



BRANCHE	SECTION(S)	ÉPREUVE ÉCRITE
INFORMATIQUE	B	<i>Durée de l'épreuve : 3 heures</i> <i>Date de l'épreuve :</i>

```
# import section -----
import pygame, sys
from pygame.locals import *
from random import randint
from math import sqrt
from copy import deepcopy
# classes section -----
class Element:      #-----
#-----
    def __init__(self, x, y, fill_color, border_color):
        self.x = x                # coordinates
        self.y = y
        self.fill_color = fill_color    # colors
        self.border_color = border_color
#-----
    def draw(self):
        pygame.draw.circle(screen, self.fill_color, (self.x, self.y), 7, 0) # inside
        pygame.draw.circle(screen, self.border_color, (self.x, self.y), 7, 1) # border
#-----
    def move(self, dx, dy):                # move the element by dx, dy of pixels
        self.x += dx
        self.y += dy
#-----
    def has_touched(self, other):
        return sqrt((self.x - other.x)**2 + (self.y - other.y)**2) < 10
#-----
class Snake:      #-----
#-----
    def __init__(self, x, y):
        self.head = Element(x, y, Color("purple1"), Color("purple4")) # create a purple head
        self.body = []          # create the body list
        for i in range(1,6):    # and add five elements
            self.body.append(Element(x + i*10, y, Color("gold1"), Color("gold4")))
        self.energy = 200        # set energy to 200
        self.direction = (-1,0) # set direction to left
#-----
    def move(self):
        new_element = Element(self.head.x, self.head.y, Color("gold1"), Color("gold3"))
        self.body.insert(0, new_element) # insert new element with head coords
        self.body.pop()                 # cut one element from body tail
        dx, dy = self.direction         # get the directions
        dx, dy = 10*dx, 10*dy           # multiply by the number of pixels to move
        self.head.move(dx, dy)          # move the snake accordingly
        self.energy -= 1                # reduce energy by one
#-----
    def draw(self):
        pygame.draw.rect(screen, Color("blue"), (5,5,200,5)) # draw red backgrd
        pygame.draw.rect(screen, Color("yellow"), (5,5,self.energy,5)) # draw green e-bar
        for element in self.body:      # draw the body
            element.draw()
        self.head.draw()               # draw the head
#-----
    def grow(self):
        for _ in range(8):              # add 8 elements to body tail
            self.body.append(deepcopy(self.body[-1]))
#-----
    def has_hit_wall(self, max_x, max_y): # test distance to the four walls
        return (self.head.x < 7) or (self.head.x >= max_x - 7) \
            or (self.head.y < 7) or (self.head.y >= max_y - 7)
#-----
    def has_touched(self, elements):
        hit = False                    # nothing yet been hit...
        for element in elements:      # for any element in the list:
            if self.head.has_touched(element): # does the head touch it?
                hit = True             # then we have a hit
        return hit                    # return the result
```

```

#-----
# pygame section -----
#-----
pygame.init()
game_color = Color("white")
screen_width, screen_height = 600, 400
screen = pygame.display.set_mode((screen_width, screen_height))
pygame.display.set_caption("Snake - Lycée - Numéro")
screen.fill(game_color)
FPS = 8
clock = pygame.time.Clock()
done = False
is_running = False
# main pygame loop -----
while not done:
    for event in pygame.event.get():
        # quit game -----
        if event.type == QUIT:
            done = True
        # game and snake control -----
        if event.type == KEYDOWN:
            if event.key == K_n:
                # start a new game
                # create snake -----
                sx = randint(100, screen_width - 100) // 10 * 10 # stay 100px from borders
                sy = randint(100, screen_height - 100) // 10 * 10
                snake = Snake(sx, sy) # create snake
                # create food -----
                food = [] # create 8 static food elements
                for _ in range(8):
                    fx = randint(20, screen_width - 20) // 10 * 10
                    fy = randint(20, screen_height - 20) // 10 * 10
                    food.append(Element(fx, fy, Color("orange1"), Color("orange4")))
                # create poison -----
                poison = [] # create 20 static poison elements
                for _ in range(20):
                    px = randint(20, screen_width - 20) // 10 * 10
                    py = randint(20, screen_height - 20) // 10 * 10
                    poison.append(Element(px, py, Color("cyan1"), Color("cyan4")))
                # start the game -----
                is_running = True
                game_color = Color("white")
            elif event.key == K_LEFT:
                # memorize the moving direction
                snake.direction = (-1, 0)
            elif event.key == K_RIGHT:
                snake.direction = (1, 0)
            elif event.key == K_DOWN:
                snake.direction = (0, 1)
            elif event.key == K_UP:
                snake.direction = (0, -1)
        # game management -----
        if is_running:
            # snake update -----
            snake.move() # move snake
            if snake.has_touched(food[:1]):
                # if snake found food
                del food[0] # then delete food item from list,
                snake.energy = min(200, snake.energy+20) # increase energy,
                snake.grow() # and grow
            if len(food) == 0:
                # all food is eaten?
                is_running = False # then game is over and won
                game_color = Color("khaki") # show it by the khaki color
            elif snake.has_touched(poison) or snake.has_touched(snake.body) \
                or snake.has_hit_wall(screen_width, screen_height) \
                or snake.energy <= 0:
                # any touch, wall, or starvation ?
                game_color = Color("magenta") # then game is over & lost
                is_running = False
            # display update -----
            screen.fill(game_color) # clear the display with given color
            for item in poison:
                item.draw() # draw the poison elements
            for item in food[:1]:
                item.draw() # draw first food item if possible
            snake.draw() # draw snake
        #-----
        pygame.display.update() # perform display update
        clock.tick(FPS) # wait for clock tick

# close pygame environment and exit the program -----
pygame.quit()
sys.exit()

```