TP INTRO. PROGRAMACION

“palabras encolumnadas”

INTEGRANTES:

- Cristian Pereyra Lopez
- Francisco Barrientos
- Tobias Schenfeld
- Agustin Ortiz

PROFESORES:

- Luis Veronesi
- Flavia Bottino
- Nora Martinez

EL JUEGO:

Nuestro juego es un juego para chicos con la temática del Rey León. El objetivo es que los niños armen nombres de animales utilizando las letras que van cayendo en la pantalla, dando cierto puntaje por animales según las letras que componen su nombre y otro mayor especial adicional si la palabra corresponde al nombre de un personaje de la película (por ejemplo, Simba, Scar, Mufasa, etc). Las letras elegidas tienen que respetar un orden de izquierda a derecha o no serán consideradas válidas. También el juego reproduce sonidos de animales al azar en los aciertos comunes, mientras que si es un acierto especial reproduce “Hakuna Matata”.

EJEMPLO:

FUNCIONES:

```
def cargarListas(lista, listaIzq, listaMedio, listaDer, posicionesIzq , posicionesMedio, posicionesDer):
```

Se elige una palabra de la lista, carga las letras en las 3 listas y les inventa una posición para que aparezcan arriba en la columna correspondiente.

```
## elijo una palabra al azar del lemario
    palabraAzar = lista[random.randint(0, len(lista)-1)]
## la divido en tres para repartir las letras
    palabra = palabraAzar
    palabraEn3 = len(palabra) // 3
    cont = 0

    for letra in palabra:
            if cont < palabraEn3:
                listaIzq.append(letra)
                posicionesIzq.append([random.randrange(0, ANCHO // 3 - 30), 25])

                cont += 1
            elif cont < palabraEn3 * 2:
                listaMedio.append(letra)
                posicionesMedio.append([random.randrange(ANCHO // 3 + 30, (ANCHO // 3) * 2) - 30, 25])

                cont += 1
            else:
                listaDer.append(letra)
                posicionesDer.append([random.randrange((ANCHO // 3) * 2 + 30, ANCHO - 30), 25])
```

-Al principio teníamos pensado en vez de dividir la palabra, hacer un “random” con un rango de 1 a 3 para distribuir la palabra en las columnas, pero estaba la posibilidad de que entre la palabra entera en una columna.

```
def bajar(listaLetras,posiciones):
```

Hace bajar las letras y elimina las que tocan el piso.

```
ALTO = 600
for i in range(len(posiciones)-1,-1,-1):
    posiciones[i][1] = posiciones[i][1] + 40
    if posiciones[i][1]>ALTO-100:
        listaLetras.pop(i)
        posiciones.pop(i)
```

```
def actualizar(lista, listaIzq, listaMedio, listaDer, posicionesIzq , posicionesMedio, posicionesDer):
```

Llama a otras funciones para bajar las letras, eliminar las que tocan el piso y cargar nuevas letras a la pantalla.

```
cargarListas(lista, listaIzq, listaMedio, listaDer, posicionesIzq , posicionesMedio, posicionesDer)
bajar(listaIzq,posicionesIzq)
bajar(listaMedio,posicionesMedio)
bajar(listaDer,posicionesDer)
```

```
def puntos(candidata):
```

Recibe una palabra y retorna el puntaje total correspondiente a la palabra formada.

```
#devuelve el puntaje que le corresponde a candidata
puntaje = 0
for e in candidata.lower():
    if e < "a" or e > "z" :
        puntaje = 0
        return puntaje

for e in candidata.lower():
    if e in "aeiou":
        puntaje = puntaje + 1
    else:
        if e in "jkqwxyz":
            puntaje = puntaje + 5
        else:
            puntaje = puntaje + 2

return puntaje
```

```
def procesar(lista, candidata, listaIzq, listaMedio, listaDerecha, listaPJ):
```

Chequea que candidata sea correcta en cuyo caso devuelve el puntaje y 0 si no es correcta.

```
if(esValida(lista, candidata, listaIzq, listaMedio, listaDerecha)):
    if esPersonaje(candidata, listaPJ):
        si es personaje da un adicional de puntos
        return (puntos(candidata) + 10)
    else:
        sonidos(candidata)
        return (puntos(candidata))
else:
    return 0
```

```
def esValida(lista, candidata, listaIzq, listaMedio, listaDerecha):
```

Chequea si la palabra cumple con los requisitos.

```
if candidata not in lista:
    return False
else:
    esta = 0
    for i in range(len(candidata)):
        if esta==0:
            if candidata[i] in listaIzq:
                esta=esta + 1
        if esta==1:
            if candidata[i] in listaMedio:
                esta=esta + 1
        if esta==2:
            if candidata[i] in listaDerecha:
                return True
return False
```

FUNCIONES EXTRAS:

```
def esPersonaje(candidata, listaPJ):
```

Chequea si la palabra es un personaje de la película “Rey León”.

```
if candidata in listaPJ:
  sonidoAnimal = pygame.mixer.Sound("sonidos/Hakuna.mp3")
  sonidoAnimal.play()
  return True
```

```
def sonidos(candidata):
```

Si la palabra no es un personaje, reproduce un sonido de animal al azar.

```
if sonidoAnimal == 1:
        sonidoAnimal = pygame.mixer.Sound("sonidos/Hiena.mp3")
        sonidoAnimal.play()
elif sonidoAnimal == 2:
        sonidoAnimal = pygame.mixer.Sound("sonidos/Cocodrilo.mp3")
        sonidoAnimal.play()
elif sonidoAnimal == 3:
        sonidoAnimal = pygame.mixer.Sound("sonidos/Elefante.mp3")
        sonidoAnimal.play()
elif sonidoAnimal == 4:
        sonidoAnimal = pygame.mixer.Sound("sonidos/Jabali.mp3")
        sonidoAnimal.play()
elif sonidoAnimal == 5:
        sonidoAnimal = pygame.mixer.Sound("sonidos/Leon.mp3")
        sonidoAnimal.play()
elif sonidoAnimal == 6:
        sonidoAnimal = pygame.mixer.Sound("sonidos/Rinoceronte.mp3")
        sonidoAnimal.play()
elif sonidoAnimal == 7:
        sonidoAnimal = pygame.mixer.Sound("sonidos/Tigre.mp3")
        sonidoAnimal.play()
else:
        sonidoAnimal = pygame.mixer.Sound("sonidos/Zebra.mp3")
        sonidoAnimal.play()
```

```
def sinAcentos(lista):
```

Busca si la palabra tiene tildes.

```
for palabra in lista:
    for letra in palabra:
        if letra == "á":
            letra = "a"
        if letra == "é":
            letra = "e"
        if letra == "í":
            letra = "i"
        if letra == "ó":
            letra = "o"
        if letra == "ú":
            letra = "u"
```

```
def dibujarNombreJugador(screen, nombre):
```

Muestra el nombre del jugador en la pantalla inicial.

```
letras = pygame.font.SysFont("Arial_Black", 60)
config = letras.render(nombre, 1, (255, 126, 0))
screen.blit(config, (200, 350))
```

PROBLEMAS Y DIFICULTADES:

Estuvimos haciendo muchos arreglos a nuestro juego, como hacer fondos especiales, títulos de las ventanas, icono del juego, música en cada ventana, cambiamos el color de las letras de las columnas, etc. Finalmente guardamos los puntajes de los jugadores y sus nombres en un archivo externo para recurrir a él y mostrar qué jugador tuvo el mejor puntaje y el récord más alto a romper:

```
abre un .txt para guardar los jugadores
archivo = open("jugadores.txt","a", encoding='ISO8859')
jugador = nombre
archivo.writelines(jugador + "\n")
archivo.close()

guarda los nombres de los jugadores en una lista
archivo= open("jugadores.txt","r", encoding='ISO8859')
for linea in archivo.readlines():
    listaJugadores.append(linea[0:-1])

abre un .txt para guardas los puntajes
archivo = open("records.txt","a", encoding='ISO8859')
archivo.writelines(str(puntos) + "\n")
archivo.close()

guarda los puntajes en una lista
archivo= open("records.txt","r", encoding='ISO8859')
for linea in archivo.readlines():
    listaRecords.append(linea[0:-1])

for i in range(len(listaRecords)):
    listaRecords[i] = int(listaRecords[i])

busco el mayor puntaje, luego lo vuelvo a pasar a str para poder mostrarlo en pantalla
recordMax = max(listaRecords)
  recordMax = str(recordMax) + "  PUNTOS"
busca el jugador con mejor puntaje y lo muestra en pantalla
for i in range(len(listaRecords)):
    if listaRecords [i] == recordMax:
        jugadorRecord = "El record del juego es de  " + listaJugadores[i] + "  con  " + str(recordMax) + "  puntos"
        escribirEnPantalla(screen, jugadorRecord , (120, 500), 40, (191,122,54))
```

Algunos problemas a los cuales nos enfrentamos durante la creación del juego fueron:

- El tiempo máximo del juego disminuía en la pantalla inicial, a pesar de que esta no mostraba dicho tiempo. Para resolverlo, empleamos la siguiente variable que tomaba el tiempo empleado en la pantalla inicial y luego lo sumaba al tiempo máximo para el juego principal.

```
milisegundos = []

milisegundos.append(pygame.time.get_ticks())
segundos = TIEMPO_MAX - pygame.get_ticks()/1000 + milisegundos [0]/1000
```

- Las letras caían muy rápido en el juego principal, así que le añadimos un retraso al final de la función “actualizar” con la siguiente sintaxis:

```
pygame.time.delay(1000)
```

- Al estar en mayúsculas, las letras que caían no coincidían con las que se ingresaban para formar “candidata”. Entonces, decidimos modificar en la función “dameLetraApretada” las letras asignadas a cada botón para resolver este problema.

```
def dameLetraApretada(key):
    if key == K_a:
        return("A")
    elif key == K_b:
        return("B")
    elif key == K_c:
        return("C")
```

...

- Intentamos resolver la función “estaCerca” pero no supimos como implementarla:

```
def estaCerca(elem, lista):
##    if len(lista)==0:
##            return False
##    else:
##        for pos in range (len(lista)):
##            if elem - lista[pos] < 15 and elem - lista[pos] > -15:
##                    return True
##            else:
##                cambiar = False
##
##            if not cambiar:
##                return False
##        return True
```

CONCLUSIÓN:

Finalmente, sobre el desarrollo, podemos decir que al tener los cuatro solamente los conocimientos de esta cursada, nos costó bastante y obviamente hay detalles que quizás faltan y cosas a mejorar, pero nos divertimos mucho y aprendimos bastante investigando, mirando tutoriales y haciendo mil cosas a prueba y error. Esperamos que les gusten nuestras propuestas, la idea de este trabajo práctico de intentar plasmar de forma práctica nos motivó y planteó desafíos muy divertidos.