 and enemi_2 > 0 and enemi_3 > 0 and enemi_4 > 0):
        tour += 1

        if joueur.nom == "Pierre Antoine":
            if joueur.pv < 500*0.7 and passif == 0:
                joueur.defen == joueur.defen*2
                joueur.res == joueur.res*2
                passif += 1
            else:
                passif = 0


        if joueur.vit > enemi.vit:
            joueur.AttaqueMagique(enemi,arme)
            if enemi.pv > 0:
                enemi.AttaquePhysique(joueur,armure)
        else:
            enemi.AttaquePhysique(joueur,arme)
            if joueur.pv > 0:
                joueur.AttaqueMagique(enemi,armure)
```

Combat/perso.py

```
# -*- coding: utf-8 -*-
import random ,pygame,armes,armure
from pygame.locals import *

class perso:
    def __call__(self):
        print "ceci est une fonction"

    def __init__(self):
        self.jouable=False
        self.pv_max=0
        self.pv=0       #brut
        self.atk=0      #brut
        self.mag=0      #brut
        self.defen=0    #pourcent
        self.res=0      #pourcent
        self.vit=0      #brut
        self.prec=100   #pourcent
        self.crit=0     #pourcent

        #variable passif
        self.passif=0
        self.stun=False
        self.effet=False
        self.brulure=0
        self.saignement=0

        #variable equipement
        self.arme=armes.arme()
        self.armure=armure.armure()
        self.type_action=None
        self.action=None
        self.cible=None

        #variable potion
        self.armure1=-1
        self.armure1fois=0
        self.armure2=-1
        self.armure2fois=0
        self.armure3=-1
        self.armure3fois=0
        self.force1=-1
        self.force1fois=0
        self.force2=-1
        self.force2fois=0
        self.force3=-1
        self.force3fois=0
        self.critique1=-1
        self.critique1fois=0
        self.critique2=-1
        self.critique2fois=0
        self.critique3=-1
        self.critique3fois=0
        self.vitesse=-1
        self.vitessefois=0
        self.precision=-1
        self.precisionfois=0

        #variable inventaire
        self.potionvie1=3
        self.potionvie2=2
```

```
        self.potionvie3=0
        self.potionarmure1=2
        self.potionarmure2=0
        self.potionarmure3=0
        self.potionforce1=1
        self.potionforce2=0
        self.potionforce3=0
        self.potioncritique1=1
        self.potioncritique2=0
        self.potioncritique3=0
        self.potionvitesse=0
        self.potionprecision=0
        self.arme_total=[armes.arme,armes.arme,armes.arme]
        self.armure_total=[armure.armure,armure.armure,armure.armure]

    def upgrade(self,niveau):
        self.arme = self.arme_total[niveau]()
        self.armure = self.armure_total[niveau]()

    def attaque(self):
        '''
            selection du type d'attaque
        '''
        if self.stun == True:
            self.stun = False
        elif self.action == "Physique":
            self.passif_attaque_def()
            self.AttaquePhysique()
        elif self.action == "Magique":
            self.passif_attaque_def()
            self.AttaqueMagique()


    def AttaquePhysique(self):
        '''
            execution d'une attaque physique
        '''
        if random.randrange(100)<self.prec:
            self.passif_attaque_def()
            if random.randrange(100)>self.crit:
                if self.cible.defen+self.cible.armure.defen > 90:
                    self.cible.pv -= (self.atk+self.arme.atk)*(1-float(90)/100)
                else:
                    self.cible.pv -= (self.atk+self.arme.atk)*(1-float(self.cible.defen+self.cible.armure.defen)/100)
            else:
                if self.cible.defen+self.cible.armure.defen > 90:
                    self.cible.pv -= (self.atk+self.arme.atk)*(1-float(90)/100)*2
                else:
                    self.cible.pv -= (self.atk+self.arme.atk)*(1-float(self.cible.defen+self.cible.armure.defen)/100)*2
        if self.cible.pv < 0:
            self.cible.pv = 0



    def AttaqueMagique(self):
        '''
            execution d'une attaque magique
        '''
        if random.randrange(100)<self.prec:
            self.passif_attaque_def()
            if random.randrange(100)>self.crit:
                if self.cible.res+self.cible.armure.res > 90:
                    self.cible.pv -= (self.mag+self.arme.mag)*(1-float(90)/100)
```

```
            else:
                self.cible.pv -= (self.mag+self.arme.mag)*(1-float(self.cible.res+self.cible.armure.res)/100)

        else:
            if self.cible.res+self.cible.armure.res > 90:
                self.cible.pv -= (self.mag+self.arme.mag)*(1-float(90)/100)*2
            else:
                self.cible.pv -= (self.mag+self.arme.mag)*(1-float(self.cible.res+self.cible.armure.res)/100)*2
    if self.cible.pv < 0:
        self.cible.pv = 0

def passif_attaque_def(self):
    '''
        effet des passifs d'attaque
    '''
    print "test"
    pass

def passif_def(self,adv):
    '''
        effet des passifs
    '''
    pass

def effet_def(self):
    '''
        impact des altérations d'état
    '''
    if self.effet==True:
        if self.brulure > 0:
            self.pv -= 5
            self.brulure -= 1
        elif self.saignement > 0:
            self.pv -= 10
            self.saignement -= 1
        else:
            self.effet=False

def popo_actif(self):
    potion=self.action
    if potion == "vie1" and self.potionvie1>0 and self.pv != self.pv_max:
        self.pv += 50
        if self.pv > self.pv_max:
            self.pv = self.pv_max
        self.potionvie1-=1

    if potion == "vie2" and self.potionvie2>0 and self.pv != self.pv_max:
        self.pv += 100
        if self.pv > self.pv_max:
            self.pv = self.pv_max
        self.potionvie2-=1

    if potion == "vie3" and self.potionvie3>0 and self.pv != self.pv_max:
        self.pv += 250
        if self.pv > self.pv_max:
            self.pv = self.pv_max
        self.potionvie3-=1

    if potion == "force1" and self.potionforce1>0:
        self.atk += 50
        self.mag += 50
        self.force1 =2
        self.force1fois +=1
```

```
    self.potionforce1-=1

if potion == "force2" and self.potionforce2>0:
    self.atk +=100
    self.mag +=100
    self.force1 =2
    self.force2fois +=1
    self.potionforce2-=1

if potion == "force3" and self.potionforce3>0:
    self.atk +=150
    self.mag +=150
    self.force3 =1  #"tour"
    self.force3fois +=1
    self.potionforce3-=1

if potion == "armure1" and self.potionarmure1>0:
    self.defen +=5
    self.res +=5
    self.armure1 =2 #"tour"
    self.armure1fois +=1
    self.potionarmure1-=1

if potion == "armure2" and self.potionarmure2>0:
    self.defen +=10
    self.res +=10
    self.armure2 =2 #"tour"
    self.armure2fois +=1
    self.potionarmure2-=1

if potion == "armure3" and self.potionarmure3>0:
    self.defen +=20
    self.res +=20
    self.armure3 =1 #"tour"
    self.armure3fois +=1
    self.potionarmure3-=1

if potion == "critique1" and self.potioncritique1>0:
    self.crit += 5
    self.critique1 =4 #"tour"
    self.critique1fois +=1
    self.potioncritique1-=1

if potion == "critique2" and self.potioncritique2>0:
    self.crit += 10
    self.critique2 =3#"tour"
    self.critique2fois +=1
    self.potioncritique2-=1

if potion == "critique3" and self.potioncritique3>0:
    self.crit += 15
    self.critique3 =2 #"tour"
    self.critique3fois +=1
    self.potioncritique3-=1

if potion == "vitesse" and self.potionvitesse>0:
    self.vit += 100
    self.vitesse =1 #"combat"
    self.vitessefois +=1
    self.potionvitesse-=1

if potion == "precision" and self.potionprecision>0:
    self.prec += 15
```

```
        self.precision =1 #"combat"
        self.precisionfois +=1
        self.potionprecision-=1

def popo_def(self):

    if self.critique3fois >= 0:
        if self.critique3 == 0:
            self.crit -= 15
            self.critique3fois -= 1
            self.critique3 -=1
        else:
            self.critique3 -=1

    if self.critique2fois >= 0:
        if self.critique2 == 0:
            self.crit -= 10
            self.critique2fois -= 1
            self.critique2 -=1
        else:
            self.critique2 -=1

    if self.critique1fois >= 0:
        if self.critique1 == 0:
            self.crit -= 5
            self.critique1fois -= 1
            self.critique1 -=1
        else:
            self.critique1 -=1

    if self.armure1fois >= 0:
        if self.armure1 == 0:
            self.defen -= 5
            self.res -=5
            self.armure1fois -= 1
            self.armure1 -=1
        else:
            self.armure1 -=1

    if self.armure2fois >= 0:
        if self.armure2 == 0:
            self.defen -= 10
            self.res -= 10
            self.armure2fois -= 1
            self.armure2 -=1
        else:
            self.armure2 -=1

    if self.armure3fois >= 0:
        if self.armure3 == 0:
            self.defen -= 20
            self.res -= 20
            self.armure3fois -= 1
            self.armure3 -=1
        else:
            self.armure3 -=1

    if self.force3fois >= 0:
        if self.force3 == 0:
            self.atk -= 150
            self.mag -= 150
            self.force3fois -= 1
            self.force3 -=1
```

```
            else:
                self.force3 -=1

        if self.force2fois >= 0:
            if self.force2 == 0:
                self.atk -= 100
                self.mag -= 100
                self.force2fois -= 1
                self.force2 -=1
            else:
                self.force2 -=1

        if self.force1fois >= 0:
            if self.force1 == 0:
                self.atk -= 50
                self.mag -= 50
                self.force1fois -= 1
                self.force1 -=1
            else:
                self.force1 -=1

    def popo_def_2(self):

        if self.precision == 1:
            self.precision -= 15 * self.precisionfois
            self.precisionfois -= 1
            self.precision -=1

        if self.vitesse == 1:
            self.vitesse -= 15 * self.vitessefois
            self.vitessefois -= 1
            self.vitesse -=1

class AssassinsMagique(perso):
    def __init__(self):
        perso.__init__(self)
        self.nom="Sebastien"
        self.classe="Assassin Magique"
        self.jouable=True
        self.pv_max=250
        self.pv=250
        self.atk=10
        self.mag=100
        self.defen=5
        self.res=5
        self.vit=80
        self.prec=100
        self.crit=10
        self.arme_total=[armes.LamesE,armes.Fouet,armes.Crescent]
        self.armure_total=[armure.Gilet,armure.Veste,armure.Armurean]
        self.image_complet = pygame.image.load("data/Sebastien/complet.png").convert_alpha()
        self.img= pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()
        self.img_gauche = pygame.image.load("data/perso.png").convert_alpha()
        self.img_droite = pygame.image.load("data/perso.png").convert_alpha()
        self.img_haut = pygame.image.load("data/perso.png").convert_alpha()
        self.img_bas = pygame.image.load("data/perso.png").convert_alpha()
        self.ls_haut = [self.image_complet.subsurface((64*i, 64*8, 64, 64)) for i in range(9)]
        self.ls_gauche = [self.image_complet.subsurface((64*i, 64*9, 64, 64)) for i in range(9)]
        self.ls_bas = [self.image_complet.subsurface((64*i, 64*10, 64, 64)) for i in range(9)]
        self.ls_droite = [self.image_complet.subsurface((64*i, 64*11, 64, 64)) for i in range(9)]
        self.ls_images = [self.ls_gauche, self.ls_droite, self.ls_haut, self.ls_bas]
```

```
        def passif_attaque_def(self):
            print 5
            self.pv += 5

class Mage(perso):

    def __init__(self):
        perso.__init__(self)
        self.nom="Emeric"
        self.classe="Mage"
        self.jouable=True
        self.pv_max=150
        self.pv=150
        self.atk=10
        self.mag=125
        self.defen=5
        self.res=10
        self.vit=80
        self.prec=95
        self.crit=10
        self.arme_total=[armes.Livre,armes.Parchemin,armes.Grimoire]
        self.armure_total=[armure.Robe,armure.TuniqueI,armure.TogeI]
        self.image_complet = pygame.image.load("data/Emeric/complet.png").convert_alpha()
        self.img= pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()
        self.img_gauche = pygame.image.load("data/perso.png").convert_alpha()
        self.img_droite = pygame.image.load("data/perso.png").convert_alpha()
        self.img_haut = pygame.image.load("data/perso.png").convert_alpha()
        self.img_bas = pygame.image.load("data/perso.png").convert_alpha()
        self.ls_haut = [self.image_complet.subsurface((64*i, 64*8, 64, 64)) for i in range(9)]
        self.ls_gauche = [self.image_complet.subsurface((64*i, 64*9, 64, 64)) for i in range(9)]
        self.ls_bas = [self.image_complet.subsurface((64*i, 64*10, 64, 64)) for i in range(9)]
        self.ls_droite = [self.image_complet.subsurface((64*i, 64*11, 64, 64)) for i in range(9)]
        self.ls_images = [self.ls_gauche, self.ls_droite, self.ls_haut, self.ls_bas]

    def passif_attaque_def(self):
        if random.randrange(10)<1:
            self.cible.stun=True

class AssassinsPhysique(perso):
    def __init__(self):
        perso.__init__(self)
        self.nom="Nassim"
        self.classe="Assassin Physique"
        self.jouable=True
        self.pv_max=250
        self.pv=250
        self.atk=100
        self.mag=10
        self.defen=5
        self.res=5
        self.vit=80
        self.prec=100
        self.crit=10
        self.arme_total=[armes.Dagues,armes.LamesD,armes.Hachettes]
        self.armure_total=[armure.Tenue,armure.Manteau,armure.Armureco]
        self.image_complet = pygame.image.load("data/Nassim/complet.png").convert_alpha()
        self.img= pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()
        self.img_gauche = pygame.image.load("data/perso.png").convert_alpha()
        self.img_droite = pygame.image.load("data/perso.png").convert_alpha()
        self.img_haut = pygame.image.load("data/perso.png").convert_alpha()
        self.img_bas = pygame.image.load("data/perso.png").convert_alpha()
```

```
        self.ls_haut = [self.image_complet.subsurface((64*i, 64*8, 64, 64)) for i in range(9)]
        self.ls_gauche = [self.image_complet.subsurface((64*i, 64*9, 64, 64)) for i in range(9)]
        self.ls_bas = [self.image_complet.subsurface((64*i, 64*10, 64, 64)) for i in range(9)]
        self.ls_droite = [self.image_complet.subsurface((64*i, 64*11, 64, 64)) for i in range(9)]
        self.ls_images = [self.ls_gauche, self.ls_droite, self.ls_haut, self.ls_bas]

     def passif_attaque_def(self):
        self.cible.saignement = 1

class Combattant(perso):
    def __init__(self):
        perso.__init__(self)
        self.nom="Pierre Antoine"
        self.classe="Combattant"
        self.jouable=True
        self.pv_max=450
        self.pv=450
        self.atk=75
        self.mag=10
        self.defen=10
        self.res=5
        self.vit=50
        self.prec=100
        self.crit=5
        self.arme_total=[armes.Epee,armes.Hache,armes.EpeeRL]
        self.armure_total=[armure.Armurecu,armure.Plastron,armure.Cuirasse]
        self.image_complet = pygame.image.load("data/PA/complet.png").convert_alpha()
        self.img= pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()
        self.img_gauche = pygame.image.load("data/perso.png").convert_alpha()
        self.img_droite = pygame.image.load("data/perso.png").convert_alpha()
        self.img_haut = pygame.image.load("data/perso.png").convert_alpha()
        self.img_bas = pygame.image.load("data/perso.png").convert_alpha()
        self.ls_haut = [self.image_complet.subsurface((64*i, 64*8, 64, 64)) for i in range(9)]
        self.ls_gauche = [self.image_complet.subsurface((64*i, 64*9, 64, 64)) for i in range(9)]
        self.ls_bas = [self.image_complet.subsurface((64*i, 64*10, 64, 64)) for i in range(9)]
        self.ls_droite = [self.image_complet.subsurface((64*i, 64*11, 64, 64)) for i in range(9)]
        self.ls_images = [self.ls_gauche, self.ls_droite, self.ls_haut, self.ls_bas]
    def passif_def(self,adv):
        if self.pv < 500*0.7 and self.passif == 0:
            self.defen == self.defen*2
            self.res == self.res*2
            self.passif += 1
        elif self.pv < 500*0.7 and self.passif != 0:
            pass
        else:
            self.defen == self.defen/2
            self.res == self.res/2
            self.passif = 0

class Archer(perso):
    def __init__(self):
        perso.__init__(self)
        self.nom="Martin"
        self.classe="Archer"
        self.jouable=True
        self.pv_max=100
        self.pv=100
        self.atk=125
        self.mag=10
        self.defen=5
        self.res=5
        self.vit=100
```

```
        self.prec=90
        self.crit=20
        self.arme_total=[armes.ArcL,armes.Arbalete,armes.ArcA]
        self.armure_total=[armure.Cape,armure.Justaucorps,armure.Cote]
        self.image_complet = pygame.image.load("data/Martin/complet.png").convert_alpha()
        self.img= pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()
        self.img_gauche = pygame.image.load("data/perso.png").convert_alpha()
        self.img_droite = pygame.image.load("data/perso.png").convert_alpha()
        self.img_haut = pygame.image.load("data/perso.png").convert_alpha()
        self.img_bas = pygame.image.load("data/perso.png").convert_alpha()
        self.ls_haut = [self.image_complet.subsurface((64*i, 64*8, 64, 64)) for i in range(9)]
        self.ls_gauche = [self.image_complet.subsurface((64*i, 64*9, 64, 64)) for i in range(9)]
        self.ls_bas = [self.image_complet.subsurface((64*i, 64*10, 64, 64)) for i in range(9)]
        self.ls_droite = [self.image_complet.subsurface((64*i, 64*11, 64, 64)) for i in range(9)]
        self.ls_images = [self.ls_gauche, self.ls_droite, self.ls_haut, self.ls_bas]
        self.cible2=None

    def attaque2(self):
        '''
            selection du type d'attaque
        '''
        if self.stun == True:
            self.stun = False
            pass
        elif self.action == "Physique":
            self.AttaquePhysique2()
        elif self.action == "Magique":
            self.AttaqueMagique2()

    def AttaquePhysique2(self):
        '''
            execution d'une attaque physique
        '''
        if random.randrange(100)<self.prec:
            if random.randrange(100)>self.crit:
                self.cible2.pv -= (self.atk+self.arme.atk)*(1-float(self.cible2.defen+self.cible2.armure.defen)/100)
            else:
                self.cible2.pv -= (self.atk+self.arme.atk)*(1-float(self.cible2.defen+self.cible2.armure.defen)/100)*2
        if self.cible2.pv < 0:
            self.cible2.pv = 0

    def AttaqueMagique2(self):
        '''
            execution d'une attaque magique
        '''
        if random.randrange(100)<self.prec:
            if random.randrange(100)>self.crit:
                self.cible2.pv -= (self.mag+self.arme.mag)*(1-float(self.cible2.res+self.cible2.armure.res)/100)
            else:
                self.cible2.pv -= (self.mag+self.arme.mag)*(1-float(self.cible2.res+self.cible2.armure.res)/100)*2
        if self.cible2.pv < 0:
            self.cible2.pv = 0

    def passif_def(self,adv):
        if len(adv) > 2:
            while 1:
                i=random.randrange(len(adv))
                if adv[i].jouable == False and adv[i] != self.cible:
                    self.cible2=adv[i]
                    self.attaque2()
                    break
```

```
class Soigneur(perso):
    def __init__(self):
        perso.__init__(self)
        self.nom="Clarisse"
        self.classe="Soigneur"
        self.jouable=True
        self.pv_max=175
        self.pv=175
        self.atk=10
        self.mag=75
        self.defen=5
        self.res=5
        self.vit=30
        self.prec=95
        self.crit=0
        self.arme_total=[armes.Baton,armes.Orbe,armes.Sceptre]
        self.image_complet = pygame.image.load("data/Emeric/complet.png").convert_alpha()
        self.img= pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()
        self.img_gauche = pygame.image.load("data/perso.png").convert_alpha()
        self.img_droite = pygame.image.load("data/perso.png").convert_alpha()
        self.img_haut = pygame.image.load("data/perso.png").convert_alpha()
        self.img_bas = pygame.image.load("data/perso.png").convert_alpha()
        self.ls_haut = [self.image_complet.subsurface((64*i, 64*8, 64, 64)) for i in range(9)]
        self.ls_gauche = [self.image_complet.subsurface((64*i, 64*9, 64, 64)) for i in range(9)]
        self.ls_bas = [self.image_complet.subsurface((64*i, 64*10, 64, 64)) for i in range(9)]
        self.ls_droite = [self.image_complet.subsurface((64*i, 64*11, 64, 64)) for i in range(9)]
        self.ls_images = [self.ls_gauche, self.ls_droite, self.ls_haut, self.ls_bas]
    def passif_def(self,adv):
        self.pv += 15
        if self.pv>self.pv_max:
            self.pv=self.pv_max

class mobs(perso):
    def __init__(self):
        perso.__init__(self)
        self.nombre=1

    def cible_def(self, participant):
        '''
            IA basique pour definir le type d'attaque
            des ennemis envers le joueur
        '''
        if self.jouable==False:
            if self.atk > self.mag:
                self.action="Physique"
            else:
                self.action="Magique"
            for i in range (len(participant)):
                if participant[i].jouable==True:
                    self.cible=participant[i]

class Rats(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Rats"
        self.pv_max=70
        self.pv=70
        self.atk=15
        self.defen=5
        self.res=10
        self.vit=20
        self.nombre=4
```

```
        self.img = pygame.image.load("data/Ennemi/rat_carte.png").convert_alpha()
        self.img_combat = pygame.image.load("data/Ennemi/rat_combat.png").convert_alpha()

class Gobelins(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Gobelins"
        self.pv_max=40
        self.pv=40
        self.atk=0
        self.mag=15
        self.defen=5
        self.res=10
        self.vit=40
        self.nombre=4
        self.img = pygame.image.load("data/Ennemi/goblin_carte.png").convert_alpha()
        self.img_combat = pygame.image.load("data/Ennemi/goblin_combat.png").convert_alpha()

class Aigles(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Aigles"
        self.pv_max=25
        self.pv=25
        self.atk=30
        self.mag=0
        self.defen=10
        self.res=0
        self.vit=120
        self.nombre=2
        self.img = pygame.image.load("data/Ennemi/aigle_carte.png").convert_alpha()
        self.img_combat = pygame.image.load("data/Ennemi/aigle_combat.png").convert_alpha()

class Slime(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Slime"
        self.pv_max=100
        self.pv=100
        self.atk=0
        self.mag=30
        self.defen=10
        self.res=20
        self.vit=40
        self.nombre=1
        self.img = pygame.image.load("data/Ennemi/slime_carte.png").convert_alpha()
        self.img_combat = pygame.image.load("data/Ennemi/slime_combat.png").convert_alpha()

class Centaures(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Centaures"
        self.pv_max=80
        self.pv=80
        self.atk=0
        self.mag=40
        self.defen=10
        self.res=20
        self.vit=70
        self.nombre=3
        self.img = pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()
```

```
class Loup_garou(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Loup Garou"
        self.pv_max=250
        self.pv=250
        self.atk=50
        self.mag=0
        self.defen=20
        self.res=10
        self.vit=70
        self.nombre=1
        self.img = pygame.image.load("data/Ennemi/loupgarou_carte.png").convert_alpha()
        self.img_combat = pygame.image.load("data/Ennemi/loupgarou_combat.png").convert_alpha()

class Araingnees(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Araingnees"
        self.pv_max=20
        self.pv=20
        self.atk=0
        self.mag=30
        self.defen=5
        self.res=5
        self.vit=120
        self.nombre=3
        self.img = pygame.image.load("data/Ennemi/araignee_carte.png").convert_alpha()
        self.img_combat = pygame.image.load("data/Ennemi/araignee_combat.png").convert_alpha()

class Carapateur(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Carapateur"
        self.pv_max=300
        self.pv=300
        self.atk=0
        self.mag=0
        self.defen=25
        self.res=25
        self.vit=10
        self.nombre=1
        self.img = pygame.image.load("data/Ennemi/carapateur_carte.png").convert_alpha()
        self.img_combat = pygame.image.load("data/Ennemi/carapateur_combat.png").convert_alpha()

class Golems(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Golem"
        self.pv_max=300
        self.pv=300
        self.atk=0
        self.mag=40
        self.defen=15
        self.res=25
        self.vit=20
        self.nombre=2
        self.img = pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()

class Treant(mobs):
    def __init__(self):
        mobs.__init__(self)
```

```
        self.nom="Treant"
        self.pv_max=500
        self.pv=500
        self.atk=0
        self.mag=45
        self.defen=20
        self.res=25
        self.vit=10
        self.nombre=1
        self.img = pygame.image.load("data/Ennemi/treant_carte.png").convert_alpha()
        self.img_combat = pygame.image.load("data/Ennemi/treant_combat.png").convert_alpha()

class Geant(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Geant"
        self.pv_max=400
        self.pv=400
        self.atk=40
        self.mag=45
        self.defen=25
        self.res=20
        self.vit=10
        self.nombre=1
        self.img = pygame.image.load("data/Ennemi/geant_carte.png").convert_alpha()
        self.img_combat = pygame.image.load("data/Ennemi/geant_combat.png").convert_alpha()

class Nains(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Nains"
        self.pv_max=100
        self.pv=100
        self.atk=40
        self.mag=0
        self.defen=20
        self.res=25
        self.vit=40
        self.nombre=3
        self.img = pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()

class Elfs(mobs):
    def __init__(self):
        mobs.__init__(self)
        self.nom="Elfs"
        self.pv_max=85
        self.pv=85
        self.atk=0
        self.mag=30
        self.defen=15
        self.res=30
        self.vit=90
        self.nombre=2
        self.img = pygame.image.load("data/perso.png").convert_alpha()
        self.img_combat = pygame.image.load("data/perso.png").convert_alpha()
```

Combat/popoactif.py

```
# -*- coding: utf-8 -*-
"""
Created on Fri Mar 24 10:20:28 2017

@author: pierre.aubinaud
"""

def popo_actif(self):
    potion=self.action
    if potion == "vie1":
        self.pv += 50
        if self.pv > self.pv_max:
            self.pv = self.pv_max

    if potion == "vie2":
        self.pv += 100
        if self.pv > self.pv_max:
            self.pv = self.pv_max

    if potion == "vie3":
        self.pv += 250
        if self.pv > self.pv_max:
            self.pv = self.pv_max

    if potion == "force1":
        self.atk += 50
    self.mag += 50
    self.force1 =2
    self.force1fois +=1

    if potion == "force2": self.atk +=100
    self.mag +=100
    self.force1 =2
    self.force2fois +=1

    if potion == "force3": self.atk +=150
    self.mag +=150
    self.force3 =1  #"tour"
    self.force3fois +=1

    if potion == "armure1": self.defend +=5
    self.res +=5
    self.armure1 =2 #"tour"
    self.armure1fois +=1

    if potion == "armure2": self.defend +=10
    self.res +=10
    self.armure2 =2 #"tour"
    self.armure2fois +=1

    if potion == "armure3": self.defend +=20
    self.res +=20
    self.armure3 =1 #"tour"
    self.armure3fois +=1

    if potion == "critique1": self.crit += 5
    self.critique1 =4 #"tour"
    self.critique1fois +=1

    if potion == "critique2": self.crit += 10
    self.critique2 =3#"tour"
```

```
    self.critique2fois +=1

    if potion == "critique3": self.crit += 15
    self.critique3 =2 #"tour"
    self.critique3fois +=1

    if potion == "vitesse": self.vit += 100
    self.vitesse =1 #"combat"
    self.vitessefois +=1

    if potion == "precision": self.prec += 15
    self.precision =1 #"combat"
    self.precisionfois +=1


def popo_def(self):
        if self.precisionfois > 0:
            if self.precision == 0:
                self.prec -= 10
                self.precisionfois -= 1
            else:
                self.precison -=1

        if self.vitessefois > 0:
            if self.vitesse == 0:
                self.vit -= 10
                self.vitessefois -= 1
            else:
                self.vitesse -=1

        if self.critique3fois > 0:
            if self.critique3 == 0:
                self.crit -= 15
                self.critique3fois -= 1
            else:
                self.critique3 -=1

        if self.critique2fois > 0:
            if self.critique2 == 0:
                self.crit -= 10
                self.critique2fois -= 1
            else:
                self.critique2 -=1

        if self.critique1fois > 0:
            if self.critique1 == 0:
                self.crit -= 5
                self.critique1fois -= 1
            else:
                self.critique1 -=1

        if self.armure1fois > 0:
            if self.armure1 == 0:
                self.defend -= 5
                self.res -=5
                self.armure1fois -= 1
            else:
                self.armure1 -=1

        if self.armure2fois > 0:
            if self.armure2 == 0:
                self.defend -= 5
                self.res -=5
```

```
            self.armure2fois -= 1
        else:
            self.armure2 -=1

    if self.armure3fois > 0:
        if self.armure3 == 0:
            self.defend -= 5
            self.res -=5
            self.armure3fois -= 1
        else:
            self.armure3 -=1

    if self.force3fois > 0:
        if self.force3 == 0:
            self.atk -= 150
            self.mag -= 150
            self.force3fois -= 1
        else:
            self.force3 -=1

    if self.force2fois > 0:
        if self.force2 == 0:
            self.atk -= 100
            self.mag -= 100
            self.force2fois -= 1
        else:
            self.force2 -=1

    if self.force1fois > 0:
        if self.force1 == 0:
            self.atk -= 50
            self.mag -= 50
            self.force1fois -= 1
        else:
            self.force1 -=1
```

Combat/potion classe.py

```
# -*- coding: utf-8 -*-

class potionvie:
    def __init__(self):
        self.pv=0

class PoVie1(potionvie):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="potion de vie"
        self.pv=50

class PoVie2(potionvie):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="grande potion de vie"
        self.pv=100

class PoVie3(potionvie):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="Énorme potion de vie"
        self.pv=250

class potionforce:
    def __init__(self):
        self.atk=0
        self.mag=0

class PoForce1(potionforce):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="potion de force"
        self.atk=50
        self.mag=50

class PoForce2(potionforce):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="grande potion de force"
        self.atk=100
        self.mag=100

class PoForce3(potionforce):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="Énorme potion de force"
        self.atk=150
        self.mag=150

class potionarmure:
    def __init__(self):
        self.defend=0
        self.res=0

class PoArmure1(potionarmure):
    def __init__(self):
        potionarmure.__init__(self)
        self.nom="potion d'armure"
        self.defend=5
        self.res=5
```

```
class PoArmure2(potionarmure):
    def __init__(self):
        potionarmure.__init__(self)
        self.nom="grande potion d'armure"
        self.defend=10
        self.res=10

class PoArmure3(potionarmure):
    def __init__(self):
        potionarmure.__init__(self)
        self.nom="Énorme potion d'armure"
        self.defend=20
        self.res=20

class potioncritique:
    def __init__(self):
        self.crit=0

class PoCritique1(potioncritique):
    def __init__(self):
        potioncritique.__init__(self)
        self.nom="potion de critique"
        self.crit=5

class PoCritique2(potioncritique):
    def __init__(self):
        potioncritique.__init__(self)
        self.nom="grande potion de critique"
        self.crit=10

class PoCritique3(potioncritique):
    def __init__(self):
        potioncritique.__init__(self)
        self.nom="Énorme potion de critique"
        self.crit=15

class potionprecision:
    def __init__(self):
        self.vit=100

class potionvitesse:
    def __init__(self):
        self.prec=50
```

Combat/potion.py

```
# -*- coding: utf-8 -*-
"""
Created on Fri Mar 17 10:44:33 2017

@author: pierre.aubinaud
"""

class potionvie:
    def __init__(self):
        self.pv=0

class PoVie1(potionvie):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="potion de vie"
        self.pv=50

class PoVie2(potionvie):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="grande potion de vie"
        self.pv=100

class PoVie3(potionvie):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="Énorme potion de vie"
        self.pv=250

class potionforce:
    def __init__(self):
        self.atk=0
        self.mag=0

class PoForce1(potionforce):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="potion de force"
        self.atk=50
        self.mag=50

class PoForce2(potionforce):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="grande potion de force"
        self.atk=100
        self.mag=100

class PoForce3(potionforce):
    def __init__(self):
        potionvie.__init__(self)
        self.nom="Énorme potion de force"
        self.atk=150
        self.mag=150

class potionarmure:
    def __init__(self):
        self.defend=0
        self.res=0

class PoArmure1(potionarmure):
```

```
    def __init__(self):
        potionarmure.__init__(self)
        self.nom="potion d'armure"
        self.defend=5
        self.res=5

class PoArmure2(potionarmure):
    def __init__(self):
        potionarmure.__init__(self)
        self.nom="grande potion d'armure"
        self.defend=10
        self.res=10

class PoArmure3(potionarmure):
    def __init__(self):
        potionarmure.__init__(self)
        self.nom="Énorme potion d'armure"
        self.defend=20
        self.res=20

class potioncritique:
    def __init__(self):
        self.crit=0

class PoCritique1(potioncritique):
    def __init__(self):
        potioncritique.__init__(self)
        self.nom="potion de critique"
        self.crit=5

class PoCritique2(potioncritique):
    def __init__(self):
        potioncritique.__init__(self)
        self.nom="grande potion de critique"
        self.crit=10

class PoCritique3(potioncritique):
    def __init__(self):
        potioncritique.__init__(self)
        self.nom="Énorme potion de critique"
        self.crit=15

class potionprecision:
    def __init__(self):
        self.vit=100

class potionvitesse:
    def __init__(self):
        self.prec=50
```

Combat/sanstitre1.py

```
# -*- coding: utf-8 -*-
"""
Created on Fri Mar 17 11:55:22 2017

@author: pierre.aubinaud
"""

from random import randint

class Character:
  def __init__(self):
    self.name = ""
    self.health = 1
    self.health_max = 1
  def do_damage(self, enemy):
    damage = min(
        max(randint(0, self.health) - randint(0, enemy.health), 0),
        enemy.health)
    enemy.health = enemy.health - damage
    if damage == 0: print "%s evades %s's attack." % (enemy.name, self.name)
    else: print "%s hurts %s!" % (self.name, enemy.name)
    return enemy.health <= 0

class Enemy(Character):
  def __init__(self, player):
    Character.__init__(self)
    self.name = 'a goblin'
    self.health = randint(1, player.health)

class Player(Character):

  def __init__(self):
    Character.__init__(self)
    self.state = 'normal'
    self.health = 10
    self.health_max = 10


  def quit(self):
    print "%s can't find the way back home, and dies of starvation.\nR.I.P." % self.name
    self.health = 0


  def help(self): print Commands.keys()


  def status(self): print "%s's health: %d/%d" % (self.name, self.health, self.health_max)


  def tired(self):
    print "%s feels tired." % self.name
    self.health = max(1, self.health - 1)


  def rest(self):
    if self.state != 'normal': print "%s can't rest now!" % self.name; self.enemy_attacks()
    else:
      print "%s rests." % self.name
      if randint(0, 1):
        self.enemy = Enemy(self)
        print "%s is rudely awakened by %s!" % (self.name, self.enemy.name)
```

```
      self.state = 'fight'
      self.enemy_attacks()
    else:
      if self.health < self.health_max:
        self.health = self.health + 1
      else: print "%s slept too much." % self.name; self.health = self.health - 1


  def explore(self):
    if self.state != 'normal':
      print "%s is too busy right now!" % self.name
      self.enemy_attacks()
    else:
      print "%s explores a twisty passage." % self.name
      if randint(0, 1):
        self.enemy = Enemy(self)
        print "%s encounters %s!" % (self.name, self.enemy.name)
        self.state = 'fight'
      else:
        if randint(0, 1): self.tired()


  def flee(self):
    if self.state != 'fight': print "%s runs in circles for a while." % self.name; self.tired()
    else:
      if randint(1, self.health + 5) > randint(1, self.enemy.health):
        print "%s flees from %s." % (self.name, self.enemy.name)
        self.enemy = None
        self.state = 'normal'
      else: print "%s couldn't escape from %s!" % (self.name, self.enemy.name); self.enemy_attacks()


  def attack(self):
    if self.state != 'fight': print "%s swats the air, without notable results." % self.name; self.tired()
    else:
      if self.do_damage(self.enemy):
        print "%s executes %s!" % (self.name, self.enemy.name)
        self.enemy = None
        self.state = 'normal'
        if randint(0, self.health) < 10:
          self.health = self.health + 1
          self.health_max = self.health_max + 1
          print "%s feels stronger!" % self.name
      else: self.enemy_attacks()


  def enemy_attacks(self):
    if self.enemy.do_damage(self): print "%s was slaughtered by %s!!!\nR.I.P." %(self.name, self.enemy.name)

Commands = {
  'quit': Player.quit,
  'help': Player.help,
  'status': Player.status,
  'rest': Player.rest,
  'explore': Player.explore,
  'flee': Player.flee,
  'attack': Player.attack,
  }

p = Player()
p.name = raw_input("What is your character's name? ")
print "(type help to get a list of actions)\n"
print "%s enters a dark cave, searching for adventure." % p.name
```

```
while(p.health > 0):
  line = raw_input("> ")
  args = line.split()
  if len(args) > 0:
    commandFound = False
    for c in Commands:
      if args[0] == c[:len(args[0])]:
        print c
        Commands[c](p)
        commandFound = True
        break
    if not commandFound:
      print "%s doesn't understand the suggestion." % p.name
```

Combat/test.py

```
# -*- coding: utf-8 -*-
import combat
'''
print "class joueur"
class_joueur=raw_input()
print "class enemi"
class_enemi=raw_input()

if class_joueur == "Mage":
    joueur=combat.Mage()
if class_joueur == "Combattant":
    joueur=combat.Combattant()
if class_joueur == "AssassinsMagique":
    joueur=combat.AssassinsMagique()
if class_joueur == "AssassinsPhysique":
    joueur=combat.AssassinsPhysique()
if class_joueur == "Archer":
    joueur=combat.Archer()
if class_joueur == "Soigneur":
    joueur=combat.Soigneur()

if class_enemi == "Mage":
   enemi=combat.Mage()
if class_enemi == "Combattant":
    enemi=combat.Combattant()
if class_enemi == "AssassinsMagique":
    enemi=combat.AssassinsMagique()
if class_enemi == "AssassinsPhysique":
    enemi=combat.AssassinsPhysique()
if class_enemi == "Archer":
    enemi=combat.Archer()
if class_enemi == "Soigneur":
    enemi=combat.Soigneur()
'''
'''
joueur=combat.AssassinsMagique()
enemi=combat.AssassinsPhysique()

tour=0


while joueur.pv > 0 and enemi.pv > 0:
    tour += 1

    if joueur.vit > enemi.vit:
        joueur.AttaqueMagique(enemi)
        if enemi.pv > 0:
            enemi.AttaquePhysique(joueur)
    else:
        enemi.AttaquePhysique(joueur)
        if joueur.pv > 0:
            joueur.AttaqueMagique(enemi)

    print ("tour "+str(tour))

if joueur.pv <= 0 and enemi.pv >= 0:

    print "perdu"
    print joueur.pv
    print enemi.pv
```

```
elif enemi.pv <= 0 and joueur.pv >= 0:

    print "gagner"
    print joueur.pv
    print enemi.pv

else:

    print "egalite"
    print joueur.pv
    print enemi.pv
'''

joueur=combat.AssassinsMagique()
enemi_1=combat.ennemi_test()
enemi_2=combat.ennemi_test()
enemi_3=combat.ennemi_test()
enemi_4=combat.ennemi_test()
armure=combat.TogeA()
arme=combat.Livre()
enemi=[enemi_1,enemi_2,enemi_3,enemi_4]
combat.combat_phase(joueur,arme,armure,enemi_1,"Magique",enemi)
print joueur.pv,enemi_1.pv,enemi_2.pv,enemi_3.pv,enemi_4.pv
enemi=[enemi_1,enemi_2,enemi_3,enemi_4]
combat.combat_phase(joueur,arme,armure,enemi_2,"Magique",enemi)
print joueur.pv,enemi_1.pv,enemi_2.pv,enemi_3.pv,enemi_4.pv
enemi=[enemi_1,enemi_2,enemi_3,enemi_4]
combat.combat_phase(joueur,arme,armure,enemi_3,"Magique",enemi)
print joueur.pv,enemi_1.pv,enemi_2.pv,enemi_3.pv,enemi_4.pv
enemi=[enemi_1,enemi_2,enemi_3,enemi_4]
combat.combat_phase(joueur,arme,armure,enemi_4,"Magique",enemi)
print joueur.pv,enemi_1.pv,enemi_2.pv,enemi_3.pv,enemi_4.pv
enemi=[enemi_1,enemi_2,enemi_3,enemi_4]
```

Combat/testaffichage.py

```
# -*- coding: utf-8 -*-
"""
Created on Mon May 01 20:26:04 2017

@author: nassim
"""
import pygame
import combat

pygame.init()
fenetre=pygame.display.set_mode((640,640))
pygame.display.set_caption('Programme Pygame de base')

joueur = combat.Mage
ennemi = Elfs

combat.affiche_combat(fenetre,joueur,ennemi)
pygame.display.flip()
```

creation_carte/dernier_niveau_auto.py

```
# -*- coding: utf-8 -*-
import random
import pickle


nb_cell = [48, 48]

def creer_labyrinthe():

    mat = creer_matrice(nb_cell[0], nb_cell[1])
    x = random.randrange(nb_cell[0])
    y = random.randrange(nb_cell[1])

    mat[x][y] = 1
    ls_possible = [(x, y)]

    while ls_possible != []:
        effacer_coords(x, y, ls_possible)
        trouver_nv(x, y, mat, ls_possible)

        if ls_possible != []:
            (x, y) = random.choice(ls_possible)
            mat[x][y] = 1
    return mat

def creer_matrice(x, y):
    mat = []
    for i in range(x):
        mat.append([])
        for j in range(y):
            mat[i].append(0)
    return mat

def nb_voisin(x, y, mat):

    nb_voisin = 0
    voisins = ((0, 1), (1, 0), (0, -1), (-1, 0))
    for a in voisins:
        if x+a[0]>=0 and x+a[0]<nb_cell[0] and y+a[1]>=0 and y+a[1]<nb_cell[1]:
            nb_voisin += mat[x+a[0]][y+a[1]]

    return nb_voisin


def trouver_nv(x, y, mat, ls):
    voisins = ((0, 1), (1, 0), (0, -1), (-1, 0))
    for a in voisins:
        if x+a[0]>=0 and x+a[0]<nb_cell[0] and y+a[1]>=0 and y+a[1]<nb_cell[1]:
            if nb_voisin(x+a[0], y+a[1], mat) == 1 and mat[x+a[0]][y+a[1]] == 0:
                ls.append((x+a[0], y+a[1]))
    return ls

def effacer_coords(x, y, ls):
    voisins = ((0, 1), (1, 0), (0, -1), (-1, 0), (0, 0))
    for a in voisins:
        if (x+a[0], y+a[1]) in ls:
            ls.remove((x+a[0], y+a[1]))

def creer_mat_case(lab):
    matrice = []
    for x in range(50):
```

```
        matrice.append([])
        for y in range(50):
            matrice[x].append("mur")
    for x in range(24):
        for y in range(24):
            if lab[x][y] == 1:
                matrice[(x*2)+1][(y*2)+1] = "herbe"

    for x in range(1, 47, 2):
        for y in range(1, 47, 2):
            if matrice[x][y] == "herbe" and matrice[x+2][y] == "herbe":
                matrice[x+1][y] = "herbe"

            if matrice[x][y] == "herbe" and matrice[x][y+2] == "herbe":
                matrice[x][y+1] = "herbe"
    return matrice

def c_matrice2(N):
    l = []
    for x in range(N):
        l.append([])
        for y in range(N):
            l[x].append("")
    return l

def creer_fichier():
    with open("carte.mp", "wb") as fichier:
        pick = pickle.Pickler(fichier)
        pick.dump([mat, c_matrice2(50)])

lab = creer_labyrinthe()
mat = creer_mat_case(lab)
print mat
creer_fichier()
```

Editeur_Map/Editeur de map.py

```
# -*- coding: utf-8 -*-

from Tkinter import *
import pickle
import tkFileDialog

nombre_case = 50
l_case=10
largeur = nombre_case*l_case
hauteur = nombre_case*l_case

type_case = "herbe"
vitesse = 0
bouton_1 = False

pos_debut = (0, 0)
pos_fin = (0, 0)

# niv 1
#dict_case = {"herbe":"green", "mur":"grey"}
#dict_obj = {"coffre":"brown", "arbre" :"green", "ennemi":"red","porte":"black", "porte":"grey", "porte_boss":"yellow",
"boss1":"maroon",  "boss":"maroon", "coffre_boss":"brown"}

#niv 2
#dict_case = {"herbe":"green", "murs":"blue", "sols":"grey", "trous":"black"}
#dict_obj = {"coffre":"brown", "Stalagmites" :"cyan", "ennemi":"red", "boss":"maroon", "coffre_boss":"brown",
"porte_boss":"yellow"}

# niv 3
dict_case = {"herbe":"green", "mur":"blue", "sols":"grey", "trous":"black"}
dict_obj = {"coffre":"brown", "Stalagmites" :"cyan", "ennemi":"red","porte":"white", "boss":"maroon"}

def petit(a, b):
    if a<b:
        return a
    else:
        return b

def grand(a, b):
    if a>b:
        return a
    else:
        return b


def ouvrir_fichier():
        global m
        global l
        fichier = tkFileDialog.askopenfilename(title = "ouvrir le fichier", filetypes = [("fichier map", "*.mp")])
        double_mat = []
        with open(fichier, "r") as f:
            mon_pickler = pickle.Unpickler(f)
            double_mat = mon_pickler.load()
        print double_mat
        m = double_mat[0]
        l = double_mat[1]

def effobj():
    global type_case
    type_case = "effobj"
```

```
def lignes():
    for i in range(0,nombre_case*l_case,l_case):
        # les +2 sont à cause de Tkinter qui rajoute 2
        c.create_line(i+2, 0, i+2,nombre_case*l_case+2 , fill = "red")
        c.create_line(0, i+2, nombre_case*l_case+2, i+2, fill = "red")

def creer_fichier():
    with open("carte.mp", "wb") as fichier:
        pick = pickle.Pickler(fichier)
        pick.dump([m, l])

def c_matrice(N):
    m = []
    for i in range(N):
        m.append([])
        for j in range(N):
            m[i].append(type_case)

    return m

def afficher_matrice(m):
    for i in range(nombre_case):
        for j in range(nombre_case):
            case(i, j, dict_case[m[i][j]])


def case(i, j, color):
    c.create_rectangle(i*l_case , j*l_case, (i*l_case)+l_case, (j*l_case)+l_case,fill = color)

def c_matrice2(N):
    l = []
    for x in range(N):
        l.append([])
        for y in range(N):
            l[x].append("")
    return l

def afficher_matrice2(l):
    for i in range(nombre_case):
        for j in range(nombre_case):
            ob = l[i][j]
            if ob!="":
                objet(i, j, dict_obj[ob])

def click(event):
    global pos_debut
    global pos_fin
    pos_debut = (event.x/l_case, event.y/l_case)


def relache_1(event):
    global bouton_1
    global m
    global pos_fin
    pos_fin = (event.x/l_case, event.y/l_case)


    for x in range(petit(pos_debut[0], pos_fin[0]), grand(pos_debut[0], pos_fin[0])+1):
        for y in range(petit(pos_debut[1], pos_fin[1]), grand(pos_debut[1], pos_fin[1])+1):
            if type_case in dict_case.keys():
                m[x][y]=type_case
            elif type_case in dict_obj.keys():
```

```
            l[x][y] = type_case
        elif type_case == "effobj":
            l[x][y] = ""

def objet(x, y, color):
    c.create_rectangle(x*l_case+3, y*l_case+3, (x*l_case)+l_case-3, (y*l_case)+l_case-3,fill = color)

def ch_lb(event):
    global type_case
    global listbox
    try:
        i = listbox.get(listbox.curselection())
        print i
        type_case = i
    except:
        print "probleme"

fenetre = Tk()

c = Canvas(fenetre, width= largeur, height= hauteur, bg= "black", borderwidth = -2)

c.bind("<Button-1>", click)
c.bind("<ButtonRelease-1>", relache_1)

listbox = Listbox(fenetre)
for a in dict_case.keys():
    listbox.insert(END, a)

for a in dict_obj.keys():
    listbox.insert(END, a)

listbox.pack(side = LEFT)

listbox.bind('<Button-1>', ch_lb)

RQ = Button(fenetre, text="quitter", command=fenetre.quit)
RQ.pack()
SAVE = Button(fenetre, text="sauvgarde_carte", command=creer_fichier)
SAVE.pack()
EFF = Button(fenetre, text="effacer objet", command=effobj)
EFF.pack()
LOAD = Button(fenetre, text="charger_carte", command=ouvrir_fichier)
LOAD.pack()


m = c_matrice(nombre_case)
l= c_matrice2(nombre_case)

lignes()

c.pack()

c.update()

fenetre.title('Editeur - Jeu PACSMEN')
while True:
    c.after(vitesse)
    afficher_matrice(m)
    afficher_matrice2(l)
    c.update()
    c.delete(ALL)
```

Editeur_Map/editeurmapniv1.py

```
# -*- coding: utf-8 -*-

from Tkinter import *
import pickle

nombre_case = 50
l_case=10
largeur = nombre_case*l_case
hauteur = nombre_case*l_case

type_case = "herbe"
vitesse = 0
bouton_1 = False

pos_debut = (0, 0)
pos_fin = (0, 0)

dict_case = {"herbe":"green", "mur":"grey"}

dict_obj = {"coffre":"brown", "arbre" :"green", "ennemi":"red","porte":"black"}

def petit(a, b):
    if a<b:
        return a
    else:
        return b

def grand(a, b):
    if a>b:
        return a
    else:
        return b


def effobj():
    global type_case
    type_case = "effobj"


def lignes():
    for i in range(0,nombre_case*l_case,l_case):
        # les +2 sont à cause de Tkinter qui rajoute 2
        c.create_line(i+2, 0, i+2,nombre_case*l_case+2 , fill = "red")
        c.create_line(0, i+2, nombre_case*l_case+2, i+2, fill = "red")

def creer_fichier():
    with open("carte.mp", "wb") as fichier:
        pick = pickle.Pickler(fichier)
        pick.dump([m, l])

def c_matrice(N):
    m = []
    for i in range(N):
        m.append([])
        for j in range(N):
            m[i].append(type_case)

    return m

def afficher_matrice(m):
    for i in range(nombre_case):
```

```
        for j in range(nombre_case):
            case(i, j, dict_case[m[i][j]])


def case(i, j, color):
    c.create_rectangle(i*l_case , j*l_case, (i*l_case)+l_case, (j*l_case)+l_case,fill = color)

def c_matrice2(N):
    l = []
    for x in range(N):
        l.append([])
        for y in range(N):
            l[x].append("")
    return l

def afficher_matrice2(l):
    for i in range(nombre_case):
        for j in range(nombre_case):
            ob = l[i][j]
            if ob!="":
                objet(i, j, dict_obj[ob])

def click(event):
    global pos_debut
    global pos_fin
    pos_debut = (event.x/l_case, event.y/l_case)


def relache_1(event):
    global bouton_1
    global m
    global pos_fin
    pos_fin = (event.x/l_case, event.y/l_case)


    for x in range(petit(pos_debut[0], pos_fin[0]), grand(pos_debut[0], pos_fin[0])+1):
        for y in range(petit(pos_debut[1], pos_fin[1]), grand(pos_debut[1], pos_fin[1])+1):
            if type_case in dict_case.keys():
                m[x][y]=type_case
            elif type_case in dict_obj.keys():
                l[x][y] = type_case
            elif type_case == "effobj":
                l[x][y] = ""

def objet(x, y, color):
    c.create_rectangle(x*l_case+3, y*l_case+3, (x*l_case)+l_case-3, (y*l_case)+l_case-3,fill = color)

def ch_lb(event):
    global type_case
    global listbox
    try:
        i = listbox.get(listbox.curselection())
        print i
        type_case = i
    except:
        print "probleme"

fenetre = Tk()

c = Canvas(fenetre, width= largeur, height= hauteur, bg= "black", borderwidth = -2)

c.bind("<Button-1>", click)
c.bind("<ButtonRelease-1>", relache_1)
```

```
listbox = Listbox(fenetre)
for a in dict_case.keys():
    listbox.insert(END, a)

for a in dict_obj.keys():
    listbox.insert(END, a)

listbox.pack(side = LEFT)

listbox.bind('<Button-1>', ch_lb)

RQ = Button(fenetre, text="quitter", command=fenetre.quit)
RQ.pack()
SAVE = Button(fenetre, text="sauvgarde_carte", command=creer_fichier)
SAVE.pack()
EFF = Button(fenetre, text="effacer objet", command=effobj)
EFF.pack()

m = c_matrice(nombre_case)
print m
afficher_matrice(m)


l= c_matrice2(nombre_case)
afficher_matrice2(l)

lignes()

c.pack()

c.update()

fenetre.title('Editeur - Jeu PACSMEN')
while True:
    c.after(vitesse)
    afficher_matrice(m)
    afficher_matrice2(l)
    c.update()
    c.delete(ALL)
```

Editeur_Map/editeurmapniv2.py

```
# -*- coding: utf-8 -*-

from Tkinter import *
import pickle

nombre_case = 50
l_case=10
largeur = nombre_case*l_case
hauteur = nombre_case*l_case

type_case = "herbe"
vitesse = 0
bouton_1 = False

pos_debut = (0, 0)
pos_fin = (0, 0)

dict_case = {"herbe":"green", "murs":"blue", "sols":"grey", "trous":"black"}

dict_obj = {"coffre":"brown", "Stalagmites" :"cyan", "ennemi":"red","porte":"white"}

def petit(a, b):
    if a<b:
        return a
    else:
        return b

def grand(a, b):
    if a>b:
        return a
    else:
        return b


def effobj():
    global type_case
    type_case = "effobj"


def lignes():
    for i in range(0,nombre_case*l_case,l_case):
        # les +2 sont à cause de Tkinter qui rajoute 2
        c.create_line(i+2, 0, i+2,nombre_case*l_case+2 , fill = "red")
        c.create_line(0, i+2, nombre_case*l_case+2, i+2, fill = "red")

def creer_fichier():
    with open("carte.mp", "wb") as fichier:
        pick = pickle.Pickler(fichier)
        pick.dump([m, l])

def c_matrice(N):
    m = []
    for i in range(N):
        m.append([])
        for j in range(N):
            m[i].append(type_case)

    return m

def afficher_matrice(m):
    for i in range(nombre_case):
```

```
        for j in range(nombre_case):
            case(i, j, dict_case[m[i][j]])


def case(i, j, color):
    c.create_rectangle(i*l_case , j*l_case, (i*l_case)+l_case, (j*l_case)+l_case,fill = color)

def c_matrice2(N):
    l = []
    for x in range(N):
        l.append([])
        for y in range(N):
            l[x].append("")
    return l

def afficher_matrice2(l):
    for i in range(nombre_case):
        for j in range(nombre_case):
            ob = l[i][j]
            if ob!="":
                objet(i, j, dict_obj[ob])

def click(event):
    global pos_debut
    global pos_fin
    pos_debut = (event.x/l_case, event.y/l_case)


def relache_1(event):
    global bouton_1
    global m
    global pos_fin
    pos_fin = (event.x/l_case, event.y/l_case)


    for x in range(petit(pos_debut[0], pos_fin[0]), grand(pos_debut[0], pos_fin[0])+1):
        for y in range(petit(pos_debut[1], pos_fin[1]), grand(pos_debut[1], pos_fin[1])+1):
            if type_case in dict_case.keys():
                m[x][y]=type_case
            elif type_case in dict_obj.keys():
                l[x][y] = type_case
            elif type_case == "effobj":
                l[x][y] = ""

def objet(x, y, color):
    c.create_rectangle(x*l_case+3, y*l_case+3, (x*l_case)+l_case-3, (y*l_case)+l_case-3,fill = color)

def ch_lb(event):
    global type_case
    global listbox
    try:
        i = listbox.get(listbox.curselection())
        print i
        type_case = i
    except:
        print "probleme"

fenetre = Tk()

c = Canvas(fenetre, width= largeur, height= hauteur, bg= "black", borderwidth = -2)

c.bind("<Button-1>", click)
c.bind("<ButtonRelease-1>", relache_1)
```

```
listbox = Listbox(fenetre)
for a in dict_case.keys():
    listbox.insert(END, a)

for a in dict_obj.keys():
    listbox.insert(END, a)

listbox.pack(side = LEFT)

listbox.bind('<Button-1>', ch_lb)

RQ = Button(fenetre, text="quitter", command=fenetre.quit)
RQ.pack()
SAVE = Button(fenetre, text="sauvgarde_carte", command=creer_fichier)
SAVE.pack()
EFF = Button(fenetre, text="effacer objet", command=effobj)
EFF.pack()

m = c_matrice(nombre_case)
print m
afficher_matrice(m)


l= c_matrice2(nombre_case)
afficher_matrice2(l)

lignes()

c.pack()

c.update()

fenetre.title('Editeur - Jeu PACSMEN')
while True:
    c.after(vitesse)
    afficher_matrice(m)
    afficher_matrice2(l)
    c.update()
    c.delete(ALL)
```

Editeur_Map/Emeric.py

```
# -*- coding: utf-8 -*-

from Tkinter import *
import pickle

nombre_case = 50
l_case=10
largeur = nombre_case*l_case
hauteur = nombre_case*l_case

type_case = "herbe"
vitesse = 0
bouton_1 = False

pos_debut = (0, 0)
pos_fin = (0, 0)

dict_case = {"herbe":"green", "mur":"grey"}

dict_obj = {"coffre":"brown", "arbre" :"green"}

def petit(a, b):
    if a<b:
        return a
    else:
        return b

def grand(a, b):
    if a>b:
        return a
    else:
        return b


def effobj():
    global type_case
    type_case = "effobj"


def lignes():
    for i in range(0,nombre_case*l_case,l_case):
        # les +2 sont à cause de Tkinter qui rajoute 2
        c.create_line(i+2, 0, i+2,nombre_case*l_case+2 , fill = "red")
        c.create_line(0, i+2, nombre_case*l_case+2, i+2, fill = "red")

def creer_fichier():
    with open("carte.mp", "wb") as fichier:
        pick = pickle.Pickler(fichier)
        pick.dump([m, l])

def c_matrice(N):
    m = []
    for i in range(N):
        m.append([])
        for j in range(N):
            m[i].append(type_case)

    return m

def afficher_matrice(m):
    for i in range(nombre_case):
```

```
        for j in range(nombre_case):
            case(i, j, dict_case[m[i][j]])


def case(i, j, color):
    c.create_rectangle(i*l_case , j*l_case, (i*l_case)+l_case, (j*l_case)+l_case,fill = color)

def c_matrice2(N):
    l = []
    for x in range(N):
        l.append([])
        for y in range(N):
            l[x].append("")
    return l

def afficher_matrice2(l):
    for i in range(nombre_case):
        for j in range(nombre_case):
            ob = l[i][j]
            if ob!="":
                objet(i, j, dict_obj[ob])

def click(event):
    global pos_debut
    global pos_fin
    pos_debut = (event.x/l_case, event.y/l_case)


def relache_1(event):
    global bouton_1
    global m
    global pos_fin
    pos_fin = (event.x/l_case, event.y/l_case)


    for x in range(petit(pos_debut[0], pos_fin[0]), grand(pos_debut[0], pos_fin[0])+1):
        for y in range(petit(pos_debut[1], pos_fin[1]), grand(pos_debut[1], pos_fin[1])+1):
            if type_case in dict_case.keys():
                m[x][y]=type_case
            elif type_case in dict_obj.keys():
                l[x][y] = type_case
            elif type_case == "effobj":
                l[x][y] = ""

def objet(x, y, color):
    c.create_rectangle(x*l_case+3, y*l_case+3, (x*l_case)+l_case-3, (y*l_case)+l_case-3,fill = color)

def ch_lb(event):
    global type_case
    global listbox
    try:
        i = listbox.get(listbox.curselection())
        print i
        type_case = i
    except:
        print "probleme"

fenetre = Tk()

c = Canvas(fenetre, width= largeur, height= hauteur, bg= "black", borderwidth = -2)

c.bind("<Button-1>", click)
c.bind("<ButtonRelease-1>", relache_1)
```

```
listbox = Listbox(fenetre)
for a in dict_case.keys():
    listbox.insert(END, a)

for a in dict_obj.keys():
    listbox.insert(END, a)

listbox.pack(side = LEFT)

listbox.bind('<Button-1>', ch_lb)

RQ = Button(fenetre, text="quitter", command=fenetre.quit)
RQ.pack()
SAVE = Button(fenetre, text="sauvgarde_carte", command=creer_fichier)
SAVE.pack()
EFF = Button(fenetre, text="effacer objet", command=effobj)
EFF.pack()

m = c_matrice(nombre_case)
print m
afficher_matrice(m)


l= c_matrice2(nombre_case)
afficher_matrice2(l)

lignes()

c.pack()

c.update()

fenetre.title('Editeur - Jeu PACSMEN')
while True:
    c.after(vitesse)
    afficher_matrice(m)
    afficher_matrice2(l)
    c.update()
    c.delete(ALL)
```

Editeur_Map/Evy.py

```
# -*- coding: utf-8 -*-
"""
Éditeur de Spyder

Ce script temporaire est sauvegardé ici :
C:\Users\sebastien.thao\.spyder2\.temp.py
"""

# -*- coding: utf-8 -*-

from Tkinter import *
#pickle pour sauvegarder les fichiers et les réutiliser
import pickle

#variables
nombre_case = 50
l_case=10
largeur = nombre_case*l_case
hauteur = nombre_case*l_case

type_case = "herbe"

#création du canevas
fenetre = Tk()
c = Canvas(fenetre, width= largeur, height= hauteur, bg= "black", borderwidth = 0)
c.pack(side = "left")
c.update()

vitesse = 1



def lignes():
    for i in range(0,nombre_case*l_case,l_case):
        c.create_line(i+2, 0, i+2,nombre_case*l_case+2 , fill = "red")
        c.create_line(0, i+2, nombre_case*l_case+2, i+2, fill = "red")
#fonctionclick qui réagira au clic de la souris
def click(event):
    global m
    x = event.x/l_case
    y = event.y/l_case
    if type_case in ["herbe", "mur"]:
        m[x][y]=type_case
    elif type_case in ["coffre", "arbre", "pnj"]:
        l[x][y] = type_case

#definition de fonction qui permettront de différencier chaque type de case

def mur():
    global type_case
    type_case = "mur"

def herbe():
    global type_case
    type_case = "herbe"

def coffre():
    global type_case
    type_case = "coffre"

def arbre():
```

```
    global type_case
    type_case = "arbre"

def pnj():
    global type_case
    type_case = "pnj"


def creer_fichier():
    with open("map1.mp", "wb") as fichier:
        #créer un objet pickler de paramètre fichier
        print m
        pick = pickle.Pickler(fichier)
        #prends les 2 matrices m et l pour les mettre dans une liste
        m2= [m,l]
        #on utilise les 2 matrices
        pick.dump(m2)

def import_fichier():
    global m
    global l
    #pour lire le fichier que l'on a sauvegardé sous le nom map1.mp
    ml = pickle.load(open( "map1.mp", "rb"))
    #comme ml contient deux matrice m et m on lui redis dans la fonction quel argument de la liste signifie quelle
matrice
    print ml
    m = ml[0]
    l = ml[1]


#création de la matrice qui permettra de donner un type de case
def c_matrice(N):
    m = []
    for i in range(N):
        m.append([])
        for j in range(N):
            m[i].append(type_case)
    return m

def afficher_matrice(m):
    for i in range(nombre_case):
        for j in range(nombre_case):
            if m[i][j] == "herbe":
                case(i,j, "green")
            elif m[i][j] == "mur":
                case(i,j, "brown")

#création de la matrice "objet" qui apparaîtra sur la case
def c_matrice2(N):
    l = []
    for x in range(N):
        l.append([])
        for y in range(N):
            l[x].append("")
    return l

def afficher_matrice2(l):
    for x in range(nombre_case):
        for y in range(nombre_case):
            if l[x][y] == "coffre":
                objet(x,y, "red")
            elif l[x][y] == "arbre":
                objet(x,y, "blue" )
```

```
            elif l[x][y] == "pnj":
                objet(x,y, "black" )

#definition de la fonction case
def case(i, j, color):
    c.create_rectangle(i*l_case , j*l_case, (i*l_case)+l_case, (j*l_case)+l_case,fill = color)
#definition de la fonction objet qui est sur la case
def objet(x, y, color):
    c.create_rectangle(x*l_case+3, y*l_case+3, (x*l_case)+l_case-3, (y*l_case)+l_case-3,fill = color)

#Boutons qui seront affichés dans le canvas
Mur = Button(fenetre, text="mur", command=mur)
Mur.pack(side= "top")
Herb = Button(fenetre, text="herbe", command=herbe)
Herb.pack(side= "top")
Tree = Button(fenetre, text="arbre", command=arbre)
Tree.pack()
Coffre = Button(fenetre, text="coffre",command=coffre)
Coffre.pack()
Pnj = Button(fenetre, text="Pnj", command=pnj)
Pnj.pack()
RQ = Button(fenetre, text="quitter", command=fenetre.destroy)
RQ.pack()
SAVE = Button(fenetre, text="sauvegarde_carte", command=creer_fichier)
SAVE.pack()
imp = Button(fenetre, text="map_1", command=import_fichier)
imp.pack()

lignes()
m= c_matrice(nombre_case)
l= c_matrice2(nombre_case)
afficher_matrice(m)
afficher_matrice2(l)
c.bind("<Button-1>", click)
fenetre.title("Editeur - Jeu PACSMEN")

#boucle while qui permet de relancer à chaque mouvement une nouvelle carte
while True:
    c.after(vitesse)
    afficher_matrice(m)
    afficher_matrice2(l)
    c.update()
    c.delete(ALL)

fenetre.mainloop()
```

Editeur_Map/testmap3.py

```
# -*- coding: utf-8 -*-

from Tkinter import *
import pickle
import tkFileDialog

nombre_case = 50
l_case=10
largeur = nombre_case*l_case
hauteur = nombre_case*l_case

type_case = "herbe"
vitesse = 0
bouton_1 = False

pos_debut = (0, 0)
pos_fin = (0, 0)

dict_case = {"herbe":"green", "murs":"blue", "sols":"grey", "trous":"black"}

dict_obj = {"coffre":"brown", "Stalagmites" :"cyan", "ennemi":"red","porte":"white"}

def petit(a, b):
    if a<b:
        return a
    else:
        return b

def grand(a, b):
    if a>b:
        return a
    else:
        return b

def ouvrir_fichier():
        global m
        global l
        fichier = tkFileDialog.askopenfilename(title = "ouvrir le fichier", filetypes = [("fichier map", "*.mp")])
        double_mat = []
        with open(fichier, "rb") as f:
            mon_pickler = pickle.Unpickler(f)
            double_mat = mon_pickler.load()
        print double_mat
        m = double_mat[0]
        l = double_mat[1]

def effobj():
    global type_case
    type_case = "effobj"

def lignes():
    for i in range(0,nombre_case*l_case,l_case):
        # les +2 sont à cause de Tkinter qui rajoute 2
        c.create_line(i+2, 0, i+2,nombre_case*l_case+2 , fill = "red")
        c.create_line(0, i+2, nombre_case*l_case+2, i+2, fill = "red")

def creer_fichier():
    with open("carte.mp", "wb") as fichier:
        pick = pickle.Pickler(fichier)
        pick.dump([m, l])
```

```
def c_matrice(N):
    m = []
    for i in range(N):
        m.append([])
        for j in range(N):
            m[i].append(type_case)

    return m

def afficher_matrice(m):
    for i in range(nombre_case):
        for j in range(nombre_case):
            case(i, j, dict_case[m[i][j]])

def case(i, j, color):
    c.create_rectangle(i*l_case , j*l_case, (i*l_case)+l_case, (j*l_case)+l_case,fill = color)

def c_matrice2(N):
    l = []
    for x in range(N):
        l.append([])
        for y in range(N):
            l[x].append("")
    return l

def afficher_matrice2(l):
    for i in range(nombre_case):
        for j in range(nombre_case):
            ob = l[i][j]
            if ob!="":
                objet(i, j, dict_obj[ob])

def click(event):
    global pos_debut
    global pos_fin
    pos_debut = (event.x/l_case, event.y/l_case)

def relache_1(event):
    global bouton_1
    global m
    global pos_fin
    pos_fin = (event.x/l_case, event.y/l_case)

    for x in range(petit(pos_debut[0], pos_fin[0]), grand(pos_debut[0], pos_fin[0])+1):
        for y in range(petit(pos_debut[1], pos_fin[1]), grand(pos_debut[1], pos_fin[1])+1):
            if type_case in dict_case.keys():
                m[x][y]=type_case
            elif type_case in dict_obj.keys():
                l[x][y] = type_case
            elif type_case == "effobj":
                l[x][y] = ""

def objet(x, y, color):
    c.create_rectangle(x*l_case+3, y*l_case+3, (x*l_case)+l_case-3, (y*l_case)+l_case-3,fill = color)

def ch_lb(event):
    global type_case
    global listbox
    try:
        i = listbox.get(listbox.curselection())
        print i
        type_case = i
    except:
```

```
        print "probleme"

fenetre = Tk()

c = Canvas(fenetre, width= largeur, height= hauteur, bg= "black", borderwidth = -2)

c.bind("<Button-1>", click)
c.bind("<ButtonRelease-1>", relache_1)

listbox = Listbox(fenetre)
for a in dict_case.keys():
    listbox.insert(END, a)

for a in dict_obj.keys():
    listbox.insert(END, a)

listbox.pack(side = LEFT)

listbox.bind('<Button-1>', ch_lb)

RQ = Button(fenetre, text="quitter", command=fenetre.quit)
RQ.pack()
SAVE = Button(fenetre, text="sauvgarde_carte", command=creer_fichier)
SAVE.pack()
EFF = Button(fenetre, text="effacer objet", command=effobj)
EFF.pack()
LOAD = Button(fenetre, text="charger_carte", command=ouvrir_fichier)
LOAD.pack()
m = c_matrice(nombre_case)
print m
afficher_matrice(m)


l= c_matrice2(nombre_case)
afficher_matrice2(l)

lignes()

c.pack()

c.update()

fenetre.title('Editeur - Jeu PACSMEN')
while True:
    c.after(vitesse)
    afficher_matrice(m)
    afficher_matrice2(l)
    c.update()
    c.delete(ALL)
```

RC/ElProyectoRubiksCube.py (contribution de Julien Giraud)

```
#-----------------RUBIK'S CUBE-------------------------
import pygame
import random
import time
pygame.init()

#-----------------CREATION DU CUBE ET DES LISTE DE VARIABLES----------

cube=[['C3N', 'A4N', 'C4N', 'A2N', 'M1N', 'A3N', 'C1N', 'A1N', 'C2N'],
     ['C3R', 'A4R', 'C4R', 'A2R', 'M1R', 'A3R', 'C1R', 'A1R', 'C2R'],
     ['C3B', 'A4B', 'C4B', 'A2B', 'M1B', 'A3B', 'C1B', 'A1B', 'C2B'],
     ['C3O', 'A4O', 'C4O', 'A2O', 'M1O', 'A3O', 'C1O', 'A1O', 'C2O'],
     ['C3V', 'A4V', 'C4V', 'A2V', 'M1V', 'A3V', 'C1V', 'A1V', 'C2V'],
     ['C3J', 'A4J', 'C4J', 'A2J', 'M1J', 'A3J', 'C1J', 'A1J', 'C2J'],
     [['', '', '']]]

couleur1={'noir':(0, 0, 0),
          'blanc':(255, 255, 255),
          'rouge':(255, 0, 0),
          'bleu':(0, 0, 255),
          'orange':(255, 128, 0),
          'vert':(0, 255, 0),
          'jaune':(255, 255, 0)}

couleur2={'N':'blanc',
          'R':'rouge',
          'B':'bleu',
          'O':'orange',
          'V':'vert',
          'J':'jaune'}

xy_cube=[10, 55, 100, 145, 190, 235, 280, 325, 370, 415, 460, 505, 550, 595, 640]

dxy_cube={'gx':15, 'gy':-45,
     'nx':-45, 'ny':45+12,
     'rx':0, 'ry':0,
     'bx':-45,'by':-30,
     'ox':-45*3+12, 'oy':-8,
     'vx':45, 'vy':0,
     'jx':0, 'jy':-45}

mouv=["R", "R'", "U", "U'", "L", "L'", "D", "D'", "F", "F'", "B", "B'"]

#listes de coordonnees pour la fonction affichage
#(et oui c'est de la compression manuelle et c'est extremement moche)
aff=[0, 14, 12, 10, 18]+[0]*4+[8]*3+[0, 6]+[0]*3+[2]+[4]*3+[6]+[5]*3
aff+=[6, 0, 0, 3, 6, 1, 0, 3, 6, 0, 0, 3, 6, 0, 6]+[0]*3+[2]+[4]*3
aff+=[6]+[5]*3+[6, 0, 1, 4, 7, 5]+[1, 4, 7, 0, 1, 4, 7, 0]+[6]+[0]*3
aff+=[6]+[4]*3+[6]+[5]*3+[6, 0]+[2, 5, 8, 0]*3+[0]*3+[4, 1]+[2]*3+[0]*5
aff+=[6]*3+[2, 3]+[1]*3+[6]*5+[0]*3+[2, 3, 0, 3, 6]+[0]*5+[6]*4+[3]+[1]*3
aff+=[6]*5+[0]*4+[4, 1, 4, 7]+[0]*5+[6]*4+[3]+[1]*3+[6]*5+[0]*4+[5, 2, 5, 8]
aff+=[0]*10+[2]*3+[0]*5+[6]*3+[3, 6]+[2]*3+[6]*5+[0]*6+[3, 6]+[0]*5+[6]*3
aff+=[3, 6, 2, 2]+[6]*6+[0]*3+[1, 0, 4, 7]+[0]*6+[6]*3+[3, 6, 2 ]+[6]*7+[0]*3
aff+=[2, 0, 8]+[0]*12+[2]+[0]*7+[6]*5+[3]+[6]*7+[0]*5+[6]+[0]*7+[6]*5+[3]
aff+=[6]*7+[0]*5+[7]+[0]*7+[6]*5+[3]+[6]*7+[0]*5+[8]+[0]*7
listg=[6]*3+[17]*3+[28]*3+([0]+[-11]+[-22])*3+[7, 8, 9, 6, 7, 8, 5, 6, 7]
listg+=([8]+[9]+[10])*3+[22]*3+[11]*3+[0]*3+([28, 39, 50])*3+[0]*3+[1]*3
listg+=[2]*3+[4, 3, 2, 5, 4, 3, 6, 5, 4]+[46]*3+[57]*3+[68]*3+([28]+[17]+[6])*3
listg+=[22]*3+[11]*3+[0]*3+([0]+[11]+[22])*3+[18]*3+[29]*3+[40]*3
listg+=([28]+[17]+[6])*3+[50]*3+[39]*3+[28]*3+([40]+[51]+[62])*3+[-22]*3
listg+=[-11]*3+[0]*3+([0]+[-11]+[-22])*3+[50]*3+[39]*3+[28]*3+([68]+[79]+[90])*3
```

```
listg+=([4]+[5]+[6])*3+[4]*3+[5]*3+[6]*3+([12]+[13]+[14])*3+[2]*3+[3]*3+[4]*3
listg+=([0]+[1]+[2])*3+[4]*3+[5]*3+[6]*3+([4]+[5]+[6])*3+[8]*3+[9]*3+[10]*3

#listes de valeurs pour les rotations/mouvement de base du cube
var_cube=[]
var1_cube=[]
for i in range(54):
    var_cube.append('')
    var1_cube.append(i)
var2_cube=[0]*9+[1]*9+[2]*9+[3]*9+[4]*9+[5]*9
var3_cube=[0, 1, 2, 3, 4, 5, 6, 7, 8]*6
varu=[0]*9+[1]*3+[2]*3+[3]*3+[4]*3+[0, 1, 2, 3, 4, 5, 6, 7, 8]+[0, 1, 2]*4
varu+=[6, 3, 0, 7, 4, 1, 8, 5, 2, 12, 13, 14, 15, 16, 17, 18, 19, 20, 9, 10, 11]
varf=[42, 39, 36, 43, 40, 37, 44, 41, 38, 15, 12, 9, 16, 13, 10, 17, 14, 11, 6]
varf+=[3, 0, 7, 4, 1, 8, 5, 2, 29, 32, 35, 28, 31, 34, 27, 30, 33, 51, 48, 45]
varf+=[52, 49, 46, 53, 50, 47, 24, 21, 18, 25, 22, 19, 26, 23, 20]
varr=[9, 10, 11, 12, 13, 14, 15, 16, 17, 45, 46, 47, 48, 49, 50, 51, 52, 53]
varr+=[24, 21, 18, 25, 22, 19, 26, 23, 20, 8, 7, 6, 5, 4, 3, 2, 1, 0, 38, 41]
varr+=[44, 37, 40, 43, 36, 39, 42, 35, 34, 33, 32, 31, 30, 29, 28, 27]

#------------------FIN DES LISTES------------------

#----------DEBUT DES FONCTIONS DE MOUVEMENT-----------

def fB3(): # f B' en langage universel (ou fw)
    for i in range(54):
        var_cube[var1_cube[i]]=cube[var2_cube[i]][var3_cube[i]]
    for i in range(54):
        cube[var2_cube[i]][var3_cube[i]]=var_cube[varf[i]]

def fB33(): # fw' en langage universel
    for i in range(3):
        fB3()

def rL3(): # r L' en langage universel (ou rw)
    for i in range(54):
        var_cube[var1_cube[i]]=cube[var2_cube[i]][var3_cube[i]]
    for i in range(54):
        cube[var2_cube[i]][var3_cube[i]]=var_cube[varr[i]]

def rL33(): # rw' en langage universel
    for i in range(3):
        rL3()

def uD3(): # u D' en langage universel (ou uw)
    rL33()
    fB3()
    rL3()

def uD33(): # uw' en langage universel
    for i in range(3):
        uD3()

def U1(): # U en langage universel
    for i in range(21):
        var_cube[i]=cube[varu[i]][varu[i+21]]
    for i in range(21):
        cube[varu[i]][varu[i+21]]=var_cube[varu[i+21*2]]

def U3(): # U' en langage universel
    for i in range(3):
        U1()
```

```
def u1(): # u en langage universel
    D1()
    uD3()

def u3(): # u' en langage universel
    for i in range(3):
        u1()

def L1(): # L en langage universel
    fB3()
    U1()
    for i in range(3):
        fB3()

def L3(): # L' en langage universel
    for i in range(3):
        L1()

def l1(): # l en langage universel
    R1()
    for i in range(3):
        rL3()

def l3(): # l' en langage universel
    for i in range(3):
        l1()

def F1(): # F en langage universel
    rL3()
    U1()
    for i in range(3):
        rL3()

def F3(): # F' en langage universel
    for i in range(3):
        F1()

def f1(): # f en langage universel
    B1()
    fB3()

def f3(): # f' en langage universel
    for i in range(3):
        f1()

def R1(): # R en langage universel
    for i in range(3):
        fB3()
    U1()
    fB3()

def R3(): # R' en langage universel
    for i in range(3):
        R1()

def r1(): # r en langage universel
    L1()
    rL3()

def r3(): # r' en langage universel
    for i in range(3):
        r1()
```

```
def B1(): # B en langage universel
    for i in range(3):
        rL3()
    U1()
    rL3()

def B3(): # B' en langage universel
    for i in range(3):
        B1()

def b1(): # b en langage universel
    F1()
    for i in range(3):
        fB3()

def b2(): # b2 en langage universel
    b1()
    b1()

def b3(): # b' en langage universel
    for i in range(3):
        b1()

def D1(): # D en langage universel
    fB3()
    fB3()
    U1()
    fB3()
    fB3()

def D3(): # D' en langage universel
    for i in range(3):
        D1()

def d1(): # d en langage universel
    U1()
    for i in range(3):
        uD3()

def d3(): # d' en langage universel
    for i in range(3):
        d1()

def M1(): # M' en langage universel mais chez moi c'est M
    r1()
    R3()

def M3(): # M en langage universel mais chez moi c'est M'
    for i in  range(3):
        M1()

#dictionnaire des fonctions de mouvements
dico_cube={"R":R1, "R'":R3, "r":r1, "r'":r3,
      "U":U1, "U'":U3, "u":u1, "u'":u3,
      "L":L1, "L'":L3, "l":l1, "l'":l3,
      "D":D1, "D'":D3, "d":d1, "d'":d3,
      "F":F1, "F'":F3, "f":f1, "f'":f3,
      "B":B1, "B'":B3, "b":b1, "b'":b3,
      "M":M1, "M'":M3,
      "rw":rL3, "rw'":rL33,
      "uw":uD3, "uw'":uD33,
      "fw":fB3, "fw'":fB33}
```

```
#----------FIN DES FONCTIONS DE MOUVEMENT----------

#----------DEBUT DES FONCTIONS D'AIDE----------

def afficher_cube():
    for i in range(9):
        pygame.draw.polygon(fenetre, couleur1[couleur2[cube[0][i][2]]], [[listg[i+18*0]+xy_cube[
        listg[i+18*1]]+dxy_cube['gx']+dxy_cube['nx'], listg[i+18*2]+xy_cube[listg[i+18*3]]+dxy_cube['gy']+dxy_cube[
        'ny']], [listg[i+18*4]+xy_cube[listg[i+18*1]]+dxy_cube['gx']+dxy_cube['nx'],listg[i+18*5]+xy_cube[listg[
        i+18*3]]+dxy_cube['gy']+dxy_cube['ny']], [listg[i+18*6]+xy_cube[listg[i+18*1]]+dxy_cube['gx']+dxy_cube['nx'
        ],listg[i+18*7]+xy_cube[listg[i+18*3]]+dxy_cube['gy']+dxy_cube['ny']], [listg[i+18*8]+xy_cube[listg[i+18*1]
        ]+dxy_cube['gx']+dxy_cube['nx'],listg[i+18*9]+xy_cube[listg[i+18*3]]+dxy_cube['gy']+dxy_cube['ny']]])
        pygame.draw.polygon(fenetre, couleur1[couleur2[cube[1][i][2]]], [[xy_cube[listg[i+18*10]
        ]+dxy_cube['gx']+dxy_cube['rx'], xy_cube[listg[i+18*10+9]]+dxy_cube['gy']+dxy_cube['ry']],
[xy_cube[listg[i+18*10]
        ]+dxy_cube['gx']+dxy_cube['rx']+40, xy_cube[listg[i+18*10+9]]+dxy_cube['gy']+dxy_cube['ry']],
[xy_cube[listg[i+18*10]
        ]+dxy_cube['gx']+dxy_cube['rx']+40, xy_cube[listg[i+18*10+9]]+dxy_cube['gy']+dxy_cube['ry']+40],
[xy_cube[listg[i+18*10]
        ]+dxy_cube['gx']+dxy_cube['rx'], xy_cube[listg[i+18*10+9]]+dxy_cube['gy']+dxy_cube['ry']+40]])
        pygame.draw.polygon(fenetre, couleur1[couleur2[cube[2][i][2]]], [[listg[i+18*0+9]+xy_cube[
        listg[i+18*1+9]]+dxy_cube['gx']+dxy_cube['bx'], listg[i+18*2+9]+xy_cube[listg[i+18*3+9]]+dxy_cube['gy']
+dxy_cube[
        'by']], [listg[i+18*4+9]+xy_cube[listg[i+18*1+9]]+dxy_cube['gx']+dxy_cube['bx'],listg[i+18*5+9]+xy_cube[
        listg[i+18*3+9]]+dxy_cube['gy']+dxy_cube['by']], [listg[i+18*6+9]+xy_cube[listg[i+18*1+9]]+dxy_cube['gx']
+dxy_cube[
        'bx'],listg[i+18*7+9]+xy_cube[listg[i+18*3+9]]+dxy_cube['gy']+dxy_cube['by']], [listg[i+18*8+9]+xy_cube[listg[
        i+18*1+9]]+dxy_cube['gx']+dxy_cube['bx'],listg[i+18*9+9]+xy_cube[listg[i+18*3+9]]+dxy_cube['gy']
+dxy_cube['by']]])
        pygame.draw.polygon(fenetre, couleur1[couleur2[cube[3][i][2]]], [[xy_cube[listg[i+18*11]
        ]+dxy_cube['gx']+dxy_cube['ox'], xy_cube[listg[i+18*11+9]]+dxy_cube['gy']+dxy_cube['oy']],
[xy_cube[listg[i+18*11]
        ]+dxy_cube['gx']+dxy_cube['ox']+40, xy_cube[listg[i+18*11+9]]+dxy_cube['gy']+dxy_cube['oy']],
[xy_cube[listg[i+18*11]
        ]+dxy_cube['gx']+dxy_cube['ox']+40, xy_cube[listg[i+18*11+9]]+dxy_cube['gy']+dxy_cube['oy']+40],
[xy_cube[listg[i+18*11]
        ]+dxy_cube['gx']+dxy_cube['ox'], xy_cube[listg[i+18*11+9]]+dxy_cube['gy']+dxy_cube['oy']+40]])
        pygame.draw.polygon(fenetre, couleur1[couleur2[cube[4][i][2]]], [[xy_cube[listg[i+18*12]
        ]+dxy_cube['gx']+dxy_cube['vx'], xy_cube[listg[i+18*12+9]]+dxy_cube['gy']+dxy_cube['vy']],
[xy_cube[listg[i+18*12]
        ]+dxy_cube['gx']+dxy_cube['vx']+40, xy_cube[listg[i+18*12+9]]+dxy_cube['gy']+dxy_cube['vy']],
[xy_cube[listg[i+18*12]
        ]+dxy_cube['gx']+dxy_cube['vx']+40, xy_cube[listg[i+18*12+9]]+dxy_cube['gy']+dxy_cube['vy']+40],
[xy_cube[listg[i+18*12]
        ]+dxy_cube['gx']+dxy_cube['vx'], xy_cube[listg[i+18*12+9]]+dxy_cube['gy']+dxy_cube['vy']+40]])
        pygame.draw.polygon(fenetre, couleur1[couleur2[cube[5][i][2]]], [[xy_cube[listg[i+18*13]
        ]+dxy_cube['gx']+dxy_cube['jx'], xy_cube[listg[i+18*13+9]]+dxy_cube['gy']+dxy_cube['jy']],
[xy_cube[listg[i+18*13]
        ]+dxy_cube['gx']+dxy_cube['jx']+40, xy_cube[listg[i+18*13+9]]+dxy_cube['gy']+dxy_cube['jy']],
[xy_cube[listg[i+18*13]
        ]+dxy_cube['gx']+dxy_cube['jx']+40, xy_cube[listg[i+18*13+9]]+dxy_cube['gy']+dxy_cube['jy']+40],
[xy_cube[listg[i+18*13]
        ]+dxy_cube['gx']+dxy_cube['jx'], xy_cube[listg[i+18*13+9]]+dxy_cube['gy']+dxy_cube['jy']+40]])
        pygame.display.flip()

def lire_algo(a_cube):
    algo=''
    for i in a_cube:
        if i!=' ':
            algo+=i
        else:
            try:
                dico_cube[algo]()
```

```
            except:
                print 'Erreur.'
            algo=''

def etat_cube():
    a_cube=0
    for i in range(6):
        for j in range(9):
            if cube[i][j][2]==cube[i][0][2]:
                a_cube*=1
            else:
                a_cube+=1
    if a_cube==0:
        return 1

#creation de la fenetre d'affichage
fenetre = pygame.display.set_mode((640,640))
pygame.display.set_caption("----------------------------------------TRY TO SOLVE THE
CUBE----------------------------------------")
fenetre.fill(couleur1['noir'])

#----------DEBUT DU CODE PRINCIPAL-----------

mv = pygame.image.load("tableau plein.png").convert()
fenetre.blit(mv, (88,432))
rg = pygame.image.load("rage.png").convert()
fenetre.blit(rg, (382,240))

cube_continuer=1
a_cube=""
for i in range(50):
    a_cube+=mouv[random.randint(0,11)]+' '
lire_algo(a_cube)

afficher_cube()
while cube_continuer:
    for event in pygame.event.get():
        a_cube=""
        if event.type == pygame.QUIT:
            cube_continuer = 0
        elif event.type == pygame.MOUSEBUTTONDOWN:
            if event.pos[0]>381 and event.pos[0]<582 and event.pos[1]>239 and event.pos[1]<388 and event.button==1:
                cube_continuer=0
            elif event.pos[0]>91 and event.pos[0]<148 and event.pos[1]>435 and event.pos[1]<492:
                if event.button==1:
                    a_cube="R "
                elif event.button==3:
                    a_cube="R' "
            elif event.pos[0]>151 and event.pos[0]<208 and event.pos[1]>435 and event.pos[1]<492:
                if event.button==1:
                    a_cube="U "
                elif event.button==3:
                    a_cube="U' "
            elif event.pos[0]>211 and event.pos[0]<268 and event.pos[1]>435 and event.pos[1]<492:
                if event.button==1:
                    a_cube="F "
                elif event.button==3:
                    a_cube="F' "
            elif event.pos[0]>271 and event.pos[0]<328 and event.pos[1]>435 and event.pos[1]<492:
                if event.button==1:
                    a_cube="L "
                elif event.button==3:
                    a_cube="L' "
```

```
elif event.pos[0]>331 and event.pos[0]<388 and event.pos[1]>435 and event.pos[1]<492:
    if event.button==1:
        a_cube="D "
    elif event.button==3:
        a_cube="D' "
elif event.pos[0]>391 and event.pos[0]<448 and event.pos[1]>435 and event.pos[1]<492:
    if event.button==1:
        a_cube="B "
    elif event.button==3:
        a_cube="B' "
elif event.pos[0]>91 and event.pos[0]<148 and event.pos[1]>495 and event.pos[1]<552:
    if event.button==1:
        a_cube="r "
    elif event.button==3:
        a_cube="r' "
elif event.pos[0]>151 and event.pos[0]<208 and event.pos[1]>495 and event.pos[1]<552:
    if event.button==1:
        a_cube="u "
    elif event.button==3:
        a_cube="u' "
elif event.pos[0]>211 and event.pos[0]<268 and event.pos[1]>495 and event.pos[1]<552:
    if event.button==1:
        a_cube="f "
    elif event.button==3:
        a_cube="f' "
elif event.pos[0]>271 and event.pos[0]<328 and event.pos[1]>495 and event.pos[1]<552:
    if event.button==1:
        a_cube="l "
    elif event.button==3:
        a_cube="l' "
elif event.pos[0]>331 and event.pos[0]<388 and event.pos[1]>495 and event.pos[1]<552:
    if event.button==1:
        a_cube="d "
    elif event.button==3:
        a_cube="d' "
elif event.pos[0]>391 and event.pos[0]<448 and event.pos[1]>495 and event.pos[1]<552:
    if event.button==1:
        a_cube="M "
    elif event.button==3:
        a_cube="M' "
elif event.pos[0]>91 and event.pos[0]<208 and event.pos[1]>553 and event.pos[1]<612:
    if event.button==1:
        a_cube="rw "
    elif event.button==3:
        a_cube="rw' "
elif event.pos[0]>211 and event.pos[0]<328 and event.pos[1]>553 and event.pos[1]<612:
    if event.button==1:
        a_cube="uw "
    elif event.button==3:
        a_cube="uw' "
elif event.pos[0]>331 and event.pos[0]<448 and event.pos[1]>553 and event.pos[1]<612:
    if event.button==1:
        a_cube="fw "
    elif event.button==3:
        a_cube="fw' "

lire_algo(a_cube)
afficher_cube()
if etat_cube()==1:
    print 'VICTOIRE'
    time.sleep(3)
    #----------------------------Martin c'est pour toi-----------------------------------------
    cube_continuer=0 #Fermeture du programme
```