

TUTORIALES Y BUENAS PRÁCTICAS - PYTHON

TUTORIAL DEL EJERCICIO I – IDLE



TUTORIALES Y BUENAS PRÁCTICAS - PYTHON

TUTORIAL DEL EJERCICIO I – IDLE

IDE IDLE

IDLE es el IDE (entorno de desarrollo) desarrollado por la fundación Python y se ha incluido con la implementación predeterminada del lenguaje desde hace mucho tiempo. A continuación, utilizaremos el mismo para la realización de los tutoriales.

Enunciado

A partir de las distintas problemáticas que surgen de la basura y la contaminación y que condicionan nuestro derecho a un ambiente sano, deben realizar un programa que permita al usuario reconocer esas problemáticas creando conciencia ecológica en cada acción que realice.

Contenidos:

- tipos de variables
- manejo de input output
- manejo de gráficos e imágenes
- uso de librerías
- sentencias de decisión
- bucles

ACTIVIDAD 1

Arte ASCII

Para concientizar sobre la importancia de mantener limpia la ciudad, van a realizar carteles y señales sobre el cuidado del medio ambiente.

Realicen un dibujo mediante combinaciones de impresiones de código ASCII. El objetivo de la actividad es:

- Imprimir en pantalla.
- Conocer acerca de los códigos ASCII y su impresión en pantalla.
- Documentar Software mediante comentarios.

¿Vieron alguna vez esos dibujos hechos con símbolos?



¿Reconocen los caracteres utilizados? Todos estos caracteres están incluidos en el código ASCII. De eso se trata el “Arte ASCII”. La idea es realizar un dibujo usando símbolos y caracteres ingeniosamente ubicados durante la impresión en pantalla.

En Python 3.11 cada línea que se muestra por pantalla se inicia con la instrucción “print”. Lo que quiero visualizar, un carácter o una cadena de caracteres o string, deberá ir encerrado entre paréntesis y dos apostrofes “ ”. Así, manejando adecuadamente los espacios puedo conformar un rectángulo imprimiendo caracteres “+”, “-” y “|”. Por ejemplo:

En las siguientes líneas muestro un cartel en la pantalla

```
print("+-----+")
```

```
print("| PROTEJAMOS AL MEDIO AMBIENTE |")
```

```
print("+-----+")
```

Como parte de las buenas prácticas es recomendable documentar el software (para que otros o cada uno de nosotros en el futuro podamos entenderlo), una forma simple consiste en “comentar” el programa. En Python, el comentario de una línea se logra anteponiendo el símbolo “#” y el programa no ejecutará el código comentado.

Desarrollo de enunciado actividad 1:

- Abran un IDLE: Llámenlo “arte_ascii.py”.
- Típeen el siguiente código, grábenlo y ejecútenlo.

Código de arte_ascii.py:

```
# Este programa imprime en pantalla arte ascii
```

```
print("Una Persona:")
```

```
print(" \ O /")
```

```
print("  [ ]")
```

```
print("  /\ ")
```

```
print("")
```

```
print("Un Perro:")
```

```
print(" Q  /")
```

```
print(" #####")
```

```
print(" || ||")
```

```
# Confección de texto compuesto

print("")

print("Texto compuesto por caracteres ASCII")

print("+-----+")

print("| ### ### ## #  ## ## # ## |")

print("| # # # # # # # # # # |")

print("| # # # # # # # # # |")

print("| #### # # # # # # # # |")

print("| # # # # # # # # # |")

print("| # # # # # # # # # |")

print("| #### ### ## ## ## ## # # |")

print("+-----+")
```

Guarden el trabajo y ejecútenlo.

Salida de pantalla

```
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Sebas\Desktop\CFuente Tutoriales\arte_ascii.py =====
Una Persona:
  \ O /
  [ ]
  / \

Un Perro:
Q    /
####
|| ||

Texto compuesto por caracteres ASCII
+-----+
| ### ### ## #  ## ## # ## |
| # # # # # # # # # # |
| # # # # # # # # # # |
| #### # # # # # # # # |
| # # # # # # # # # # |
| # # # # # # # # # # |
| ### ### ## ## ## ## # # |
+-----+
>>>
```

ACTIVIDAD 2:

Versión 220524

4

Para realizar campañas sobre el cuidado del medio ambiente, hay que considerar la edad de la población a la que van dirigidas.

Realicen un programa que les pregunte nombre y año de nacimiento. Luego imprimir en pantalla un saludo con el nombre y la edad que cumplen este año. El objetivo de la actividad es:

- Usar Librerías.
- Usar la librería “time” para manejo de fechas.
- Ingresar datos por teclado.
- Armar una cadena con diferentes datos.

Introducción a Librerías:

Las librerías son herramientas de software hechas por desarrolladores que permiten realizar acciones específicas. En Python se las llama con la sentencia “import”. Sin embargo, para definir qué parte de la librería se desea es común usar: “from (nombre) import (lo que necesito)”, cuando necesito usar varios componentes de la librería uso el carácter asterisco.

```
from time import *
```

En este ejemplo, se importará la librería “time” completa.

Uso de la librería “time”:

Esta librería se especifica en el manejo de tiempos y fechas.

```
from time import *
```

```
nombre=input('¿Cómo te llamás?')
```

```
nacimiento=input('En qué año naciste?')
```

```
ahora=int(strftime('%Y', gmtime()))
```

```
edad=int(ahora)-int(nacimiento)
```

```
print(edad)
```

En esta porción de código el programa invita a que ingresen el año de nacimiento. La variable “ahora” es la fecha actual en formato año (%Y). Al poner “int(algo)” a ese “algo” lo convierte a número entero, para operar y la edad obtenida la resta entre el año actual y el año de nacimiento.

Ingreso de datos por teclado:

Mediante la función “input” puede solicitarse al usuario que ingrese datos por teclado. La sintaxis es: `variable=input('pregunta')`. Aparecerá en pantalla la pregunta, y el valor que el usuario ingrese por teclado hasta presionar “enter”, luego se almacenará en la variable elegida.

```
nombre=input('Ingresa tu nombre')
```

Armado de cadenas:

Muchas veces se requiere armar una oración formada por diferentes textos, algunos, variables. En Python los textos se transforman en cadenas encerrándolos con comillas simples o dobles. Si se quiere concatenar ese texto con el contenido de una variable, se usa “+” y “str(variable)”

```
cadena='Hola, Mundo! Soy '+str(nombre)+' y este año cumplo '+str(edad)+' años.'
```

Desarrollo del enunciado actividad 2:

- Abran un nuevo IDLE. Llámenlo “saludo.py”
- Tipeen el siguiente código, guárdenlo y ejecútenlo.

Código de saludo.py:

```
from time import *

nombre=input('¿Cómo te llamas?')

nacimiento=input('En qué año naciste?')

ahora=int(strftime('%Y', gmtime()))

edad=int(ahora)-int(nacimiento)

cadena='Hola, Mundo! Mi nombre es '+str(nombre)+' y este año cumplo '+str(edad)+' años.'

print(cadena)
```

Salida de pantalla:

```
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Sebas\Desktop\CFuente Tutoriales\saldudo.py =====
¿Cómo te llamas?Juan
En qué año naciste?2001
Hola, Mundo! Mi nombre es Juan y este año cumplo 22 años.
>>>
```

ACTIVIDAD 3:

Con el objetivo de crear conciencia sobre el cuidado de las playas y mares, dado que también son contaminados por los desechos que se arrojan, trabajarán con un fondo marino.

Programen un juego que contenga una pantalla con un fondo obtenido a partir de una imagen y sobre él presenten un objeto que responderá a las siguientes instrucciones: adelantar, atrasar, girar a la derecha y girar a la izquierda, por cada presión de las teclas de flechas: arriba, abajo, derecha e izquierda. El objetivo de la actividad es:

- Usar la librería “turtle”.
- Definir y usar funciones.
- Usar imágenes y el teclado para animar objetos.

La librería “turtle”:

La librería “Turtle” está basada en el lenguaje Logo desarrollado por Wally Feurzig and Seymour Papert para la enseñanza de programación en niños “. Dicho lenguaje se desarrolló como librería y hoy está disponible en Python. Permite dibujar y crear animaciones de modo sencillo. Algunos métodos y/o atributos son:

- color()
- shape(forma)
- goto(x,y)
- pendown()

Recuerden importar la librería Turtle desde el símbolo del sistema mediante el comando `pip install Turtle`.

Un ejemplo de uso:

```
import turtle
```

```
tortuga=turtle.Turtle() # crea un objeto “tortuga”
```

```
tortuga.shape("turtle") # forma de tortuga
```

```
tortuga.color("green") # tendrá color verde
```

```
tortuga.penup()
```

```
tortuga.goto(100,200) # se moverá a la posición (100,200) sin dejar huella
```

Funciones:

Una función es un conjunto de código que se ejecuta solo cuando es llamado. Puede recibir parámetros o no. El uso de funciones evita la repetición de código. Se recomienda consultar el manual de buenas prácticas de programación para mejorar su uso.

A continuación, se presenta un ejemplo de definición de una función con su correcta sintaxis y un llamado a dicha función:


```
def realizar_calculo (parametro):  # inicio de la definición de la función

    # "realizar_calculo".

    x=parametro**2                # eleva parametro al cuadrado

    y=2*parametro+1

    punto=(x,y)

    return punto                  # fin de la definición de la función

    # "realizar_calculo".

    nueva_posicion=realizar_calculo(5)    # devolverá (25,11)

    print(nueva_posicion)
```

Notar que para que pueda ejecutarse un llamado a la función, ésta deberá haberse definido previamente (antes del llamado).

Imágenes y animación de objetos usando la librería "turtle":

"Turtle" posee métodos y funciones que permiten que pueda programarse muy fácilmente la animación de un objeto.

Una animación básica consiste en un objeto que puede moverse automáticamente o controlado por botones sobre un fondo. Tanto el fondo como el objeto pueden lograrse a partir de archivos de imágenes importados.

Para definir el fondo:

```
import turtle

import os

screen = turtle.Screen()

# Imágenes

# Tamaño de la pantalla

screen.setup(500, 400)

# Imagen del fondo

archivo_fondo=str(os.getcwd())+"/imagenes/bajo-el-mar.png"

screen.bgpic(archivo_fondo)

# El objeto a animar: Una tortuga verde

turtle.shape('turtle')
```



```
turtle.color('green')
```

Para mover el objeto por teclado, se definen como funciones los movimientos:

- mover hacia adelante
- mover hacia atrás
- rotar hacia la derecha
- rotar hacia la izquierda

```
# Seteo del movimiento
```

```
velocidad_de_movimiento = 10
```

```
velocidad_de_giro = 10
```

```
# Funciones de movimiento
```

```
def adelantar():
```

```
    turtle.forward(velocidad_de_movimiento)
```

```
def atrasar():
```

```
    turtle.backward(velocidad_de_movimiento)
```

```
def girar_a_la_izquierda():
```

```
    turtle.left(velocidad_de_giro)
```

```
def girar_a_la_derecha():
```

```
    turtle.right(velocidad_de_giro)
```

Luego, a cada tecla se le asocia una función de movimiento. Observar que “screen.onkey” recibe dos parámetros: la función de movimiento, y la tecla asignada.

```
screen.onkey(adelantar, "Up")
```

```
screen.onkey(atrasar, "Down")
```

```
screen.onkey(girar_a_la_izquierda, "Left")
```

```
screen.onkey(girar_a_la_derecha, "Right")
```

```
screen.listen()
```

Desarrollo del enunciado actividad 3:

- Abran un nuevo IDLE.
- Tipeen el siguiente código. Llámenlo ‘bajo-el-mar.py’ y ejecútenlo.

Código de 'Bajo-el-mar.py':

```
import turtle

import os

screen = turtle.Screen()

# Imágenes

# Tamaño de la pantalla

screen.setup(500, 400)

# Imagen del fondo

archivo_fondo=str(os.getcwd())+"/imagenes/bajo-el-mar.png"

screen.bgpic(archivo_fondo)

# El objeto a animar: Una tortuga verde

turtle.shape('turtle')

turtle.color('green')

# Seteo del movimiento

velocidad_de_movimiento = 10

velocidad_de_giro = 10

# Funciones de movimiento

def adelantar():

    turtle.forward(velocidad_de_movimiento)

def atrasar():

    turtle.backward(velocidad_de_movimiento)

def girar_a_la_izquierda():

    turtle.left(velocidad_de_giro)

def girar_a_la_derecha():

    turtle.right(velocidad_de_giro)

turtle.penup()

turtle.speed(0)

turtle.home()
```

Asociar las funciones de movimiento con presión de teclas

```
screen.onkey(adelantar, "Up")
```

```
screen.onkey(atrasar, "Down")
```

```
screen.onkey(girar_a_la_izquierda, "Left")
```

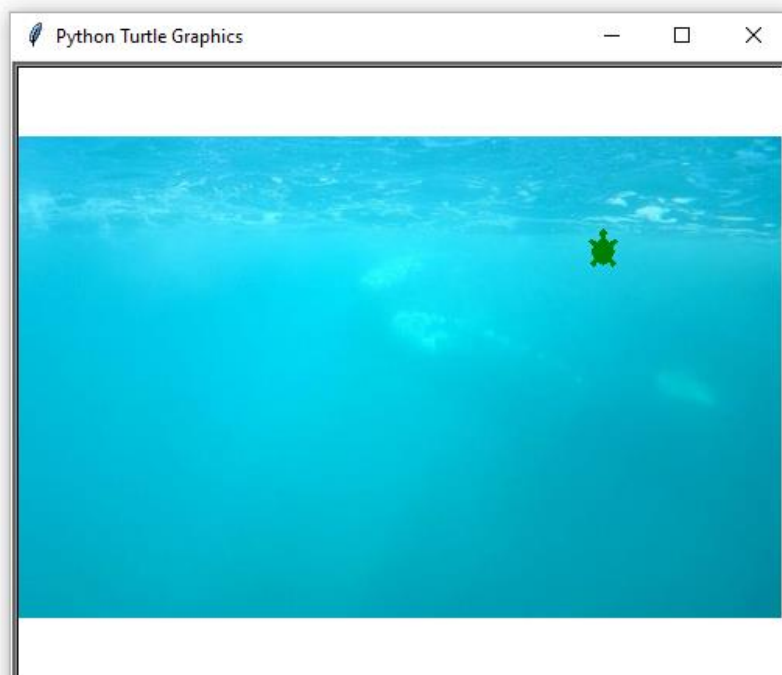
```
screen.onkey(girar_a_la_derecha, "Right")
```

```
screen.listen()
```

```
turtle.done()
```

Salida de Pantalla:

```
>>> Type "help", "copyright", "credits" or "license()" for more information.
===== RESTART: C:\Users\Sebas\Desktop\CFuente Tutoriales\bajo-el-mar.py =====
```



ACTIVIDAD 4:

A partir de la utilización de formas y colores, tendrán que armar el mensaje "Salvemos el planeta".

Creen una pantalla de 400x400 píxeles, asignenle un color, escriban un texto en otro color, dibujen un círculo y un rectángulo, todo en diferentes colores y ubicaciones. El objetivo de la actividad es:

- Usar colores almacenados, en hexa o en formato RGB.
- Usar colores en textos o en formas estándar.

- Dibujar formas

Los colores: representación

La función “color” de la librería “turtle” pintará a los posteriores objetos que se generen con el color que se presenta como argumento. Hay dos formas muy usadas para la representación de colores:

- Por su nombre: varios colores ya están definidos por su nombre en inglés. Algunos de ellos son: red, blue, green, yellow, black, white, y más.
- Por sus componentes RGB: Un color se puede representar como una terna donde el primer componente corresponde a la cantidad de rojo (R), el segundo componente corresponde a la cantidad de verde (G) y el tercer a la cantidad de azul (B). Cada cantidad se corresponde con un número entre 0 y 255. Por ejemplo: color(250,15,64). Algunos ejemplos: rojo=color(255,0,0), amarillo=color(255,255,0), verde=color(0,255,0), azul=color(0,0,255), blanco=color(255,255,255), negro=color(0,0,0).

Animación y movimiento del cursor mediante la librería “turtle”

La librería “turtle” brinda facilidades para el movimiento del cursor útiles también para dibujar. Algunas se presentan a continuación:

- forward(x): se mueve “x” píxeles hacia adelante.
- Backward(x): se mueve “x” píxeles hacia atrás.
- right(y): rota “y” grados hacia la derecha.
- left(y): rota “y” grados hacia la izquierda.
- Penup(): levanta el lápiz y deja de escribir.
- pendown(): baja el lápiz y empieza a escribir.

El siguiente ejemplo muestra un código que dibuja un triángulo equilátero de 100 píxeles de lado, con los lados dibujados en rojo.

```
from turtle import *  
  
color('red')  
  
pendown()  
  
forward(100)
```

```
right(60)
```

```
forward(100)
```

```
right(60)
```

```
forward(100)
```

```
right(60)
```

```
hideturtle()
```

Desarrollo del enunciado actividad 4

Abran un IDLE. Tipeen el código que figura abajo. Llámenlo 'colores.py'.

El código 'colores.py':

```
from turtle import *
```

```
# La pantalla
```

```
screen=Screen()
```

```
screen.colormode(255)
```

```
screen.setup(400,400)
```

```
# Formamos color de fondo en RGB
```

```
rojo=50 # cantidad de Rojo (entre 0 y 255)
```

```
verde=255 # cantidad de Verde (entre 0 y 255)
```

```
azul=50 # cantidad de azul (entre 0 y 255)
```

```
screen.bgcolor(rojo,verde,azul)
```

```
# El texto
```

```
rojo=255
```

```
verde=255
```

```
azul=255
```

```
color(rojo,verde,azul)
```

```
style=('Arial',13,'bold')
```


```
write('SALVEMOS EL PLANETA',font=style,align='center')
```

```
# El círculo:
```

```
penup()
goto(10,-100)
rojo=10 # cantidad de Rojo (entre 0 y 255)
verde=10 # cantidad de Verde (entre 0 y 255)
azul=255 # cantidad de azul (entre 0 y 255)
color(rojo,verde,azul)
dot(100)
goto(-140,60)
color('yellow')
pendown()
forward(280)
right(90)
forward(100)
right(90)
forward(280)
right(90)
forward(100)
hideturtle()
```

Salida de Pantalla

```
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
>>>
```



Ln: 5 Col: 0