Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

Informe de Trabajo Final

INTRODUCCIÓN

Para el trabajo final, se ha completado el desarrollo de un software utilitario de videojuego con código ya existente, escrito en python y basado en la librería PyGame.

El juego consiste en adivinar la mayor cantidad de nombres de países en cierto tiempo. Al ingresar correctamente el nombre del país, el software premia con puntos y agrega más tiempo al restante y ofrece una nueva adivinanza. Si el nombre es incorrecto, tarda mucho en contestar o decide pasar de país se penaliza perdiendo 5 segundos de tiempo.

El juego termina cuando se acaba el tiempo. Si el jugador es uno de los mejores 10, se le permite guardar su nombre y puntaje para la posteridad.

Cada vez que el juego desafía con un nuevo país, muestra la capital del país como referencia y en el área central de la interfaz gráfica del programa, expone de forma desordenada y aleatoria, las letras que forman dicho país.

DESARROLLO

Se completaron las funciones vacías

```
def lectura(listaNombres,listaAyuda):#Cargar las dos listas desde los archivos

    archivoPalabras = open("paises.txt","r")
    archivoAyuda = open("capitales.txt","r")

    lineasPalabras = archivoPalabras.readlines()
    lineasAyuda = archivoAyuda.readlines()

    for linea in lineasPalabras:
        linea = linea.replace("\n","")
        listaNombres.append(linea)
    archivoPalabras.close()

    for linea in lineasAyuda:
        linea = linea.replace("\n","")
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
        listaAyuda.append(linea)
    archivoAyuda.close()
```

Abre los archivos con el diccionario de ayudas y adivinanzas, lee todas sus lineas y las recorre una a una con un bucle for para agregar cada elemento a la lista que corresponde.

```
def cargarListas(posX, posY, letrasEnPantalla, ocupados, palabra, ayuda, listaNombres, listaAyuda):
#Vaciar posX, posY, letrasenpantalla y ocupados, luego llamar cargarLetras y cargarPosiciones

    vaciarLista(posX)
    vaciarLista(letrasEnPantalla)
    vaciarLista(ocupados)
    vaciarLista(posY)
    cargarLetras(palabra,letrasEnPantalla)
    cargarPosiciones(letrasEnPantalla,posX,posY,ocupados)
```

Vacía las listas que solicita y ejecuta las funciones requeridas, se creó una nueva función para vacias una lista. Se explica más adelante

```
def cargarLetras(palabra, letrasEnPantalla):#Recorrer  palabra y apendear a letrasEnPantalla
   for letra in palabra:
       letrasEnPantalla.append(letra)
```

```
def cambiarPalabra(listaPalabra,posicionesOcupadas):#Devolver palabra elegida al azar
    contador = 0
    candidatoOcupar = random.randint(0,len(listaPalabra)-1)
    while (fueUsadaLaPalabra(candidatoOcupar,posicionesOcupadas) and contador < len(listaPalabra)):
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
#Si la palabra fue utilizada, seguirá probando un random hasta encontrar una que no haya sido utilizada
        candidatoOcupar = random.randint(0,len(listaPalabra)-1)
        contador = contador + 1
    if (contador >= len(listaPalabra)): # El que no se repitan las palabras, puede provocar que en algún momento nuestro diccionario se acabe.
Entonces si esto ocurre que el juego deje de ejecutarse
        print("No hay más palabras para adivinar")
        pygame.quit()
    posicionesOcupadas.append(candidatoOcupar) # Añade el registro de la posición ocupada
    return listaPalabra[candidatoOcupar]
```

Se agregó un parámetro a la función que posibilita cumplir con el requisito de que no se repitan las mismas palabras para adivinar. Es así que la lista posicionesOcupadas cumple el rol de almacenar las posiciones de listaPalabra y listaAyuda ya utilizadas en la ejecución actual del juego.

Una palabra es elegida con un random que debe tener la condición no haber si usado. Si lo fué, deberá buscarse otro random hasta que se cumpla la condición.
Si el juego se queda sin palabras no repetidas en el diccionario, entonces el juego sale.
Cada vez que es utilizada una letra se crea un nuevo registro en posicionesOcupadas.
Al final retorna la la palabra nueva para adivinar.

```
def cargarPosiciones(letras, posX, posY, ocupados):#Cargar listas posX y posY en ubicaciones aleatorias
    for i in letras:
        posibleX = random.randrange(50,750)
        while (estaCerca(posibleX,ocupados)): # Busca una posición lejana para la letra, si no la encuentra seguirá buscando un random que si
lo esté
            posibleX = random.randrange(50,750)
          #  print("No encuentro uno lejos") # SE USABA PARA DEBBUGEAR
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
        posX.append(posibleX)
        ocupados.append(posibleX) # añade el registro de la posición ocupada para que no se vuelva a utilizar
        posY.append(random.randrange(50,500))
    pass
```

Se utilizó el parámetro ocupados para cumplir con el requisito de que no se repitan las mismas palabras para adivinar. Es así que la lista ocupados cumple el rol de almacenar los valores de posX ya utilizadas en la ayuda mediante la cual se muestran las palabras del país desordenadas en la pantalla.

Una posX es elegida con un random que debe tener la condición no haber si usado. Si lo fué, deberá buscarse otro random hasta que se cumpla la condición.

```
def cargarAyuda(listaAyuda, listaPalabra, palabra):#Retornar sinonimo
    return listaAyuda[damePosicion(listaPalabra,palabra)]


def damePosicion(listaPalabra, palabra):#Devuelve la posicion de la palabra en listaPalabra
    for i in range(0,len(listaPalabra)):
        if listaPalabra[i] == palabra:
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
            return i
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
def estaCerca(elem, lista):        # Control de superposicion (elem es el candidato a utilizar esa posición, lista debe contener el listado
con las posiciones utilizadas)

    for i in lista:                # Por cada lugar de la lista
        if (i > elem):             # Si la pos del lugar es más grande que el candidato actual
            if (i - elem) <= 10:   # Y la diferencia es menor que diez
                return True        #Entonces está cerca
        else:
            if (i < elem):          # Si la pos del lugar es más pequeña que el candidato actual
                if (elem - i) <= 10:# Y la diferencia es menor que diez
                    return True     #Entonces está cerca
            else:
                if (i == elem):     # Si la pos del lugar es igual al candidato actual
                    return True     #Entonces está cerca
    return False                    # Si supera todo el bucle, entonces está lejos
```

Para saber si una posición no fue utilizada o se superponen porque está muy cerca, entonces mediante un bucle se obtiene la diferencia cada x ya utilizada con la candidata a x actual. Si la diferencia (o distancia) es menor a 10, entonces retorna True para indicar que está cerca, de lo contrario False.

```
def esCorrecta(candidata, palabra):#comprobar palabra ingresada por teclado
    if candidata == palabra:
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
        return True
    else:
        return False


def puntuar(palabra): #puntuacion

    puntos=0
    for i in range(len(palabra)): # por cada letra
        if i == "a" or i == "e" or i == "i" or i == "o" or i == "u": # si son vocales
            puntos += 1 #suma un punto
        else:
            if i == "j" or i == "k" or i == "q" or i == "w" or i == "x" or i == "y" or i == "z": #si son letras dificiles
                puntos += 5                                         #suma 5 puntos
            else:
                puntos +=2       #en los demás casos suma 2 puntos
    return puntos
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

Se crearon las funciones auxiliares:

```
def vaciarLista(lista): # Esta función deja con cero elementos la lista que se pasa como argumento
    for i in range(len(lista)):
        lista.pop()
```

Esta simple función, elimina cada elemento de la lista que se le pasa como argumento.
Esto se utiliza así porque se descubrió que nombreDeLista = [] no vacía la lista si la misma fué previamente definida.

```
def fueUsadaLaPalabra(pos,posicionesOcupadas): # Indica si una palabra fue utilizada comparada por su posición
    for i in range(0,len(posicionesOcupadas)):
        if posicionesOcupadas[i] == pos:
            return True
    return False
```

Se creó esta función para saber si una palabra (mejor dicho su posición que la identifica en listaPalabra) ya fue utilizada en la ejecución actual. Para ello se recorre con un bucle la lista que se encarga de guardar el registro de uso (posicionesOcupadas) y evalúa cada elemento con la posición candidata actual (pos). Si hay algún valor coincidente, retorna True porque la palabra ya fue usada, de lo contrario False.

```
def cambiarTiempo(penalidades,penaliza):
    if penaliza is False:
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
        penalidades += 5
        return penalidades
    else:
        penalidades -= 5
        return penalidades
```

Esta función ayuda a modificar el tiempo restante. Si el segundo parámetro es True, entonces aplica una penalización restando 5 segundos de tiempo, si es False, entonces aplica un premio de 5 segundos.
La variable penalidades contiene la suma (y resta) de todos los premios y penalidades.

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

PENALIZACIONES Y PREMIOS EN TIEMPO

Se modificó parte de la programación principal para aplicar castigos que restan 5 segundos por tener una respuesta incorrecta, pasar de país, o tardar mucho en responder. También se aplicó un premio de 5 segundos extras por cada acierto. La penalización sólo tiene lugar cuando el tiempo restante es mayor a 10 segundos, de lo contrario, el usuario tiene vía libre para pasar palabras sin penalización.

Se tuvo que inicializar una nueva variable:

```
penalidades = 0             # Variable que incorpora el tiempo de penalidad o premio por acertar o no
```

En el bloque de código de cuando se saltea un país:

```
if segundos > 10:                                  # Si todavía quedan más de 10 segundos, aplicar penalidad
       penalidades = cambiarTiempo(penalidades,True)
 # Si quedan menos de 10 segundos no se aplica para que no haya inconveniente en ejectar el bucle donde se muestra el ranking. De paso el usuario puede
aproecharse y pasar palabras las veces que pueda en 10 segundos.
```

En el bloque de código de cuando la respuesta es correcta:

```
penalidades = cambiarTiempo(penalidades,False)  # 5 segundos de regalo, por acertar la palabra
```

En el bloque de código de cuando tarda mucho en responder:

```
if segundos > 5:
                    penalidades = cambiarTiempo(penalidades,True)              # Si pasó el tiempo y no acertó aplica penalidad
```

En el bloque de código de cuando se calcula los segundos restantes:

```
segundos = penalidades + TIEMPO_MAX - pygame.time.get_ticks()/1000  #Se modificó esta fórmula para que aplique permanente las penalidades o premios en tiempo
```

EFECTOS DE SONIDO

Se agregaron efectos de sonido. Música para todo el juego, excepto al finalizar. Sonido para tecla presionada, sonido para pasar palabra, sonido cuando se acierta. También una música especial si se llega a los 10 mejores.

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
song= pygame.mixer.Sound("song.ogg")        # Música del juego, carga el archivo
 song.set_volume(0.2)                        # Establecer el volumen como una cortina de fondo, bajito
 song.play()                                 # Darle play a la música
 tecla = pygame.mixer.Sound("sou.ogg")       # Efecto de sonido para cuando se presiona una tecla
 pasar = pygame.mixer.Sound("pasar.ogg")     # Efecto de sonido cuando se hace pasapalabra
 corre = pygame.mixer.Sound("correcta.ogg")  # Efecto de sonido cuando se acierta a la palabra
 ganaste = pygame.mixer.Sound("win.ogg")     # Efecto de sonido cuando se gana en el ranking
 pasar.set_volume(0.5)                       # Corrección de volumen
 tecla.set_volume(0.2)                       # Corrección de volumen
```

En el bloque de código de cuando se presiona una tecla:

```
if e.type == KEYDOWN:
                tecla.play()
```

En el bloque de código de cuando se salta palabra

```
if e.key == K_RETURN :
                    if candidata=='1': #Siguiente palabra
                        pasar.play()
```

En el bloque de código de cuando se acierta

```
 if(esCorrecta(candidata, palabra)): #acerto
                                  corre.play()
```

En el bloque de código cuando el usuario llega al top 10

```
if (resultado is not False): #Si merece estar en el ranking
        pygame.time.wait(1000)       # para que no se repitan letras si el usuario tarda menos de 100 milisegundos en levantar el dedo de la tecla
        ganaste.set_volume(0.5)
        ganaste.play()
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

QUE LOS CARACTERES QUE INGRESA EL USUARIO NO SUPERE LA CANTIDAD DE CARACTERES DEL PAÍS A ADIVINAR

```
if e.type == KEYDOWN:
        tecla.play()
        letra = dameLetraApretada(e.key)
        if len(candidata) < len(palabra):
            candidata += letra
```

Que sólo se agregen caracteres a candidata si su longitud es menor a la longitud de la palabra

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

RANKING TOP 10

Para incorporar un ranking top 10 se debió reutilizar y modificar código de la fuente principal, así como reutilizar funciones como dibujar(). También se crearon 2 archivos txt que contienen nombres y puntos respectivamente. El juego al escapar del bucle segundos > fps/1000 ingresa en dos posibles bucles. En el primero sólo si está dentro del ranking y en el segundo si no quedo dentro.

Desde líneas 142 a 288 son modificaciones de principal.py con comentarios para su correcto funcionamiendo. A su vez, debieron crearse varias funciones auxiliares ubicadas en el archivo ranking.py

```
        screen.fill(COLOR_FONDO)                # Limpiar pantalla
    pygame.display.flip()                   # Actualizar pantalla
    song.set_volume(0)                      # Que la música no se escuche más



    # Resultado es la posicion que ocupa en el ranking, o es False si no entra al ranking
    resultado = entraEnRanking(puntos,listaRankingNombre,listaRankingPuntos)



    if (resultado is not False): #Si merece estar en el ranking

        pygame.time.wait(1000)       # para que no se repitan letras si el usuario tarda menos de 100 milisegundos en levantar el dedo de la tecla
        ganaste.set_volume(0.5)
        ganaste.play()


        while enterPresionado == False:          #Nuevo bucle para ver ranking y escribir nombre
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
            font = pygame.font.Font(None, 30)   #Inicializar fuentes (para escribir)

            screen.fill(COLOR_FONDO)        #Limpiar pantalla cada vez que itere el ciclo


            for e in pygame.event.get():    #evento que escucha si se produjo el evento que sale del programa
                if e.type == QUIT:
                    pygame.quit()
                    return









            if e.type == KEYDOWN:           #evento que escucha las letras presionadas
                tecla.play()
                letra = dameLetraApretada(e.key)
                print (letra)
                nombreuser += letra
                pygame.time.wait(135)       # para que no se repitan letras si el usuario tarda menos de 100 milisegundos en levantar el dedo de la tecla
                if e.key == K_BACKSPACE:
                    nombreuser = nombreuser[0:len(nombreuser)-1]    # Borra el último caracter si escribió backspace
                if e.key == K_RETURN :
                    escribirNuevoRanking(resultado,nombreuser,puntos,listaRankingNombre,listaRankingPuntos) # Al darle enter, ingresar los datos
definitivamente al ranking
                    enterPresionado = True
                                                                                    # Salir del programa
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
            screen.blit(font.render("Ingresa tu nombre y presiona enter", 1, COLOR_LETRAS), (50,500))    # Cartel que le dice al usuario que puede ingresar su
nombre


            # Funcion que muestra el ranking y en vivo la posicion que ocupa el usuario y cambia su nombre conforme lo tipea
            imprimirRankingMerecido(listaRankingNombre,listaRankingPuntos,screen,resultado,nombreuser,puntos)




            # Reutilizar la funcion que dibuja con valores especificos.
                # No se necesitan letras sueltas por ahi. Entonces,
                # nada para la letrasEnPantalla
                # nada para posiciones con PosX y posY
                # ahora la candidata es el nombre del usuario
                # no hay palabra que adivinar asi que "" (nada)
                # En la ayuda puede mostrar un cartel que diga "Ingresa tu nombre y presiona enter"
                # -e-n- -s-e-g-u-n-d-o-s- -q-u-e- -m-u-e-s-t-r-e-n- -l-o-s- -s-e-g-u-n-d-o-s- -q-u-e- -q-u-e-d-a-n- -p-a-r-a- -q-u-e-
                # c-o-m-p-l-e-t-e- -s-u- -n-o-m-b-r-e-
                # ya no usamos los segundos como límite. Esperamos que se presione enter
                # en t0 o t1 ya no tiene importancia la distancia entre el momento actual y el momento de la aparición de la palabra. se le pasa el valor de
segundos
                # screen es el puntero que maneja la pantalla, hay que pasarlo
                # se quieren mostrar los puntos actuales, asi que tambien se pasan los puntos
            dibujar(nada, nada, nada, nombreuser, "", "Ingresa tu nombre y enter", 0, 0, 0, screen, puntos) #Segundo llamado



            # Actualizar pantalla
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
            pygame.display.flip()


    else:       #Si NO merece estar en el ranking
        while enterPresionado == False:          #Nuevo bucle para ver ranking y escribir nombre



            font = pygame.font.Font(None, 30)   #Inicializar fuentes (para escribir)

            screen.fill(COLOR_FONDO)        #Limpiar pantalla cada vez que itere el ciclo


            for e in pygame.event.get():
                if e.type == QUIT:
                    pygame.quit()
                    return

            if e.type == KEYDOWN:           # Se espera enter para salir
                if e.key == K_RETURN :
                    pygame.quit()
                    return




             # Funcion que muestra SOLO el ranking
            imprimirRanking(listaRankingNombre,listaRankingPuntos,screen)
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
            # Reutilizar la funcion que dibuja con valores especificos.
                # No se necesitan letras sueltas por ahi. Entonces,
                # nada para la letrasEnPantalla
                # nada para posiciones con PosX y posY
                # ahora la candidata es el nombre del usuario PERO NO PUEDE ESCRIBIRLO PORQUE NO ENTRO AL RANKING
                # no hay palabra que adivinar asi que "" (nada)
                # En la ayuda puede mostrar un cartel que diga "Más suerte para la próxima"
                # -e-n- -s-e-g-u-n-d-o-s- -q-u-e- -m-u-e-s-t-r-e-n- -l-o-s- -s-e-g-u-n-d-o-s- -q-u-e- -q-u-e-d-a-n- -p-a-r-a- -q-u-e-
                # c-o-m-p-l-e-t-e- -s-u- -n-o-m-b-r-e-
                # ya no usamos los segundos como límite. Esperamos que se presione enter
                # en t0 o t1 ya no tiene importancia la distancia entre el momento actual y el momento de la aparición de la palabra. se le pasa el valor de
segundos
                # screen es el puntero que maneja la pantalla, hay que pasarlo
                # se quieren mostrar los puntos actuales, asi que tambien se pasan los puntos
            dibujar(nada, nada, nada, nombreuser, "", "Más suerte para la próxima", 0, 0, 0, screen, puntos) #Segundo llamado




            # Actualizar pantalla
            pygame.display.flip()

    while 1:



        for e in pygame.event.get():
            if e.type == QUIT:
                pygame.quit()
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
            return

        if e.type == KEYDOWN:                   # Se espera enter para salir
            if e.key == K_RETURN :
                pygame.quit()
                return
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

RANKING.PY

```
def leerRanking(listaRankingNombre,listaRankingPuntos):
    archivoRankingNombre = open("ranking_nombre.txt","r")
    lineasRankingNombre = archivoRankingNombre.readlines()
    for linea in lineasRankingNombre:
        linea = linea.replace("\n","")
        listaRankingNombre.append(linea)
    archivoRankingNombre.close()

    archivoRankingPuntos = open("ranking_puntos.txt","r")
    lineasRankingPuntos = archivoRankingPuntos.readlines()
    for linea in lineasRankingPuntos:
        linea = int(linea.replace("\n",""))
        listaRankingPuntos.append(linea)
    archivoRankingPuntos.close()
```

La función leerRanking() carga en correspondientes entre sí dos listas con nombres y puntajes.

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

RANKING.PY

```
def entraEnRanking(puntos,listaRankingNombre,listaRankingPuntos):
for r in range(len(listaRankingPuntos)):
    if puntos > listaRankingPuntos[r]:
        for s in range(len(listaRankingPuntos)-1,r,-1):
            listaRankingPuntos[s] = listaRankingPuntos[s-1]
            listaRankingNombre[s] = listaRankingNombre[s-1]

        listaRankingPuntos[r] = puntos
        listaRankingNombre[r] = ""
        return r
return False
```

La función entraEnRanking() devuelve la posición que ocupa el nuevo record del jugador o retorna False si no merece estar en el top 10.

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

RANKING.PY

```
    def escribirNuevoRanking(pos,nombre, puntos,listaRankingNombre,listaRankingPuntos):
listaRankingNombre[pos] = puntos
listaRankingNombre[pos] = nombre

archivoRankingNombre = open("ranking_nombre.txt","w")
archivoRankingNombre.seek(0)
archivoRankingNombre.truncate()
for linea in listaRankingNombre:
    archivoRankingNombre.write(linea)
    archivoRankingNombre.write("\n")
archivoRankingNombre.close()

archivoRankingPuntos = open("ranking_puntos.txt","w")
archivoRankingPuntos.seek(0)
archivoRankingPuntos.truncate()
for linea in listaRankingPuntos:
    archivoRankingPuntos.write(str(linea))
    archivoRankingPuntos.write("\n")
archivoRankingPuntos.close()
```

La función escribirNuevoRanking() escribe los nuevos valores del ranking en ficheros txt. Los abre, borra su contenido, reemplaza el nuevo valor adquirido, y graba en el txt linea por linea, registro por regristo.

RANKING.PY

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
    def imprimirRankingMerecido(listaRankingNombre,listaRankingPuntos,screen,pos,nombreuser,puntos):
    font = pygame.font.Font(None, 30)  #Inicializar fuentes (para escribir)

    y = 100
    for r in range(len(listaRankingPuntos)):

        if r == pos:
            screen.blit(font.render("Pos: "+ str(r+1), 1, COLOR_LETRAS), (50,y))
            screen.blit(font.render(nombreuser[0:10], 1, COLOR_LETRAS), (200,y))
            screen.blit(font.render("       Puntos: " + str(puntos), 1, COLOR_LETRAS), (450,y))
            y = y + 30

        else:

            screen.blit(font.render("Pos: "+ str(r +1), 1, COLOR_LETRAS), (50,y))
            screen.blit(font.render(listaRankingNombre[r][0:10], 1, COLOR_LETRAS), (200,y))
            screen.blit(font.render("       Puntos: " + str(listaRankingPuntos[r]), 1, COLOR_LETRAS), (450,y))
            y = y + 30
```

La función imprimirRankingMerecido() muestra el top 10 actual y permite mostrar como se escribe y edita el nombre del jugador en la posición actual que va a ocupar en el ranking. Por cada registro del ranking renderiza 3 lineas en pantalla. Si la linea a renderizar es justo la del jugador, entonces muestra lo escrito actualmente, sino los registros ya almacenados.

RANKING.PY

```
def imprimirRanking(listaRankingNombre,listaRankingPuntos,screen):
    font = pygame.font.Font(None, 30)  #Inicializar fuentes (para escribir)
```

Alumnos: Ignacio Mariano Tula (itula@edsm.com.ar)
Juan Emmanuel Colace (emmanueli33.0@gmail.com)
Materia: Introducción a la Programación **Com:** 5
Profesores: Daniel Bressky y Esteban Fassio

```
    y = 100
    for r in range(len(listaRankingPuntos)):



        screen.blit(font.render("Pos: "+ str(r +1), 1, COLOR_LETRAS), (50,y))
        screen.blit(font.render(listaRankingNombre[r][0:10], 1, COLOR_LETRAS), (200,y))
        screen.blit(font.render("       Puntos: " + str(listaRankingPuntos[r]), 1, COLOR_LETRAS), (450,y))
        y = y + 30
```

La función `imprimirRanking`() muestra el top 10 actual. Por cada registro del ranking renderiza 3 lineas en pantalla. Si la linea a renderizar es justo la del jugador, entonces muestra lo escrito actualmente, sino los registros ya almacenados. En este caso no se muestra la edición del nombre del usuario porque no tiene suficientes puntos para estar en el ranking.