UNIVERSIDAD NACIONAL GENERAL SARMIENTO

Introducción a la programación

TRABAJO PRÁCTICO

PyGame - TutiFrutiUNGS

Comisión 07

Turno Noche

Grupo 4

Peralta Laura
Bertani Luana
Lopez Gonzalo

Profesores: ***Cinthya Anabela Cardozo, Luis Santiago Veronesi***

07 de Junio del 2020
Peralta Laura, Bertani Luana, Lopez Gonzalo,
Universidad Nacional General Sarmiento
Introducción a la programación
Comisión 07
Profesores: Cinthya Cardozo, Luis Veronesi

Trabajo Práctico PyGame

***Consigna**:*

*Implementar las funciones requeridas para el correcto funcionamiento del juego. **esCorrecta(), unaAlAzar(), juegaCompu()***

Pensar e implementar funciones auxiliares que resuelvan tareas intermedias, de forma tal que el código sea más claro, sencillo, ordenado, legible y fácil de corregir. Las funciones que reciben listas como parámetros deberán también chequear que dichas listas permanezcan en el estado correcto luego de utilizada la función.

Informe: 07-junio-2020

FUNCIONES AGREGADAS

def unaAlAzar(abc)

```
def unaAlAzar(abc):
    letraElegida = random.choice(abc)
    return letraElegida
```

Se encarga de retornar una letra al azar para que se use en el juego.

def esCorrecta(palabraUsuario, letra, item, items, listaDeTodo):.

```
def esCorrecta(palabraUsuario, letra, item, items, listaDeTodo):
    palabraUsuario = palabraUsuario.lower()
    indiceDelItem = items.index(item)
    opcionesDelItem = listaDeTodo[indiceDelItem]
    if (palabraUsuario[0] == letra and palabraUsuario in opcionesDelItem):
        return 10
    return -5
```

Esta función verifica si la opción del usuario empieza con la letra indica y también si se encuentra dentro de las disponibles en la lista de todas las opciones según el ítem correspondiente.

def juegaCompu(letraAzar, listaDeTodo):

```
def juegaCompu(letraAzar, listaDeTodo):
    listaLetra = []
    resultado = []

    for idx in range(len(listaDeTodo)):
        resultado.append("...")

        for palabra in listaDeTodo[idx]:
            if palabra[0] == letraAzar:
                listaLetra.append(palabra)

        if len(listaLetra)>0:
            resultado.pop(-1)
            resultado.append(random.choice(listaLetra))
            listaLetra=[]
    return resultado
```

juegaCompu devuelve una lista con una opción, por cada ítem, que cumpla el criterio de la letra elegida y se encuentren dentro del ítem correspondiente, en caso de no hallar ninguna opción devuelve como respuesta "..."

def guardar_puntajes(puntajes) y def recuperar_puntajes()

```
def guardar_puntajes(puntajes):
    historial = open("datos/historial.txt", "w")
    for puntaje, tiempo in puntajes:
        historial.write(str(puntaje)+","+tiempo+"\n")
    historial.close()


def recuperar_puntajes():
    puntajes = []
    historial = open("datos/historial.txt", "r")

    for linea in historial:
        puntaje, tiempo = linea.rstrip("\n").split(",")
        puntajes.append((int(puntaje),tiempo))
    historial.close()

    return puntajes
```

Estas funciones son las encargadas de mantener un historial con el mejor puntaje, y su tiempo, que se hayan logrado hasta el momento.

def cargarItems()

```
def cargarItems():
    nombres=[]
    animales=[]
    colores=[]
    sustantivos_comunes=[]
    paises=[]
    marcas=[]
    cap_prov_arg=[]

    #Nombres
    nombres_txt=codecs.open("items/nombres.txt","r","utf-8")
    datos_nombres=nombres_txt.read()
    nombres.append(datos_nombres)
    nombres_txt.close()
    nombres=nombres[0].split(",")

    '''
    Demas items
    '''

    listaDeTodo=[nombres,colores,sustantivos_comunes,paises,marcas,cap_prov_arg]

    return listaDeTodo
```

Con la presente función se leen los archivos de textos que contienen las opciones válidas para cada ítem y se los carga en formato de lista dentro de la variable listaDeTodo la cual será usada durante el juego para chequear si las opciones son correctas o no.

def dibujarPresentacion(screen, imgPresentacion, segundos)

```
def dibujarPresentacion(screen, imgPresentacion, segundos):
    screen.blit(imgPresentacion, (0, 0))

    segundos *= -1

    defaultFontMUYGRANDE = pygame.font.Font(pygame.font.get_default_font(), TAMANO_LETRA_MUYGRANDE)

    RenCtaRegresiva = defaultFontMUYGRANDE.render(str(segundos), 1, COLOR_BLANCO)

    screen.blit(RenCtaRegresiva, (396, 115))
```

La presentación del juego es lograda por esta función, la cual recibe por parámetro la ventana del juego, una imagen y los segundos que son usados en la cuenta regresiva inicial.

MODIFICACIONES

```
while True:
    if juegoNuevo:
        # Controladores de ciclo
        juegoNuevo = False
        presentacion = True
        habilitarReinicio = False
        ctaRegresiva = 4
        i = 0

        # Tiempo total del juego
        gameClock = pygame.time.Clock()
        totaltime = 0
        fps = FPS_INICIAL

        # Variables
        puntos = 0
        palabraUsuario=""
        eleccionUsuario=[]
        eleccionCompu=[]
        aciertos = 0
        incorrectas = 0
        segundos = 0
        letraAzar = unaAlAzar(abc)

        # Musica
        pygame.mixer.music.load("sonidos/intro.mp3")
        pygame.mixer.music.play()
        aycaramba=pygame.mixer.Sound("sonidos/aycaramba.wav")
```

*En el archivo **principal.py** se agregó un nuevo ciclo, por encima del que ya estaba, para habilitar la posibilidad de reiniciar el juego, en este ciclo también se encuentran definidas las variables que se usan y se resetean en caso de que se inicie un juego nuevo.*

```
#letra = dameLetraApretada(e.key)
letra = e.unicode
palabraUsuario += letra
```

Se comentó la funcion dameLetraApretada y se habilitó el uso de tildes y la letra ñ

```
#segundos = pygame.time.get_ticks() / 1000
segundos = (math.ceil((totaltime / 100)/10) - ctaRegresiva)
```

Se rehizo la forma de contar los segundos para permitir el reinicio por cada juego.

*Por otro lado en la función **dibujarSalida()** del archivo **extras.py** se agregaron varías lineas de código para renderizar los resultados del juego, como también para verificar si se logró un nuevo récord y en tal caso llamar a la función **guardar_Puntajes().***

```
'''
    EXTRAS AGREGADOS
'''

# Resultados
ptsCoincidencia = 0
for idx in range(0, (len(eleccioncompu))):

total = puntos + ptsCoincidencia - segundos

renAcierto = defaultFont.render      ("Aciertos:             " + str(aciertos) + "pts", 1, COLOR_TEXTO)
renCoincidenecia = defaultFont.render("Coincidencias:    " + str(ptsCoincidencia) + "pts", 1, COLOR_TEXTO)
renIncorrectas = defaultFont.render  ("Incorrectas:        " + str(incorrectas) + "pts", 1, COLOR_TEXTO)
renDescTiempo = defaultFont.render   ("Tiempo:              - " + str(segundos) + "pts", 1, COLOR_TEXTO)
renTotal = defaultFont.render        ("TOTAL:        " + str(total) + "pts", 1, COLOR_TEXTO)

screen.blit(renAcierto, (100, 300))
screen.blit(renCoincidenecia, (100, 330))
screen.blit(renIncorrectas, (100, 360))
screen.blit(renDescTiempo, (100, 390))
screen.blit(renTotal, (150, 430))

# Record
ultimo_record = recuperar_puntajes()
record = ultimo_record[0][0]
tiempo = ultimo_record[0][1]

# Renderizar nuevo record
if total > record:
    #  Musica ganador
    pygame.mixer.music.load("sonidos/ta-ra-ra-ra-hey.mp3")
    pygame.mixer.music.play()

    renFelicidades = defaultFont.render("FELICIDADES NUEVO RECORD", 1, COLOR_LETRAS)
    renNuevoRecord = defaultFont.render("El nuevo record es de : " + str(total) + "pts", 1, COLOR_LETRAS)

    screen.blit(renFelicidades, (400, 350))
    screen.blit(renNuevoRecord, (400, 400))

    #Guardar nuevo record
    puntajes = [(total, str(int(segundos)))]
    guardar_puntajes(puntajes)

# Renderizar record anterior
else:
    #  Musica perdedor
    pygame.mixer.music.load("sonidos/ouch..mp3")
    pygame.mixer.music.play()


    renRecord= defaultFont.render("Tu record anterior fué de " + str(record)+ "pts, en " + tiempo + " segundos", 1, COLOR_LETRAS)
    screen.blit(renRecord, (100, 470))

# Jugar de nuevo
renReiniciar = defaultFont.render("PRESIONE ENTER PARA JUGAR DE NUEVO", 1, COLOR_LETRA)
screen.blit(renReiniciar, (100, 550))
```