

Python (Práctica 4)

Objetivo

Se trata de abordar un programa algo más serio, en el que las entradas del usuario influyen en el curso de la ejecución. Se usan, también, procedimientos de manipulación de cadenas de texto.

Programa 6: Ahorcado

La cosa se complica. El siguiente es el clásico juego del ahorcado. Pero fíjate que un buen programador debe escribir su código de forma que sea ampliable en el futuro. De ahí el uso intensivo de las funciones.

```
# -*- coding: utf-8 -*-
```

```
import random
```

```
DIBUJOS = [ '''
```

```
  +-----+
  |
  |
```

```
===== ''' , '''
```

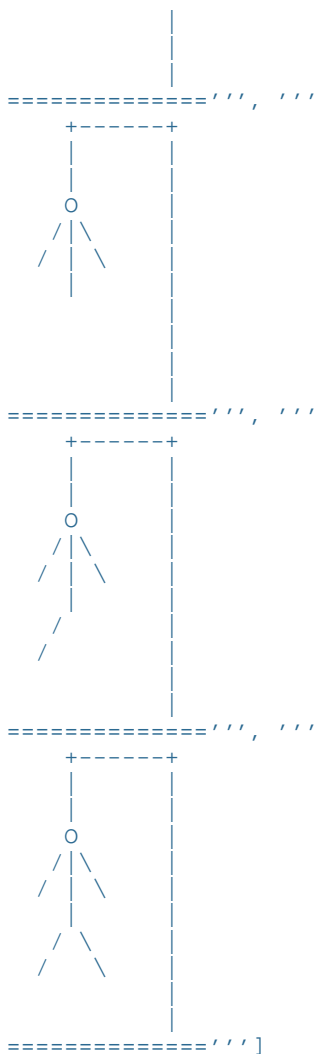
```
  +-----+
  |
  | O
  |
```

```
===== ''' , '''
```

```
  +-----+
  |
  | O
  |
  |
```

```
===== ''' , '''
```

```
  +-----+
  |
  | O
  |
  | /
  |
```



```
palabras = 'hormiga perro canario oso abeja caballo camello gato tigre mofeta cobra ' + \
'chica coyote ojo cuervo mano perro mono pato cuervo coche pez zorro rana pastel ' + \
'bizcocho tomate iguana chacal koala leche agua mago burro amigo miel gorila rata ' + \
'boca mula red fuerza lechuza tren metro loro pigmeo python camino conejo charco ' + \
'buitre rinoceronte ballena medusa cordero ternera estrella nube nariz vientre ' + \
'piedra pantano cisne trucha avispa caballero verdad inglaterra tortuga tortilla ' + \
'placton huesca vampiro normando vikingo cebra'.split()
```

```
def palabraAleatoria(listaPalabras):
    # Esta función devuelve una palabra aleatoria entre un conjunto de palabras.
    cual = random.randint(0, len(listaPalabras) - 1)
    return listaPalabras[cual]

def mostrarAhorcado(DIBUJOS, letrasMalas, letrasBuenas, palabraSecreta):
    print DIBUJOS[len(letrasMalas)]
    print
    print 'Errores:',
    for letra in letrasMalas:
        print letra,
    print
    huecos = '_' * len(palabraSecreta)
    for i in range(len(palabraSecreta)): # reemplazar huecos con letras acertadas
        if palabraSecreta[i] in letrasBuenas:
            huecos = huecos[:i] + palabraSecreta[i] + huecos[i+1:]
    for letra in huecos: # mostrar palabra secreta con huecos entre letras
        print letra,
    print

def intento(yaIntentado):
    # Devuelve la letra intentada. Se asegura que se introduce una letra y no otra cosa.
    while True:
```

```

    print 'Intenta con una letra.'
    candidata = raw_input()
    candidata = candidata.lower()
    if len(candidata) != 1:
        print 'Por favor, introduce sólo una letra.'
    elif candidata in yaIntentado:
        print 'Ya has intentado esa letra. Elige otra.'
    elif candidata not in 'abcdefghijklmnopqrstuvwxyz':
        print 'Por favor, introduce una LETRA.'
    else:
        return candidata

def jugarOtraVez():
    # Esta función devuelve True si el jugador quiere jugar otra vez.
    print '¿Quieres jugar otra vez? (s ó n)'
    return raw_input().lower().startswith('s')

print 'A H O R C A D O'

letrasFalladas = ''
letrasAcertadas = ''
palabraSecreta = palabraAleatoria(palabras)
juegoTerminado = False

while True:
    mostrarAhorcado(DIBUJOS, letrasFalladas, letrasAcertadas, palabraSecreta)

    # Pedir al jugador que introduzca una letra.
    pruebaLetra = intento(letrasFalladas + letrasAcertadas)

    if pruebaLetra in palabraSecreta:
        letrasAcertadas = letrasAcertadas + pruebaLetra
        # Mirar si el jugador ha ganado
        encontradasTodas = True
        for i in range(len(palabraSecreta)):
            if palabraSecreta[i] not in letrasAcertadas:
                encontradasTodas = False
                break
        if encontradasTodas:
            print '¡Sí! ¡La palabra secreta es "' + palabraSecreta + '"! ¡Has ganado!'
            juegoTerminado = True
    else:
        letrasFalladas = letrasFalladas + pruebaLetra
        # Mirar si son demasiados intentos y el jugador pierde
        if len(letrasFalladas) == len(DIBUJOS) - 1:
            mostrarAhorcado(DIBUJOS, letrasFalladas, letrasAcertadas, palabraSecreta)
            print '¡Demasiados intentos!\nDespués de ' + str(len(letrasFalladas)) + \
                ' intentos fallidos y ' + str(len(letrasAcertadas)) + \
                ' intentos correctos, la palabra era "' + palabraSecreta + '"'
            juegoTerminado = True

    # Preguntar al jugador si quiere jugar otra vez (pero sólo si se ha terminado).
    if juegoTerminado:
        if jugarOtraVez():
            letrasFalladas = ''
            letrasAcertadas = ''
            juegoTerminado = False
            palabraSecreta = palabraAleatoria(palabras)
        else:
            break

```

Como viene siendo habitual, una vez que compruebes que funciona, modifica el programa y mejóralo (incluso puedes cambiarlo por completo). Envía tu nuevo programa por correo electrónico con el nombre de **ahorcado.py** junto con su diagrama de flujo y el resultado de su ejecución. Pero...

¡¡Detente!!

La complejidad de este guión es mucho mayor, así que muy probablemente necesitarás un poco de orientación para comprenderlo:

Comprendiendo el juego del Ahorcado

En este juego empleamos multitud de elementos del lenguaje Python. Es interesante que analices el código con detenimiento antes de seguir. **Responde para ello al siguiente cuestionario**; te ayudará a progresar como programador o programadora... Hazlo en un documento de **Google Docs** (identifícalo con tu **nick** y **Cuestionario Python Práctica 4**) y compártelo con tu profesor.

1

En primer lugar, el código del juego está dividido en dos partes claramente diferenciadas. La primera se encarga de realizar las definiciones necesarias y, en ella, Python no hace nada en concreto, sólo toma nota de variables y definiciones.

Indica de qué tipo son los siguientes objetos y de qué se encargan:

- DIBUJOS
- palabras
- palabraAleatoria()
- mostrarAhorcado()
- intento()
- jugarOtraVez()

2

La segunda parte es la acción de verdad y comienza después de todas las definiciones de las funciones. El código implicado es, fundamentalmente, el siguiente:

```
....  
  
print 'A H O R C A D O'  
  
letrasFalladas = ''  
letrasAcertadas = ''  
palabraSecreta = palabraAleatoria(palabras)  
juegoTerminado = False  
  
while True:  
    ...
```

¿Para qué sirve cada una de esas cuatro variables que se definen? Por cierto, la condición del bucle **while** es siempre verdadera. ¿Cómo se las arregla entonces el juego para terminar?

3

Explica cómo trabaja (es decir, qué argumentos toma, qué hace con ellos y qué devuelve) la función **palabraAleatoria()**.

```
def palabraAleatoria(listaPalabras):  
    # Esta función devuelve una palabra aleatoria entre un conjunto de palabras.  
    cual = random.randint(0, len(listaPalabras) - 1)  
    return listaPalabras[cual]
```

Sugerencia: Observa cada línea, pregúntate y apúntate qué hace cada instrucción y para qué. Una vez que comprendas todas las líneas del cuerpo de la función, será fácil comprender el conjunto. Puedes incluso hacerte un mini-diagrama de flujo para ayudarte. Hazte preguntas, al mismo tiempo, del estilo de ¿qué tipo de dato es **listaPalabras**? ¿qué hace **len()**?...

4

Explica, de la misma forma, cómo trabaja la función **mostrarAhorcado()**.

```
def mostrarAhorcado(DIBUJOS, letrasMalas, letrasBuenas, palabraSecreta):
    print DIBUJOS[len(letrasMalas)]
    print
    print 'Errores:',
    for letra in letrasMalas:
        print letra,
    print
    huecos = '_' * len(palabraSecreta)
    for i in range(len(palabraSecreta)): # reemplazar huecos con letras acertadas
        if palabraSecreta[i] in letrasBuenas:
            huecos = huecos[:i] + palabraSecreta[i] + huecos[i+1:]
    for letra in huecos: # mostrar palabra secreta con huecos entre letras
        print letra,
    print
```

5

Y, análogamente, ¿cómo trabaja la función **intento()**?

```
def intento(yaIntentado):
    # Devuelve la letra intentada. Se asegura que se introduce una letra y no otra cosa.

    while True:
        print 'Intenta con una letra.'
        candidata = raw_input()
        candidata = candidata.lower()
        if len(candidata) != 1:
            print 'Por favor, introduce sólo una letra.'
        elif candidata in yaIntentado:
            print 'Ya has intentado esa letra. Elige otra.'
        elif candidata not in 'abcdefghijklmnopqrstuvwxyz':
            print 'Por favor, introduce una LETRA.'
        else:
            return candidata
```

6

Para acabar con las funciones, explica la última, **jugarOtraVez()**.

```
def jugarOtraVez():
    # Esta función devuelve True si el jugador quiere jugar otra vez.
    print '¿Quieres jugar otra vez? (s ó n)'
    return raw_input().lower().startswith('s')
```

7

Fijémonos ahora en el bucle **while**, que se encarga del ensamblar todas las partes y hacer que el juego funcione:

```
while True:
    mostrarAhorcado(DIBUJOS, letrasFalladas, letrasAcertadas, palabraSecreta)

    # Pedir al jugador que introduzca una letra.
    pruebaLetra = intento(letrasFalladas + letrasAcertadas)

    if pruebaLetra in palabraSecreta:
        letrasAcertadas = letrasAcertadas + pruebaLetra
        # Mirar si el jugador ha ganado
        encontradasTodas = True
        for i in range(len(palabraSecreta)):
            if palabraSecreta[i] not in letrasAcertadas:
                encontradasTodas = False
                break
        if encontradasTodas:
            print '¡Sí! ¡La palabra secreta es "' + palabraSecreta + '"! ¡Has ganado!'
            juegoTerminado = True
    else:
        letrasFalladas = letrasFalladas + pruebaLetra
        # Mirar si son demasiados intentos y el jugador pierde
        if len(letrasFalladas) == len(DIBUJOS) - 1:
            mostrarAhorcado(DIBUJOS, letrasFalladas, letrasAcertadas, palabraSecreta)
            print '¡Demasiados intentos!\nDespués de ' + str(len(letrasFalladas)) + \
                ' intentos fallidos y ' + str(len(letrasAcertadas)) + \
                ' intentos correctos, la palabra era "' + palabraSecreta + '"'
            juegoTerminado = True

    # Preguntar al jugador si quiere jugar otra vez (pero sólo si se ha terminado).
    if juegoTerminado:
        if jugarOtraVez():
            letrasFalladas = ''
            letrasAcertadas = ''
            juegoTerminado = False
            palabraSecreta = palabraAleatoria(palabras)
        else:
            break
```

El bucle realiza tareas que pueden dividirse en cuatro partes. **Escribe el código de cada una de ellas y explica cómo funciona (desglosando cada una de sus líneas):**

- Dibujar el ahorcado
- Pedir al jugador que introduzca una letra
- Mirar si esa letra es correcta o no
- Comprobar si el juego ha de finalizarse

Recapitulación

Independientemente del cuestionario anterior y como hemos hecho en cada práctica, aquí tienes un listado de algunas de las nuevas características de Python que han aparecido o están relacionadas:

- ¿Qué diferencias hay entre las listas y las tuplas? ¿Para qué sirven?
- ¿De qué formas pueden extraerse los elementos de las listas y las tuplas?
- ¿Qué son los diccionarios en Python? ¿Cómo se usan?
- Las funciones `split()` y `len()`.
- ¿Qué son los métodos? ¿Qué diferencia hay con las funciones?
- ¿Qué son los atributos? ¿Qué diferencia hay entre una clase y un objeto?
- Otras operaciones con cadenas de texto. ¿Funcionan, también, como listas?
- El bloque `for`. ¿Qué son los iterables?
- ¿Qué son los argumentos o parámetros de una función?
- ¿Para qué sirve el `else` en un `if`? ¿Cuál es su forma general?
- El operador `in` ¿dónde puede usarse? ¿Qué otros tipos de operadores conoces?

Documentate y pregunta todo lo que necesites... ¡No te quedes con dudas!