

## ANEXO II - TUTORIAS Y BUENAS PRÁCTICAS - PYTHON

### TUTORIAL DEL EJERCICIO III - IDLE

Realicen un programa en el que a través de un laberinto se recorra una ciudad recogiendo la basura que contamina el lugar.

Deben utilizarse los comandos y herramientas trabajados anteriormente, tener por lo menos dos niveles de complejidad (dos laberintos diferentes) y objetos que representen obstáculos para lograr pasar de nivel.

Contenidos: tipos de variables; manejo de input output; manejo de gráficos e imágenes; sentencias de decisión; bucles; manejo de datos; módulos; manejo de archivos; uso de librerías.

#### Actividad 1

Existe una ciudad en donde los ciudadanos dejan restos de basura en diferentes partes contaminando el medio ambiente. Deben recorrer esta ciudad e ir recolectando o eliminando la “Basura en la Ciudad”.

- Presenten el escenario de la ciudad puede o no ser el mismo fondo para nivel 1 y nivel 2.
- Al iniciar el juego en nivel 1, aparecerá aleatoriamente la basura en diferentes sectores de la ciudad, recolectenla antes de que termine el tiempo.
- Si logran recolectar todos los objetos del nivel 1, pasarán al nivel 2 y otra tanda de objetos aparecerá, que deberán recolectar antes de que termine el tiempo.
- Nombren el juego.

Cada ronda durará 10 segundos y el jugador tendrá que utilizar al escobillón para barrer las bolsas de basura de la ciudad, lo cual le sumará puntos. Esto implica que hay que tener 3 imágenes como mínimo: una para la ciudad, una para el escobillón, y una para la basura.

Extras: Si el tiempo se los permite pueden agregar diferentes tipos de basura y sonido lo que dará dos puntos extras a ser considerado por los jueces.

Nivel 1: deben encontrar las bolsas de basura antes de que termine el tiempo



Basura que deben encontrar



Escobillón que manejan

Imagen final aproximada del juego terminado:



**Escriban estas líneas para referenciar librerías que utilizarán posteriormente.**

Inicialicen Pygame que los ayudará a graficar y manejar al mouse y Random, para poder hacer cosas aleatorias. Además carguen la ruta a las imágenes.

```
import os
import pygame
import random

pygame.init()

scriptDir = os.path.dirname(__file__)
impath = os.path.join(scriptDir, 'Images')
```

**Carguen las imágenes en memoria.**

```
player= pygame.image.load(os.path.join(impath, 'player.png'))
garbage= pygame.image.load(os.path.join(impath, 'garbage.png'))
background = pygame.image.load(os.path.join(impath, 'background.png'))
```

Cuentan con tres variables, cada una conteniendo la imagen lista para mostrarse.

**Preparen las cosas que necesitan para dibujar en la pantalla.**

Utilicen una variable que indique el color negro, otra para dejar un espacio sobre el mapa que permita mostrar los puntajes, y creen una ventana del tamaño correcto para mostrar el mapa.

```
BLACK = (0,0,0)
offset = (0,30)
size = background.get_size()
size=(size[0]+offset[0],size[1]+offset[1])
screen = pygame.display.set_mode(size)
screen.blit(background, (0,0))
```

Obtengan los rectángulos para cada basura y jugador.

Esto es lo que utilizarán luego para detectar si el escobillón está tocando la basura

```
playerRect=player.get_rect()
garbageRect=garbage.get_rect()
```

Cambien las tipografías e inicialicen unas variables más.

```
score=0
levelGarbage=[]
font = pygame.font.SysFont("monospace", 15)
timeLeft=0
levelCount=0
done = False
```

Realicen las funciones que dibujan la pantalla, revisan las colisiones y mueven el jugador a un punto determinado.

```
def movePlayerTo(coords):
    playerRect.center=coords

def checkCollisions():
    global score
    for g in levelGarbage:
        if playerRect.colliderect(g):
            score+=1
            levelGarbage.remove(g)

def draw():
    screen.fill(BLACK)
    screen.blit(background,offset)
    for g in levelGarbage:
        screen.blit(garbage, (g.left,g.top))
    screen.blit(player, ( playerRect.left,playerRect.top))
    label = font.render("Score:" + str(score), 1, (255, 255, 255))
    screen.blit(label, (0, 0))

    label2 = font.render("Fin del juego!", 1, (255, 255, 255))
    if levelCount>2:
        screen.blit(label2, (0+label.get_rect().width +10, 0))

    label3 = font.render("Tiempo Restante:" +str(timeLeft), 1, (255, 255, 255))
    screen.blit(label3, (0 +label.get_rect().width+ label2.get_rect().width + 20, 0))

pygame.display.flip()
```

Realicen las funciones que inician a un nivel, y que crean a la basura en posiciones aleatorias.

```
def createSingleGarbage():
    temp = garbageRect.copy()
    temp.top=random.randint(offset[0],size[1])
    temp.left =random.randint(offset[1],size[0])
    return temp

def createLevel(amount):
    global timeLeft
    timeLeft=10
    levelGarbage.clear()
    for x in range(amount):
        levelGarbage.append(createSingleGarbage())
```

Realicen la función que avanza al nivel y la que va decrementando el contador

```
def advanceLevel():
    global levelCount
    global timeLeft
    if len(levelGarbage)==0:
        levelCount+=1
        timeLeft=0
        if(levelCount<3):
            createLevel(7 +3*levelCount)
pygame.time.set_timer(pygame.USEREVENT, 1000)
def timeDown():
    global timeLeft
    global levelCount
    if(timeLeft<=0):
        levelCount=3
        return True
    timeLeft-=1
```

Armen la función del bucle del juego. Todos los juegos la tienen. Esta función moverá al jugador, revisará las colisiones contra la basura, y a lo último llama a la que grafica los elementos.

```
while not done:
    # --- Main event loop
    for event in pygame.event.get():
        if event.type == pygame.USEREVENT:
            timeDown()
        if event.type == pygame.QUIT:
            done=True
            break;
    if levelCount<3:
        advanceLevel()
        movePlayerTo(pygame.mouse.get_pos())
        checkCollisions()
    draw()

pygame.quit()
```

**Actividad 2**

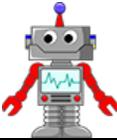
Deben limpiar las calles de una ciudad con un “Robot Reciclador de Basura” dada la una invasión de desperdicios; aparecerán bolsas de basura que nuestro robot deberá eliminar mientras esquiva a los autos mediante saltos.

- a) Presenten el escenario de la ciudad. Puede o no ser el mismo fondo para el nivel 1 y nivel 2.
- b) Al iniciar el juego en nivel 1, aparecerán autos acercándose y bolsas de basura en forma aleatoria.
- c) El robot debe moverse con las flechas izquierda y derecha del teclado. Barra espaciadora para saltar.
- d) El robot debe tocar la bolsa de residuos para sumar 10 puntos. Un sonido confirmará dicha acción.
- e) Si logran 40 puntos, se pasa al nivel 2 y los autos irán más rápido.
- f) En el caso que el robot colisione contra un auto, el juego se termina y la puntuación vuelve a 0.
- g) Nombren el juego.

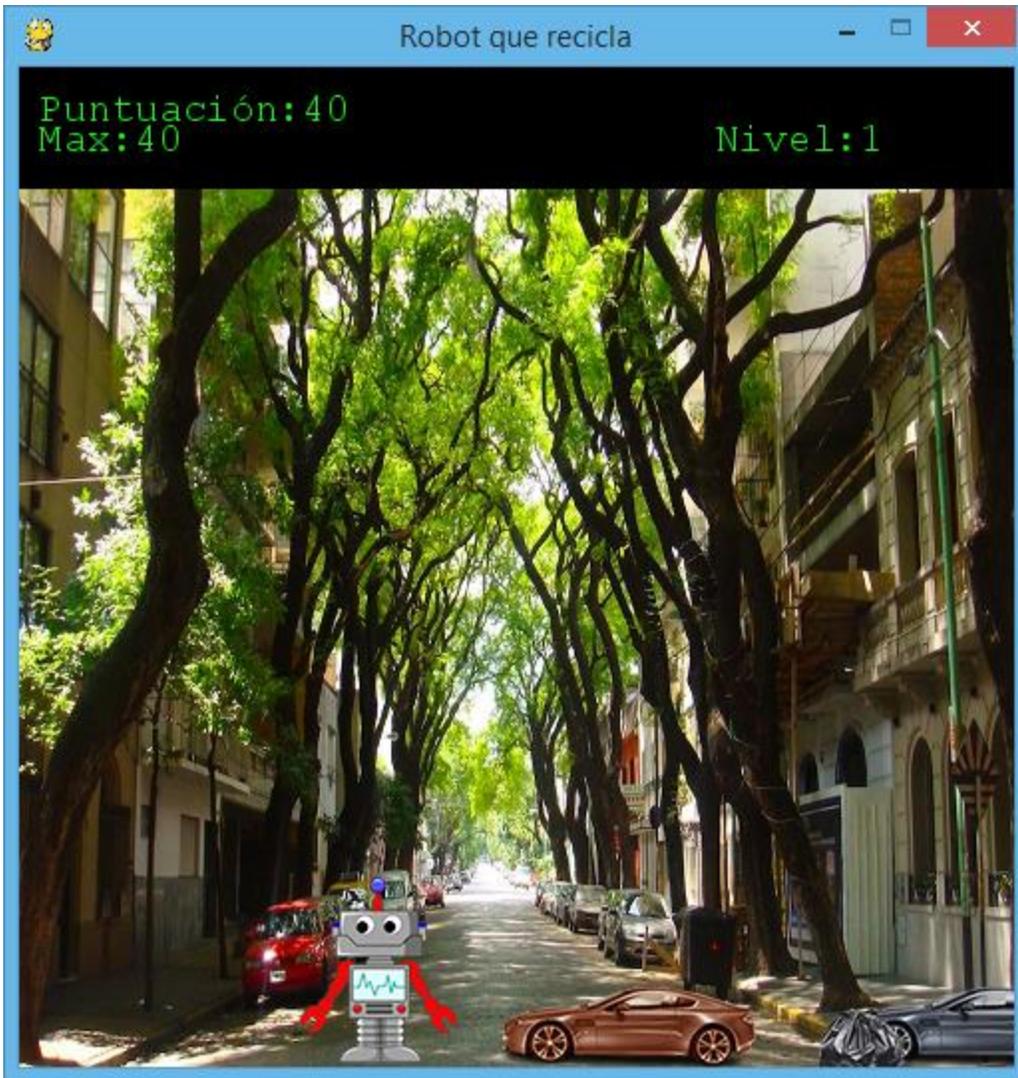
Si hubo una puntuación máxima mayor a la histórica se registrará para las partidas posteriores.

Nivel 1 y 2: deben recolectar las bolsas de basura sin chocar los autos. Cuando se aproxime un auto deben saltar.

Hay que tener 5 imágenes como mínimo: un robot, 2 autos, un fondo y una bolsa de basura y por último 2 sonidos: bolsa recolectada y salto del robot.

Bolsa de residuos que deben eliminar	
Robot reciclador de basura	
Autos a esquivar	
	

A continuación les mostramos una imagen final aproximada del juego terminado:



Realicen un código de inicialización.

```
import pygame
import os
import random

pygame.init()
pygame.mixer.init()
win = pygame.display.set_mode((500, 500))
pygame.display.set_caption("Robot que recicla")
font = pygame.font.SysFont("monospace", 20)

scriptDir = os.path.dirname(__file__)
impath = os.path.join(scriptDir, 'Imágenes')
sonidoOK= pygame.mixer.Sound("ok.ogg")
sonidoJump= pygame.mixer.Sound("jump.ogg")
robot = pygame.image.load(os.path.join(impath, 'robot.png'))
auto = pygame.image.load(os.path.join(impath, 'auto2.png'))
auto2 = pygame.image.load(os.path.join(impath, 'auto.png'))
fondo = pygame.image.load(os.path.join(impath, 'fondo.png'))
basura = pygame.image.load(os.path.join(impath, 'basura.png'))

run = True
isJump = False
x = 50
y = 440
width = 40
height = 60
vel = 7
puntos=0
max=0
jumpCount = 10
run = True
isJump = False
x = 50
y = 440
width = 40
height = 60
vel = 7
puntos=0
max=0
jumpCount = 10

autoRect = auto.get_rect()
autoRect.left = 600
autoRect.top = y + 20
auto2Rect = auto.get_rect()
auto2Rect.left = 900
auto2Rect.top = y + 20
basuraRect = basura.get_rect()
basuraRect.left = 350
basuraRect.top = y + 30

label3 = font.render("Nivel:" + str(1), 1, (20, 255, 20))
```



Realicen la función de dibujado de los elementos del juego.

```
def reDraw():
    win.fill((0, 0, 0))
    win.blit(fondo, (0, 0))
    win.blit(auto, (autoRect.left, autoRect.top))
    win.blit(auto2, (auto2Rect.left, auto2Rect.top))
    win.blit(basura, (basuraRect.left, basuraRect.top))
    win.blit(robot, (x, y - height + (height - width) + 5))

    label = font.render("Puntuación:" + str(puntos), 1, (20, 255, 20))
    label2 = font.render("Max:" + str(max), 1, (20, 255, 20))

    win.blit(label, (10, 10))
    win.blit(label2, (10, 25))
    win.blit(label3, (350, 25))

    pygame.display.update()
```

Realicen el bucle del juego (todos los juegos tienen uno).

```
while run:
    playerRect = pygame.Rect(x, y, width, height)

    if (autoRect.left < -200):
        autoRect.left = 600

    if (auto2Rect.left < -200):
        auto2Rect.left = 600

    if playerRect.colliderect(basuraRect):
        basuraRect.left = random.randint(50, 450)

        if puntos == max:
            max += 10

        puntos += 10

        sonidoOK.play()

    if playerRect.colliderect(autoRect) or playerRect.colliderect(auto2Rect):
        puntos = 0
```



```
if puntos<=40:
    autoRect.left = autoRect.left - 2
    auto2Rect.left = auto2Rect.left - 5
    label3 = font.render("Nivel:" + str(1), 1, (20, 255, 20))
else:
    autoRect.left = autoRect.left - 10
    auto2Rect.left = auto2Rect.left - 8
    label3 = font.render("Nivel:" + str(2), 1, (20, 255, 20))

pygame.time.delay(25)

for event in pygame.event.get():
    if event.type == pygame.QUIT:
        run = False

keys = pygame.key.get_pressed()

if keys[pygame.K_LEFT] and x > vel:
    x -= vel
if keys[pygame.K_RIGHT] and x < 500 - (width + robot.get_size()[0]/2):
    x += vel

if not(isJump):
    if keys[pygame.K_SPACE]:
        sonidoJump.play()
        isJump = True
else:
    if jumpCount >= -10:
        neg = 1
        if jumpCount < 0:
            neg = -1
        y -= (jumpCount ** 2) * 0.7 * neg
        jumpCount -= 1
    else:
        isJump = False
        jumpCount = 10

reDraw()

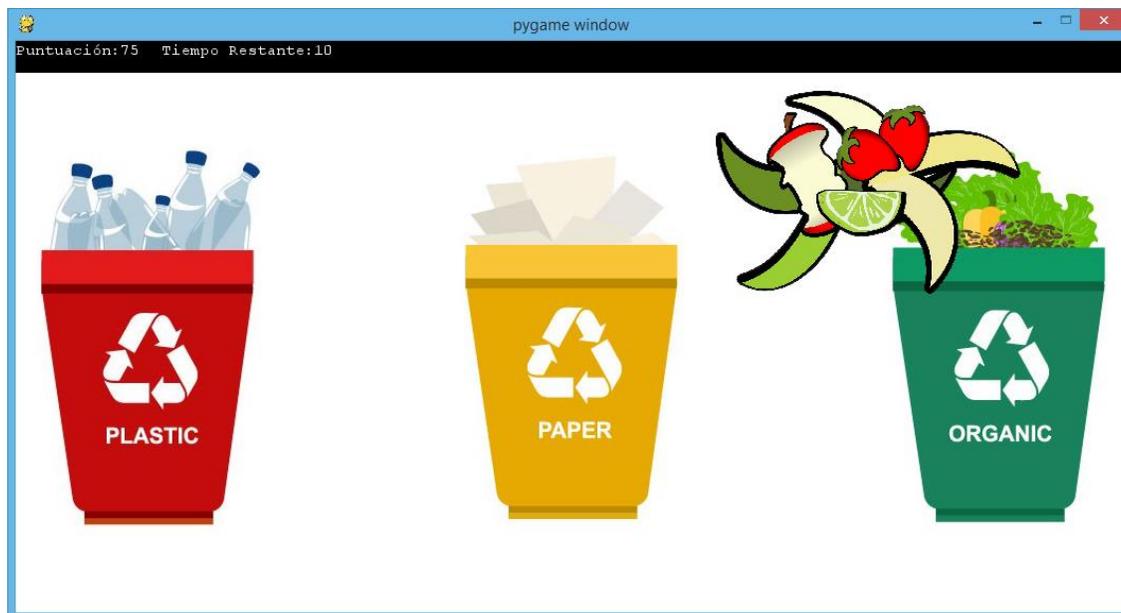
quit()
```



### Actividad 3

Este juego permite ayudar a que las personas aprendan más fácil en qué cestos van los distintos tipos de basura “Reciclando la basura en la Ciudad”.

Imagen final aproximada del juego terminado:



Cada ronda durará 10 segundos y el jugador tendrá que depositar la basura en el centro que le corresponde para sumar puntos. Esto implica que hay que tener 6 imágenes como mínimo: 3 para cada uno de los tachos y 3 para cada uno de los tipos de basura. Por último un sonido para cuando la basura entre y una imagen de fondo.

**Escriban estas líneas para referenciar librerías que utilizarán posteriormente.**

Inicialicen Pygame junto con su mixer para poder reproducir sonidos. Además carguen la ruta a las imágenes y el sonido (en este caso “true.ogg”) que ocurrirá al ingreso de basura.

```
import os
import pygame
import random

pygame.init()
pygame.mixer.init()

scriptDir = os.path.dirname(__file__)
impath = os.path.join(scriptDir, 'Images')
sonidoTrue= pygame.mixer.Sound("true.ogg")
```

**Carguen las imágenes en memoria.**

```
background = pygame.image.load(os.path.join(impath, 'background.png'))
tachopapel = pygame.image.load(os.path.join(impath, 'tachopapel.jpg'))
tachoplastico = pygame.image.load(os.path.join(impath, 'tachoplastico.jpg'))
tachorganico = pygame.image.load(os.path.join(impath, 'tachorganico.jpg'))

basuraorganica = pygame.image.load(os.path.join(impath, 'basuraorganica.png'))
basurapapel = pygame.image.load(os.path.join(impath, 'basurapapel.png'))
basuraplastico = pygame.image.load(os.path.join(impath, 'basuraplastico.png'))
```

La primer basura a descartar tiene que ser de plástico y deben dibujar la barra negra de arriba.

```
imagenActual= basuraplastico
player=imagenActual
numeroImgActual=1

BLACK = (0,0,0)
offset = (0,30)
size = background.get_size()
size=(size[0]+offset[0],size[1]+offset[1])
screen = pygame.display.set_mode(size)
screen.blit(background, (0,0))
```

Obtengan los rectángulos para cada tacho y el jugador.

```
tachoplasticoRect=tachoplastico.get_rect()
tachoplasticoRect.left=20
tachoplasticoRect.top=100

tachopapelRect=tachopapel.get_rect()
tachopapelRect.left=420
tachopapelRect.top=100

tachorganicoRect=tachorganico.get_rect()
tachorganicoRect.left=820
tachorganicoRect.top=100

playerRect=player.get_rect()
```

Cambien las tipografías e inicialicen unas variables más.

```
font = pygame.font.SysFont("monospace", 15)

puntuacion=0
done = False
tiempoRestante=20
estadoJuego='Jugando'
```

Realicen la función que dibuja la pantalla y la que mueve el jugador a un punto determinado.

```
def movePlayerTo(coords):
    playerRect.center=coords

def draw():

    screen.fill(BLACK)
    screen.blit(background,offset)
    screen.blit(tachoplastico, (tachoplasticoRect.left, tachoplasticoRect.top))
    screen.blit(tachorganico, (tachorganicoRect.left, tachorganicoRect.top))
    screen.blit(tachopapel, (tachopapelRect.left, tachopapelRect.top))
    screen.blit(player, (playerRect.left,playerRect.top))

    label = font.render("Puntuación:" + str(puntuacion), 1, (255, 255, 255))
    screen.blit(label, (0, 0))
    label2= font.render("", 1, (255, 255, 255))
    label3 = font.render("Tiempo Restante:" +str(tiempoRestante), 1, (255, 255, 255))
    screen.blit(label3, (0 +label.get_rect().width+ label2.get_rect().width + 20, 0))
    pygame.display.flip()
```

Realicen la función que detiene el juego y la que cambia la imagen de la basura por una al azar.

```
def stopGame():

    global estadoJuego
    estadoJuego= 'Parado'

def cambiarImagen():

    global numeroImgActual
    global player

    numeroImgActual = random.randint(1, 3)

    if numeroImgActual==1:
        player = basuraplastico
    elif numeroImgActual==2:
        player = basurapapel
    elif numeroImgActual==3:
        player = basuraorganica
```

Realicen la función que comienza el juego y la que va decrementando el contador

```
def startGame():

    global estadoJuego
    global puntuacion
    global tiempoRestante

    estadoJuego = 'Jugando'
    puntuacion=0
    tiempoRestante = 20
    pygame.time.set_timer(pygame.USEREVENT, 1000)

def timeDown():

    global tiempoRestante

    if(tiempoRestante<=0):
        stopGame()
        return True
    tiempoRestante -= 1
```



Realicen la función del bucle del juego. Todos los juegos la tienen. Esta función a lo último llama a la que grafica los elementos.

```

while not done:
    for event in pygame.event.get():
        if event.type == pygame.MOUSEBUTTONDOWN:
            if estadoJuego == 'Parado':
                startGame()
            else:
                if playerRect.colliderect(tachopapelRect):
                    if numeroImgActual == 2:
                        puntuacion += 25
                        sonidoTrue.play()
                        cambiarImagen()
                if playerRect.colliderect(tachoplasticoRect):
                    if numeroImgActual == 1:
                        puntuacion += 100
                        sonidoTrue.play()
                        cambiarImagen()
                if playerRect.colliderect(tachorganicoRect):
                    if numeroImgActual == 3:
                        sonidoTrue.play()
                        puntuacion += 50
                        cambiarImagen()
        if event.type == pygame.USEREVENT:
            timeDown()
        if event.type == pygame.QUIT:
            done = True
            break;

    movePlayerTo(pygame.mouse.get_pos())
    draw()
pygame.quit()

```