

Python (Práctica 5)

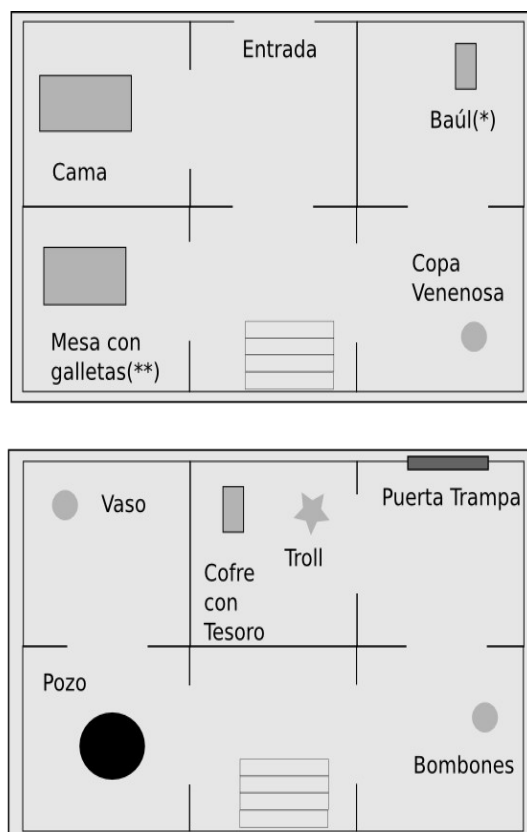
Objetivo: Escribiendo un Juego Conversacional

Vamos a seguir los pasos necesarios, con los conocimientos adquiridos, para esbozar un pequeño juego conversacional. Vamos a ponernos en la piel del programador, pensando cómo quiere que sea el juego e implementando el código de acuerdo a esas necesidades.

Programa 6: La Mansión

En **La Mansión** queremos tener una historia en la que el jugador debe recorrer una casa maldita con el objetivo de encontrar un tesoro, una piedra preciosa de incalculable valor. Estará encerrada en un cofre y protegida por un troll.

Haremos el juego sencillo, de manera que la casa tendrá sólo 2 pisos y, en cada piso, 6 habitaciones. Debemos decidir qué hay en cada lugar y qué obstáculos va a encontrar, así que lo mejor es esbozar un **plano** y situar allí los elementos. Éste va a ser el nuestro:



(*) Dentro del baúl hay una espada.

(**) Debajo de la mesa hay una llave.

Por cierto, los bombones están envenenados, también. La comida y la bebida son necesarias pues nuestro protagonista tendrá que beber y comer para sobrevivir...

Bien. Los **juegos conversacionales** (el término técnico es **ficción interactiva**) funcionan en base a una comunicación escrita. Eso quiere decir que se pide al jugador que escriba las acciones que quiere realizar. Como vamos a usar el idioma español con sus acentos, caracteres especiales y similares, es evidente que una de las primeras líneas del programa tendrá que ser la familiar

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

Sigamos. Nuestro **parser** (palabra también técnica que indica la parte del programa que procesa la comunicación entre el usuario y el ordenador) debe entender frases sencillas, pongamos que en infinitivo. Así que tendremos que almacenar los verbos en una lista. ¿Por qué? Por que bastará ver si el verbo que ha introducido el jugador está en la lista para saber qué hacer o indicar que no se entiende. ¿Qué verbos usar? Ésta va a ser nuestra lista:

```
verbos = ['ir', 'coger', 'abrir', 'atacar', 'hablar', 'beber', 'comer', 'saltar', \
          'subir', 'bajar', 'inventario', 'examinar']
```

La mayor parte de los verbos son autoexplicativos. Por ejemplo, tenemos un verbo 'atacar' por que tendremos que luchar con el troll del segundo piso. Y un verbo 'saltar' por que tendremos que atravesar el pozo (mágico) sin caernos...

Para desplazarnos, tendremos que usar direcciones (una forma típica es decir cosas del estilo de 'ir norte' etc). Añadamos de la misma forma

```
direcciones = ["norte", "sur", "este", "oeste"]
```

Como puede imaginarse, un juego real es bastante más complejo (se usan incluso librerías específicas que facilitan la vida al programador) y la forma de comunicarse es mucho más elaborada y flexible. Pero, de momento, nos valdrá lo que tenemos.

También tendremos que interaccionar con cosas. Necesitamos una lista de objetos que podamos coger:

```
objetos = ['llave', 'espada', 'copa', 'vaso', 'galletas', 'bombones']
```

Hemos optado por poner los objetos comestibles y bebibles (tendremos que cogerlos primero para poder tomarlos), la llave que nos permitirá abrir el cofre y la espada para luchar.

Esto nos lleva al concepto de **variables de estado**. ¿Cómo podrá saber Python si el jugador lleva la llave? Este tipo de cuestiones se suelen solucionar con variables booleanas, es decir, las que pueden tomar solamente valores **True** o **False**. Inicialmente, el jugador no tiene la llave. Así que en algún lugar del comienzo del juego definiremos una variable de la manera siguiente:

```
tieneLlave = False
```

Consecuentemente, en el momento en el que durante el juego se coja la llave, la variable tendrá que cambiarse a `True`. Y siempre que deseemos saber si el jugador tiene o no la llave, solo tendremos que consultar el estado de `tieneLlave`.

¿Se te ocurren otras variables de estado? La necesidad de algunas surge a poco que pensemos:

```
tieneEspada = False
haComido = False
haBebido = False
```

Y otras le aparecen al programador a medida que va implementando el programa o le surgen necesidades. Un ejemplo muy típico es

```
juegoAcabado = False
```

¿Para qué lo necesitamos? En los juegos, y en otros muchos programas, generalmente no sabemos cuánto va a durar la ejecución. En nuestro caso, el jugador va realizando acciones una y otra vez, y solo cuando surja una circunstancia extrema (como una muerte o una victoria) hay que detenerlo. Pero esa causa puede surgir en muchos sitios distintos.

La manera de abordar este problema es usar esa variable de estado que acabamos de definir, `juegoAcabado`. Metemos el desarrollo del juego dentro de un bucle infinito, para que el jugador realice sus acciones repetidamente, y cuando `juegoAcabado` sea `True`, salimos con un `break` y procedemos a terminar el juego.

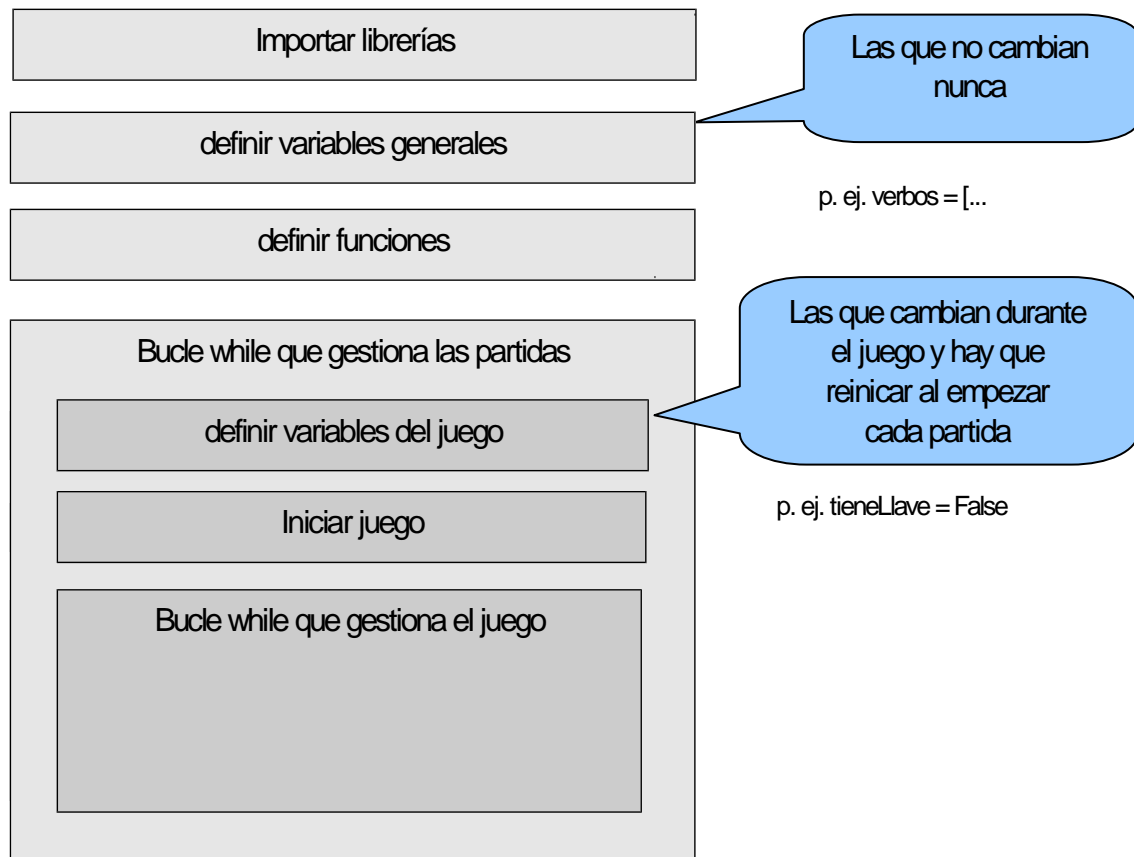
```
while True:
    ...
    ...
    if juegoAcabado:
        break
```

Otra forma es que el propio bucle `while` controle la variable de estado:

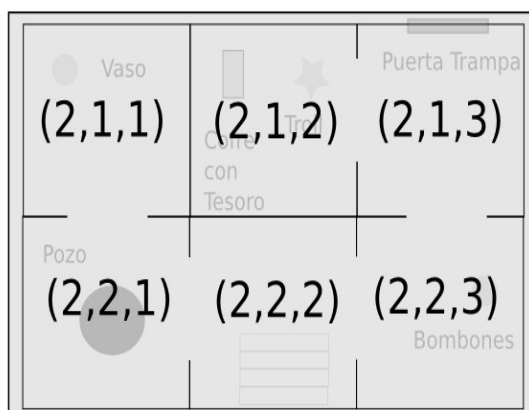
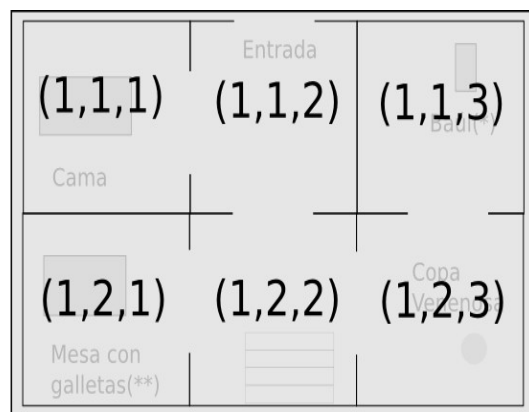
```
while not juegoAcabado:
    ...
    ...
```

Por cierto, en este tipo de juegos, es fácil que el jugador muera y tenga que probar muchas veces hasta conseguir superarlo, así que el propio programa debería ofrecer la posibilidad de volver a jugar. ¿Qué significa esto? Meter todo lo anterior en otro bucle `while` que controle el comienzo del juego tantas veces como haga falta.

El esquema de cómo debe estar organizado el programa sería algo así:



¿Cómo llevar el seguimiento de la posición del protagonista? Lo más simple es una cuadrícula. El mapa de dicha cuadrícula, es un apartado importante del juego:



No solo debemos almacenar las habitaciones que tenemos, si no que para cada una de ellas, tenemos que almacenar por dónde se puede salir. Esta información debemos definirla en el apartado de variables generales, ya que es común para todo el juego y no cambia.

```
salidas = {}  
salidas[(1,1,1)] = ["este"]  
salidas[(1,1,2)] = ["oeste", "sur"]  
...
```

(Observa que, entre las diferentes formas de hacerlo, hemos optado por usar un diccionario; las claves son la posición en la cuadrícula y los valores las direcciones).

Y para empezar el juego, en el apartado de inicializar sus variables, como el jugador comienza en la entrada, pondremos:

```
piso = 1  
fila = 1  
columna = 2
```

Indiquemos ahora dónde están los objetos que pueden cogerse (cuya definición también tendrá que estar aquí pues el jugador puede cambiar su posición durante el juego). Para ello podemos usar otro diccionario, que llamaremos `mapa`, en el que las claves serán la posición en la cuadrícula y el valor correspondiente el objeto:

```
mapa = {}  
mapa[(1,1,1)] = []  
mapa[(1,1,2)] = []  
mapa[(1,1,3)] = []  
mapa[(1,2,1)] = ["galletas"]  
...
```

¡Pasemos a las funciones! ¿Cuales necesitamos?

En el curso del juego, al entrar en cada habitación, necesitaremos que el jugador reciba una descripción de lo que se ve. Eso nos induce a definir una función que lo haga:

```
def describir(piso, fila, columna):  
    ...
```

(como ves, le pasamos a la función como argumentos el piso, la fila y la columna de la habitación que hay que describir).

Por cierto, el juego tiene que tener una introducción:

```
def intro():  
    ...
```

y esto nos recuerda que hay que hacer pausas para presentar el texto, de ahí que al comienzo del código importemos la librería `time` y usemos la función `time.sleep()` para hacerlas cuando lo necesitamos.

No podemos olvidarnos tampoco de nuestro parser, la función que se encarga de tomar el texto que escribe el jugador, analizarlo, y realizar las tareas pertinentes:

```
def procesar(instrucciones):
```

```
    ...
```

Para que la función `procesar` no quede demasiado larga (ya lo es de hecho, ¿verdad?) y por claridad, vamos a hacer dos cosas más aparte. La primera es la parte que se encarga de felicitar al jugador cuando gana...

```
def ganar():
```

```
    ...
```

... y la segunda es la parte que se encarga de gestionar la comida y la bebida, ya que el tema es delicado al existir veneno y comida sana al mismo tiempo.

```
def comerBeber(v,c):
```

```
    ...
```

(para saber si hay que comer o beber y el qué usamos los dos argumentos de la función; `v` es el verbo y `c` lo que hay que comer o beber).

Pero no retrasemos más la presentación del código; el resto de consideraciones es mejor verlas en acción. Lo tienes a continuación. Cópialo en un archivo (ten cuidado con las indentaciones al hacerlo y asegúrate que **respetas el PEP8**, eliminando en Ninja-IDE los errores y las advertencias) y ponle el nombre **TheHouse.py**. Ejecútalo, comprueba que funciona y envíaselo al profesor:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

# Importar los módulos que necesitaremos
import time

# Varias listas con terminología usada en el juego
verbos = ['ir', 'coger', 'abrir', 'atacar', 'hablar', 'beber', 'comer', 'saltar', \
          'subir', 'bajar', 'inventario', 'examinar']
objetos = ['llave', 'espada', 'copa', 'vaso', 'galletas', 'bombones']
direcciones = ["norte", "sur", "este", "oeste"]

# Mapa de salidas de cada habitación, identificadas por 3 números, respectivamente,
# el piso, la fila y la columna
salidas = {}
salidas[(1,1,1)] = ["este"]
salidas[(1,1,2)] = ["oeste", "sur"]
salidas[(1,1,3)] = ["sur"]
salidas[(1,2,1)] = ["este"]
salidas[(1,2,2)] = ["oeste", "norte", "este"]
salidas[(1,2,3)] = ["oeste", "norte"]
```

```

salidas[(2,1,1)] = ["sur"]
salidas[(2,1,2)] = ["este"]
salidas[(2,1,3)] = ["oeste", "norte", "sur"]
salidas[(2,2,1)] = ["norte", "este"]
salidas[(2,2,2)] = ["oeste", "este"]
salidas[(2,2,3)] = ["oeste", "norte"]

# Función que se encarga de describir cada sala, identificada como previamente
# se ha comentado por una tupla de tres números.
def describir(p,f,c):
    # p es el piso
    # f la fila
    # c la columna
    sala = (p,f,c)
    print "-----"
    if sala == (1,1,1):
        print "La habitación está oscura. Hay una cama desecha delante tuyo."
        print "No huele especialmente bien."
    elif sala == (1,1,2):
        print 'El Hall de la casa es más bien sombrío...'
        if trollVivo:
            print 'Se oye crujir algo a lo lejos.'
    elif sala == (1,1,3):
        print "Una pequeña ventana deja vislumbrar un viejo baúl junto a la pared."
        print "Una cucaracha corre a tus pies."
    elif sala == (1,2,1):
        print "Ves una mesa. Sobre ella, algunas galletas y restos líquidos."
    elif sala == (1,2,2):
        print "Una escalera llena de polvo conduce hacia arriba."
        if trollVivo:
            print "En algún lugar, se oyen pasos que se arrastran."
    elif sala == (1,2,3):
        print "En el suelo hay una copa. Las paredes están desconchadas."
    elif sala == (2,1,1):
        print "En una esquina hay vaso con líquido en su interior."
        if trollVivo:
            print "Un sonido produce un eco que no alcanzas a entender."
    elif sala == (2,1,2):
        if trollVivo:
            print "Un troll horrible te observa amenazante y se dirige hacia ti."
            print "Detrás de él puedes observar un cofre."
        else:
            print 'El cadaver de un troll descansa en el suelo junto a un cofre.'
    elif sala == (2,1,3):
        print 'La habitación está desierta...'
        if trollVivo:
            print '...pero notas la presencia de algo vivo no lejos de aquí.'
    elif sala == (2,2,1):
        print "Un enorme pozo se abre delante de ti. No parece tener fin."
    elif sala == (2,2,2):
        print "Estás en el piso superior. No hay casi luz y casi te dan ganas"
        print "de volver a bajar por la escalera y escapar."
    elif sala == (2,2,3):
        print "Hay unos bombones en una estantería. Una gotera en algún lugar"
        print "produce un sonido rítmico y desconcertante."
    # Indicamos hacia qué direcciones puede moverse el jugador
    print 'Direcciones posibles:',
    for s in salidas[sala]:
        print s,
    print
    print "-----"

# Función que muestra en pantalla la introducción del Juego, con pausas.
def intro():
    print "Ante ti está LA MANSIÓN. Te han encargado recuperar una hermosa joya,"
    print "una piedra preciosa de incalculable valor. Sabes que está custodiada y"
    print "que si no tienes valor e inteligencia no saldrás con vida de ella."
    time.sleep(1)
    print
    print "Respiras hondo, te maldices por no venir armado, y entras..."

```

```

time.sleep(1)
print
raw_input("Pulsa <intro> para empezar el juego.")

# Función que se encarga de los actos de comer y beber.
def comerBeber(v,c):
    # v es el verbo; comer o beber
    # c es lo que se quiere comer o beber

    # Declarar los globales
    global inventario, juegoAcabado, haComido, haBebido

    #Comprobar que se lleva lo que se quiere comer
    if c not in inventario:
        print 'No tienes ' + c
        return

    # Caso de comer...
    if v == 'comer':
        # ... galletas
        if c == 'galletas':
            print 'Comes las galletas y te sientes bien.'
            # Activar el indicador de alimentado
            haComido = True
            # Eliminar del inventario, pues se ha comido
            inventario.remove(c)
            return
        # ... bombones
        elif c == 'bombones':
            print '¡Horror! un dolor insoportable te invade...'
            time.sleep(1)
            print 'y mueres retorciendote como una serpiente'
            # Activar el indicador de fin de juego
            juegoAcabado = True
            return
        # ... cualquier otra cosa
        else:
            print 'No puedo comer ' + c
            return
    # caso de beber
    else:
        # ... vaso
        if c == 'vaso':
            print 'Bebes del vaso... agua cristalina. ¡Qué bien!'
            # Activar el indicador de alimentado
            haBebido = True
            # Eliminar del inventario, pues se ha comido
            inventario.remove(c)
            return
        # ... copa
        elif c == 'copa':
            print '¡Sientes un mareo momentáneo...'
            time.sleep(1)
            print '... y mueres fulminantemente'
            # Activar el indicador de fin de juego
            juegoAcabado = True
            return
        # ... cualquier otra cosa
        else:
            print 'No puedo beber ' + c
            return

# Función que procesa las acciones del jugador ('parser')
def procesar(instrucciones):
    # Hay que declarar como globales las variables que pueden modificarse
    global piso, fila, columna, juegoAcabado, inventario, yaDescrito, sinLlave
    global pozoAlSur, bAbierto, tieneEspada, trollActivo, trollVivo
    # Si no se ha escrito nada, no hacer nada
    if instrucciones == '':
        return

```



```

# Mostrar ayuda si se solicita
if instrucciones == 'ayuda':
    print 'Debes usar verbos en infinitivo y, si lo necesitas, un nombre.'
    print 'Acciones disponibles:'
    for i in verbos:
        print i,
    print
    return
# Separar la acción en palabras
palabras = instrucciones.split()
# El verbo ha de ser siempre el primero
verbo = palabras[0]
# Si el verbo es desconocido, no hacer nada y volver a preguntar
if verbo not in verbos:
    print "Perdona, no te entiendo"
    return
# Si el verbo es 'ir'...
if verbo == 'ir':
    # Comprobar que va acompañado de una dirección
    if len(palabras) != 2 or palabras[1] not in direcciones:
        print "No entiendo a dónde tengo que ir."
        return
    # Almacenar la dirección en la variable donde
    donde = palabras[1]
    # Si la dirección elegida no está disponible, comunicarlo
    if donde not in salidas[(piso, fila, columna)]:
        print "No puedo ir hacia el " + donde
        return
    # Desplazarse en la dirección solicitada
    elif donde == "norte":
        # Hay una dos excepciones: La primera es la puerta trampa del segundo
        # piso. Activar el indicador de fin de juego...
        if fila == 1:
            print "¡La puerta era una trampa!"
            time.sleep(1)
            print "Al otro lado hay un precipico y te despeñas por él..."
            juegoAcabado = True
            return
        elif (fila,columna) == (2,1) and not pozoAlSur:
            # La segunda es cuando se ha atravesado el pozo sin saltar
            print 'Avanzas sin cuidado y te adelantas sobre el pozo...'
            time.sleep(1)
            print '¡Caes a la profundidad y la oscuridad del submundo!'
            juegoAcabado = True
            return
        fila = fila - 1
    elif donde == "sur":
        fila = fila + 1
    elif donde == "este":
        # Hay una excepción: Atravesar el pozo sin saltar...
        if (fila,columna) == (2,1) and pozoAlSur:
            print "¡Te has olvidado del pozo!"
            time.sleep(1)
            print "¡La sima del pozo se te traga sin clemencia..."
            juegoAcabado = True
            return
        columna = columna + 1
    else:
        columna = columna - 1
    # Una vez que el jugador se ha movido, confirmarlo y activar el
    # indicador de describir la nueva sala
    print "Vas hacia el " + donde
    yaDescrito = False
    return
# Si el verbo es 'saltar'...
if verbo == 'saltar':
    # Comprobar que va acompañado de una dirección
    if len(palabras) != 2:
        print "No entiendo qué tengo que saltar."
        return

```

```

# Almacenar lo que se salta
elQue = palabras[1]
# Si no es el pozo, comunicarlo
if elQue != 'pozo' :
    print "Qué tontería..."
    return
# saltar pozo
else:
    if (piso, fila, columna) != (2,2,1):
        print 'No hay ningún pozo que saltar.'
    else:
        print 'Saltas el pozo con agilidad...'
        time.sleep(1)
        print '... y lo dejas a tus espaldas.'
        pozoAlSur = not pozoAlSur
    return
# Comprobar que 'subir' ocurre sólo al pie de la escalera.
if verbo == 'subir':
    if (piso, fila, columna) == (1,2,2):
        print 'Subes por la escalera.'
        piso = piso + 1
        yaDescrito = False
    else:
        print 'No puedo subir.'
    return
# Comprobar que bajar ocurre sólo en lo alto de la escalera
if verbo == 'bajar':
    if (piso, fila, columna) == (2,2,2):
        print 'Bajas por la escalera.'
        piso = piso - 1
        yaDescrito = False
    # ¡Ojo, una excepción! Si se quiere bajar por el pozo mágico
    # el jugador muere y hay que activar el indicador de fin de juego.
    elif (piso, fila, columna) == (2,2,1):
        print 'Desciendes por el pozo...'
        time.sleep(0.5)
        print 'Unos ojos brillantes te observan desde el fondo.'
        print '¡Algo te agarra y te devora!'
        juegoAcabado = True
    else:
        print 'No puedo bajar.'
    return
# Gestionar el comer o el beber.
if verbo == 'comer' or verbo == 'beber':
    # Asegurarse que se indica lo que se quiere comer
    if len(palabras) == 1:
        print 'Perdona... ¿el qué?'
        return
    else:
        # Enviar a la función que gestiona el alimento, tanto si
        # come o bebe como el qué
        comerBeber(verbo, palabras[1])
        return
# Coger objetos
if verbo == 'coger':
    # Asegurarse que se indica lo que se quiere coger
    if len(palabras) == 1:
        print 'Perdona... ¿coger qué?'
        return
    else:
        objeto = palabras[1]
        # Comprobar que el objeto está en la sala
        if objeto in mapa[(piso, fila, columna)]:
            # Añadirlo al inventario y quitarlo de la sala
            inventario.append(objeto)
            mapa[(piso, fila, columna)].remove(objeto)
            # Confirmar la acción
            print 'Llevas contigo: ' + objeto
        elif (piso, fila, columna) == (1,2,1) and objeto == "llave" and sinLlave:
            # Añadir la llave al inventario

```

```

        inventario.append(objeto)
        sinLlave = False
        # Confirmar la acción
        print 'Llevas contigo: ' + objeto
    elif (piso, fila, columna) == (1, 1, 3) and objeto == 'espada' and bAbierto:
        if tieneEspada:
            print 'Ya tienes la espada.'
        else:
            # Añadir la espada al inventario
            inventario.append(objeto)
            tieneEspada = True
            # Confirmar la acción
            print 'Llevas contigo: ' + objeto
    else:
        print 'No puedo hacer eso.'
    return
# Abrir objetos
if verbo == 'abrir':
    # Asegurarse que se indica lo que se quiere abrir
    if len(palabras) == 1:
        print 'Perdona... ¿abrir qué?'
        return
    else:
        objeto = palabras[1]
        # Comprobar que el objeto está es el correcto
        if (piso, fila, columna) == (1, 1, 3) and objeto == 'baúl':
            if bAbierto:
                print 'El baúl ya está abierto.'
            else:
                # abrir baúl
                bAbierto = True
                print 'Has abierto el baúl.'
        elif (piso, fila, columna) == (2, 1, 2) and objeto == 'cofre':
            if sinLlave:
                print '¡No puedes abrir el cofre sin una llave!'
            else:
                # El cofre se abre y el juego se ha ganado
                print 'Abres el cofre, lentamente...'
                time.sleep(1)
                ganar()
                juegoAcabado = True
        else:
            print 'No puedo hacer eso.'
    return
# examinar objetos
if verbo == 'examinar':
    if len(palabras) == 1:
        print 'Perdona... ¿examinar qué?'
        return
    else:
        objeto = palabras[1]
        # Comprobar que el objeto es correcto
        if objeto == 'mesa' and (piso, fila, columna) == (1, 2, 1):
            print 'La mesa parece sólida.'
            time.sleep(0.5)
            print 'Miras por debajo...'
            time.sleep(0.5)
            if sinLlave:
                print '... y ves una llave!'
            else:
                print 'No hay nada.'
        elif objeto == 'baúl' and (piso, fila, columna) == (1, 1, 3):
            if bAbierto:
                print 'Miras dentro del baúl...'
                time.sleep(0.5)
                if tieneEspada:
                    print 'Está vacío.'
                else:
                    print '¡Hay una espada!'
            else:

```

```

        print 'El baúl parece cerrado...'
        time.sleep(0.5)
        print '...pero no tiene cerradura.'
    elif objeto == 'cofre' and (piso, fila, columna) == (2, 1, 2):
        print 'Es un cofre de madera regia.'
        time.sleep(1)
        print 'Y con una cerradura muy resistente.'
    else:
        print 'Para lo que te va a servir...'
        return
# Mostrar el inventario
if verbo == 'inventario':
    # Verificar que llevas algo
    if inventario == []:
        print 'No llevas nada'
        return
    # Listar tus objetos
    print 'Llevas contigo ',
    for i in inventario:
        print i
    print
    return
# Atacar al troll
if verbo == 'atacar':
    # Asegurarse que se ataca al troll
    if len(palabras) == 1 or palabras[1] == 'troll':
        # El troll tiene que estar vivo para atacarle
        if not trollVivo:
            print 'Pero... ¡si ya está muerto!'
            return
        if (piso, fila, columna) == (2, 1, 2):
            # Atacar al troll y ponerlo activo
            trollActivo = True
            # Mirar si llevamos la espada
            if tieneEspada:
                print 'El troll acerca su rostro fétido al tuyo...'
                time.sleep(1)
                if haComido:
                    # Ejecutar al troll
                    print 'Y lo decapitas de manera fulminante con tu espada!'
                    trollVivo = False
                else:
                    # Morir por no estar fuerte
                    print ' mientras te sientes débil por no haber comido...'
                    time.sleep(0.5)
                    print 'y te desmayas mientras el troll te desmiembra.'
                    juegoAcabado = True
                    trollActivo = False
            else:
                print 'Insensato... ¡Deberías haber huído mientras podías!'
        else:
            print 'Creo que no estás en el lugar correcto para hacer eso.'
    else:
        print 'Supongo que estás de broma, ¿no?'
    return

# Función que gestiona las felicitaciones al ganar el juego
def ganar():
    print '*****'
    print '¡El resplandor de las joyas te deslumbra!'
    print '***** ENHORABUENA, CONSEGUISTE *****'
    print '***** LA PIEDRA PRECIOSA *****'
    print '*****'

# Bucle general, necesario por si se quiere volver a jugar
while True:
    # Primero inicializamos las variables del juego

    # Posición
    piso = 1

```

```

fila = 1
columna = 2

# Indicadores del estado del jugador
bAbierto = False
sinLlave = True
tieneEspada = False
haComido = False
haBebido = False
tieneEspada = False
espadaEnMano = False
pozoAlSur = False
inventario = []

# Mapa de situación de diferentes objetos
mapa = {}
mapa[(1,1,1)] = []
mapa[(1,1,2)] = []
mapa[(1,1,3)] = []
mapa[(1,2,1)] = ["galletas"]
mapa[(1,2,2)] = []
mapa[(1,2,3)] = ["copa"]
mapa[(2,1,1)] = ["vaso"]
mapa[(2,1,2)] = []
mapa[(2,1,3)] = []
mapa[(2,2,1)] = []
mapa[(2,2,2)] = []
mapa[(2,2,3)] = ["bombones"]

# Indicador de partida terminada
juegoAcabado = False

# Indicador para no repetir varias veces la descripción
yaDescrito = False

# Contador del tiempo que pasa
tiempo = 0

# Tiempo de espera del troll
trollVivo = True
esperaTroll = 0

# Indicador de que el troll te ataca
trollActivo = False

# Mostrar la introducción
intro()

# Bucle de juego. Se repite una y otra vez mientras dura el juego
while True:
    # Describir dónde está el jugador, si hace falta
    if not yaDescrito:
        describir(piso, fila, columna)
        yaDescrito = True
    # Pedir al jugador que realice una acción
    orden = raw_input('¿Qué quieres hacer? ').lower()
    # Aumentar el tiempo
    tiempo = tiempo + 1
    # Si estás con el troll, aumentar el tiempo de espera
    if (piso,fila,columna) == (2,1,2):
        esperaTroll = esperaTroll + 1
    # Procesar la acción y ejecutarla
    procesar(orden)
    # Si no se ha bebido pasado un tiempo, se pierde si ya no se ha hecho
    if tiempo>25 and not haBebido and not juegoAcabado:
        print 'Estás sediento, no puedes más, deberías haber bebido...'
        time.sleep(1)
        print 'Te desmayas...'
        time.sleep(1)
        if trollVivo:

```

```

        print '... Y un troll aprovecha la situación.'
        print 'El juego ha acabado.'
    else:
        print '... Y te consumes poco a poco en el suelo, deshidratado.'
        juegoAcabado = True
    # Si has pasado demasiado tiempo en presencia del troll
    # o si está activo, te ataca.
    # Pero siempre que no lo haya hecho ya y estés muerto
    if (esperaTroll > 2 or trollActivo) and trollVivo:
        print 'El troll te ataca...'
        time.sleep(1)
        # Si el jugador está armado, dar una oportunidad
        if tieneEspada:
            trollActivo = True
            if not espadaEnMano:
                print '... y desenvainas la espada y ¡te dispones a luchar!'
                espadaEnMano = True
        else:
            print 'Desarmado, no tienes ninguna oportunidad.'
            time.sleep(0.5)
            print 'Te despedaza y se come tus entrañas con avidez.'
            juegoAcabado = True
    # Si el troll ha atacado varias veces y no has comido, mueres
    if esperaTroll > 3 and trollVivo:
        if not haComido:
            print 'Débil por falta de comida, no ofreces resistencia al troll.'
            print '¡La comida eres tú!'
            juegoAcabado = True
    # Y si ha pasado demasiado tiempo, también pierdes
    if esperaTroll > 5 and trollVivo:
        print 'Finalmente el troll te hunde el pecho y mueres brutalmente...'
        juegoAcabado = True
    # Si el juego ha terminado, salir del bucle
    if juegoAcabado:
        break

print
print '-----'
continuar = raw_input('¿Quieres jugar otra vez? ').lower().startswith('s')
if not continuar:
    break
print '-----'
print

```

¡Es tu turno! **Crea un documento de Google Docs**, compártelo con tu profesor y en él responde a lo siguiente:

1. En primer lugar fíjate y explica el código de cada una de las funciones definidas en el programa.
2. Explica también el código que aparece dentro del while del juego.
3. En diferentes puntos del juego hay código mejorable o corregible. Indica los bugs o las mejoras que se te ocurran.