Universidad Luterana Salvadoreña

Facultad

Ciencias del Hombre y la Naturaleza

CARRERA

Licenciatura en Ciencias de la Computación

ASIGNATURA

Algoritmo

DOCENTE

Licenciado David Isaac Clímaco Orellana

TITULO DEL TRABAJO

Propuesta didáctica para la aprender el concepto y uso del módulo turtle de Python

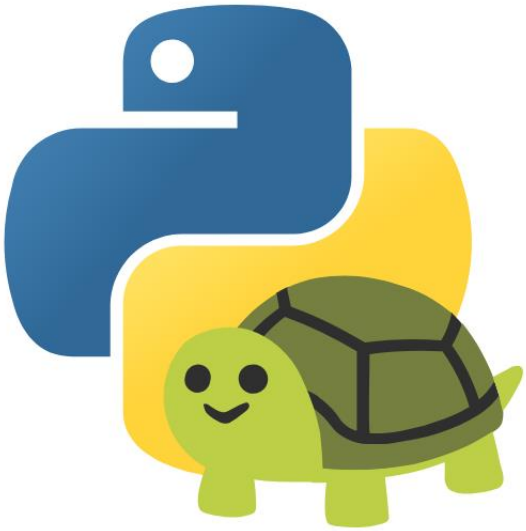
NOMBRE DE LOS ESTUDIANTES

Javier Enrique López

Carlos Javier Rivera Flores

José David Martínez Chávez

William Antonio Romero Alvarado



Propuesta didáctica para aprender el concepto y uso del módulo tortuga de Python

Contenido

Introducción.

1 Breve introducción al módulo turtle de Python

Acerca de Python 6

Instrucciones de instalación 6

Ambiente de desarrollo integrado IDLE Python 7

Geometría de turtle 8

Ejemplo 1: construcción de un cuadrado 9

Ejemplos basados en constelaciones 10

Python en la nube 11

Recomendaciones 16

Conclusiones 17

Anexos ..... 17 - 28

Bibliografía 29

Introducción

Esta propuesta didáctica se ha elaborado como una herramienta para la enseñanza del concepto del módulo turtle de Python. Está dirigida a estudiantes principiantes del tema, ya que se plantean ejercicios y actividades que nos guiaran en el uso básico de esta herramienta.

Con el propósito de estudiar las propiedades de turtle desde una perspectiva, intuitiva y experimental, se ha organizado el escrito de la siguiente forma: breve introducción a Python, su IDLE, luego vamos a ver un poco sobre la geometría del módulo turtle, veremos como de una forma simple pero a veces extensa o de una forma corta pero requiriendo de un poco más de comprensión sobre las funciones que Python nos ofrece, podemos crear desde simples imágenes hasta animaciones complejas mediante el uso de conceptos esenciales de programación en Python, usando ejemplos sencillos para llegar naturalmente a la construcción de figuras y de ese modo, que los interesados puedan desarrollar soluciones a problemas reales a través de la elaboración de programas computacionales

Objetivos

- Conocer sobre Python IDLE, como instalarlo y requerimientos básicos para empezar a trabajar con el módulo turtle.

- Enseñar el proceso secuencial que se debe hacer para usar el módulo Turtle en Python, comandos básicos y algunos ejemplos de aplicación de este módulo.

Breve introducción al módulo turtle de Python

Acerca de Python

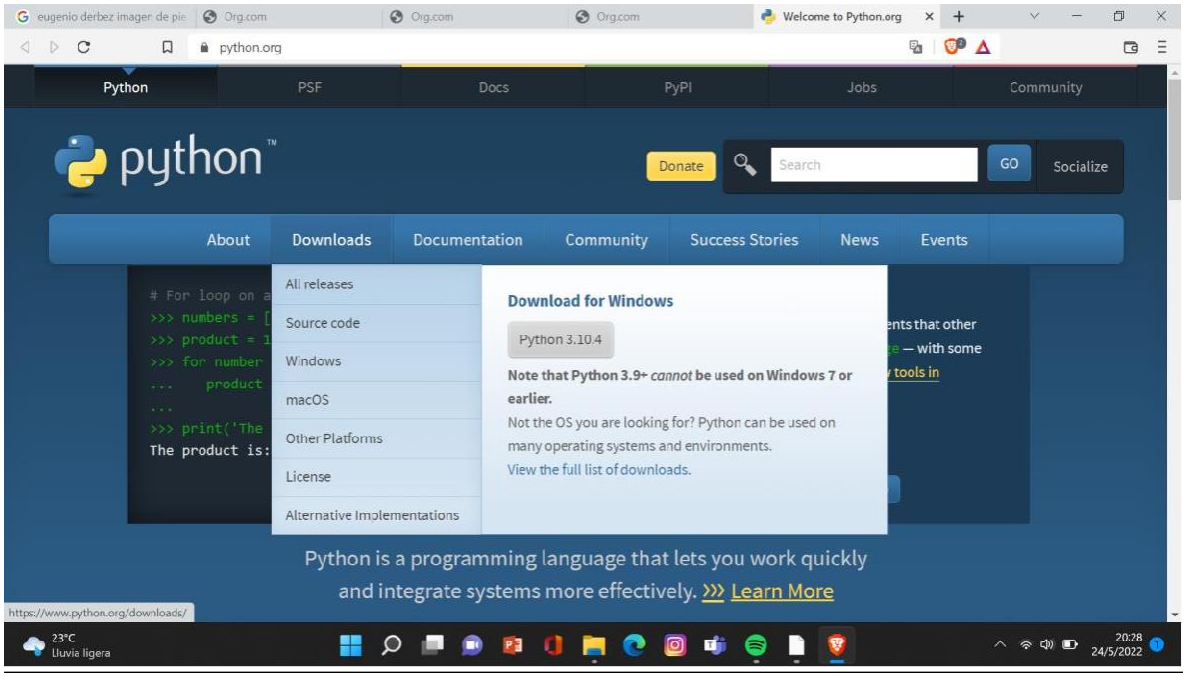
Python es un lenguaje de programación interpretado y multiplataforma; es decir que el código de un programa se ejecuta a través de un intérprete, funciona en casi cualquier sistema operativo, está orientado a objetos y su tipado es dinámico.

Fue creado a principios de los noventa por el investigador holandés Guido Van Rossum, quien motivado por desarrollar el sistema operativo conocido como **amoeba**, opto por crear un lenguaje con menos limitaciones y problemas que el ABC, lenguaje que usualmente utilizaba el Centrum Wiskunde & Informatica

Este programa se distribuye actualmente con licencia GNU (General public license) significa los usuarios finales de Python tienen la libertad de usar, estudiar, compartir (copiar) y modificar el software.

Instrucciones de instalación

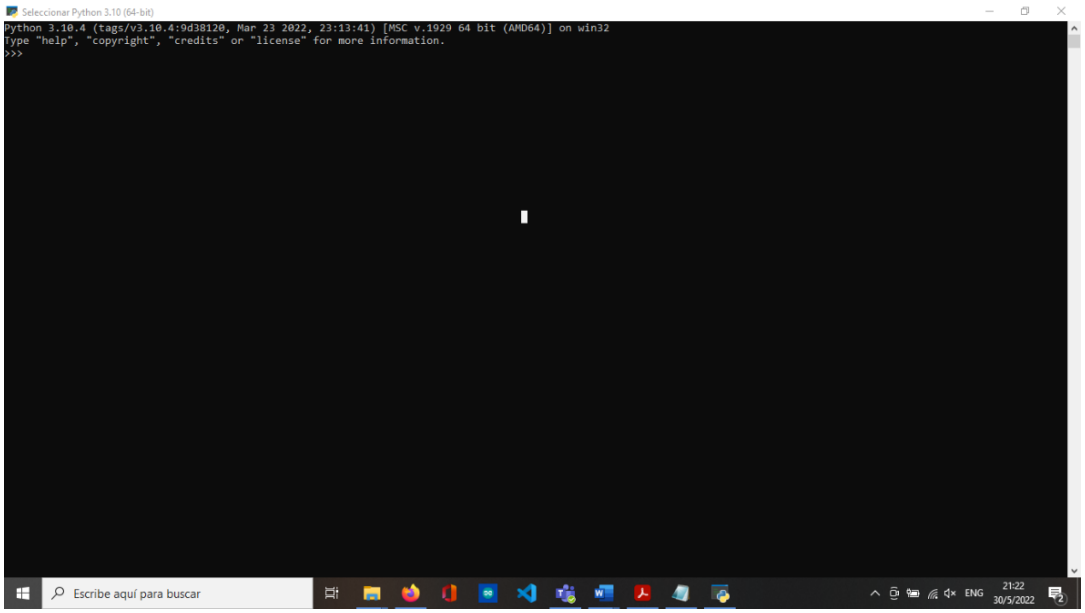
- Ingresar a la página oficial de Python: https://www.python.org/
- Acceder a la pestaña “Downloads" donde se presenta una lista con las diferentes versiones y sistemas operativos disponibles.
- Descargar el instalador más reciente; actualmente es Python 3.4.1 (la descarga no tiene costo)
- Importante recordar que en las versiones inferiores a la 3.0, deberemos de instalar TK (TKINTER) de forma manual para poder usar el módulo turtle.
- Instalar y abrir mediante la ruta de acceso en Windows: Inicio, todos los programas, Python 3.4.1, IDLE (Python GUI).



Ambiente de desarrollo integrado IDLE

IDLE es un editor predeterminado que ofrece funciones de búsqueda y depuración simbólica.

Al ingresar a él se muestra una ventana interactiva encabezada por la versión, fecha y plataforma.



Después de ingresar al ambiente IDLE, conviene abrir un nuevo espacio de trabajo, seleccionando la opción “New file" en la pestaña File. Al terminar programa se oprime F5 para ejecutarlo, orden que se cumplirá siempre que se haya guardado el archivo con antelación.

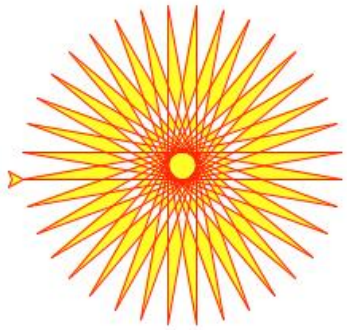
Geometría turtle

La geometría de la tortuga, fue desarrollada a finales de la década de los sesenta por el grupo Logo de investigación de inteligencia artificial del MIT; como un conjunto de procedimientos que le permitían al usuario controlar a un robot virtual con forma de tortuga que podía dejar o no, un trazo sobre los lugares por los cuales se movía.

Imagina una tortuga robot que siga las siguientes coordenadas:

```
from turtle import *
color('red', 'yellow')
begin_fill()
while True:
    forward(200)
    left(170)
    if abs(pos()) < 1:
        break
end_fill()
done()
```

Obtendremos la siguiente imagen:



Al combinar estos comandos y otros similares, se pueden dibujar figuras intrincadas y formas, por ejemplo, imagine que desea mover la tortuga de un punto cualquiera a otro.

En primer lugar, necesitara un punto de partida o de referencia tal como el origen de un sistema cartesiano. Para llegar al punto deseado se podrá indicar el número de unidades (rigurosamente hablando, se trata de pixeles) que debe desplazarse en una dirección determinada por un ángulo.

En el módulo turtle ya están predefinas este tipo de instrucciones y se establece como posición inicial aquella en la que la cabeza de la tortuga apunta hacia el este o la derecha. Si lo que se quiere es mover la tortuga 100 unidades hacia la dirección que apunta su cabeza, utilizando la instrucción forward () o abreviadamente fd (); se puede escribir:

forward (100)

Sin embargo, se debe recordar que en el lenguaje de programación Python*, es necesario llamar al módulo turtle como se muestra a continuación:

from turtle import*
forward (100)

Por otra parte, si es necesario introducir un ángulo para cambiar la orientación actual de la tortuga en términos de un ángulo en grados y en el sentido de las manecillas del reloj, se utiliza la instrucción right() o rt(). Las instrucciones reciprocas a forward y right son respectivamente: back()- bk() y left() - lt() es decir, moverse en la dirección opuesta a la que apunta la cabeza de la tortuga y rotar en el sentido opuesto a las manecillas del reloj.

Ejemplo 1: construcción de un cuadrado

A continuación, se muestran diferentes métodos para construir un cuadrado con lados paralelos a los ejes, de 80 unidades de lado:

```
from turtle import*
fd (80)
rt (90)
fd (80)
rt (90)
fd (80)
rt (90)
fd (80)
```

Figura 1: Cuadrado en turtle

Con este método se traza un cuadrado en el lienzo con la base en la dirección del eje x, si se cambia la instrucción rt() por lt(), se obtiene un cuadrado simétrico al anterior con respecto al mismo eje.

Cuando un método como el descrito presenta regularidades o instrucciones que se repiten ordenadamente, se puede recurrir a la iteración o a una función para resumir el programa, en otras palabras, se logra aumentar su eficiencia reduciendo el número de órdenes.

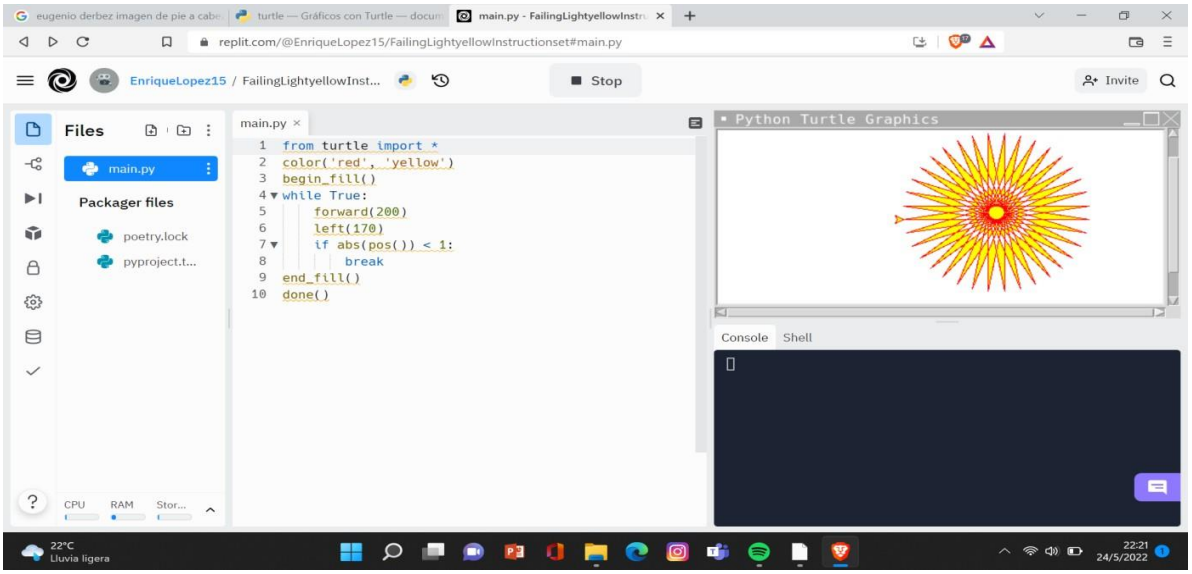
El código usando iteración con el comando **for** es el siguiente (se omitirá a partir de este punto el encabezado from turtle import*):

```
for i in range (1,5):
        fd (80)
        rt (90)
```

La sangría en la segunda y tercera línea se conoce como “indentado", es imprescindible y equivale a cuatro espacios horizontales o un espacio del tabulador. Otra opción consiste en escribir una función en términos de la longitud del lado:

```
def cuadrado1(x):
        i = 1
        while i <= 4:
                fd (x)
                rt (90)
                i = i+1
```

Otra forma de utilizar Python y el módulo turtle es haciéndolo desde Replit .



Repl.it es un entorno de desarrollo en online. Si estas iniciándote al mundo de la programación debes saber que existen diferentes herramientas online con las que se puede compilar y ejecutar código en decenas de lenguajes de programación desde el propio navegador. Los puntos fuertes de Replit son que no es necesario instalar nada basta con crear una cuenta con tu correo electrónico y el otro fuerte es que Replit es totalmente gratuito

Otra forma de usar Python en la nube es Google Colab

¿Qué es Google Colab?

Google Colab es un sitio en la web que permite escribir, ejecutar y compartir código dentro de Google Drive. Utiliza documentos notebook que se componen de células, cada una de las cuales puede contener código, texto, imágenes y demás.

Colab conecta tu notebook a un runtime basado en la nube, lo cual significa que puedes ejecutar código Python sin necesidad de instalación previa.

¿Qué podemos hacer en Google Colab?

- Escribir y ejecutar código en Python.
- Documentar códigos que admiten ecuaciones matemáticas.
- Crear y compartir notebooks.
- Importar y guardar archivos desde Google Drive.
- Importar y publicar notebooks desde Github.
- Usar la GPU de Google.

Ventajas

- No requiere configuración.
- Da acceso gratuito a GPUs.
- Permite compartir contenido fácilmente.
- Colab puede facilitar el trabajo, ya sea un estudiante, científico de datos o investigador de IA.
- Bibliotecas pre-instaladas.
- Guardado en la nube.
- Colaboración.

Desventajas

- No se ejecuta sin conexión.
- Conjuntos de datos que se importan al entorno sin ser montado desde Google Drive se perderán cuando la máquina virtual se apague.

¿Cómo utilizar Google Colab?

Podemos crear un notebook del Colaboratory desde nuestro propio Google Drive, solo damos click derecho sobre un espacio en blanco y agregamos un archivo de tipo Google Colaboratory.

Ejemplos basados en constelaciones utilizando el módulo turtle.

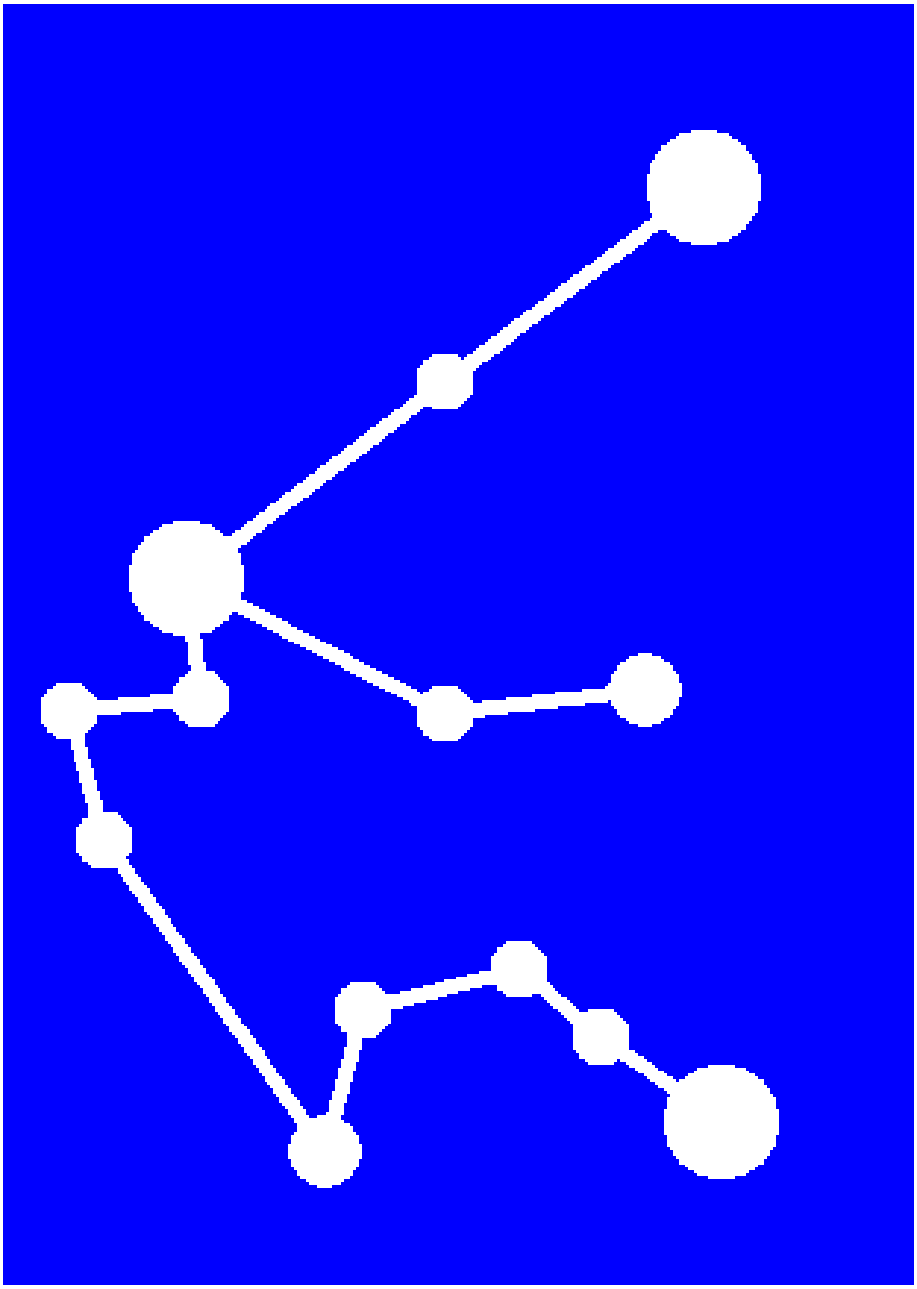
En el siguiente ejemplo podremos ver el código escrito de la manera básica, sin ningún tipo de función especial, lo cual lo hace más fácil de realizar, sin embargo, nuestro código será un poco más extenso dependiendo de la figura a realizar.

Acuario

```
from turtle import *

up()
goto(125,-125)
down()
dot(40)
left(145)
forward(50)
dot(20)
right(5)
forward(37)
dot(20)
left(55)
forward(55)
dot(20)
left(60)
forward(50)
dot(25)
right(130)
forward(130)
dot(20)
right(20)
forward(45)
dot(20)
right(100)
forward(45)
dot(20)
left(92)
forward(42)
dot(40)
right(60)
forward(110)
dot(20)
forward(110)
dot(40)
backward(220)
```

Obtendremos la forma:



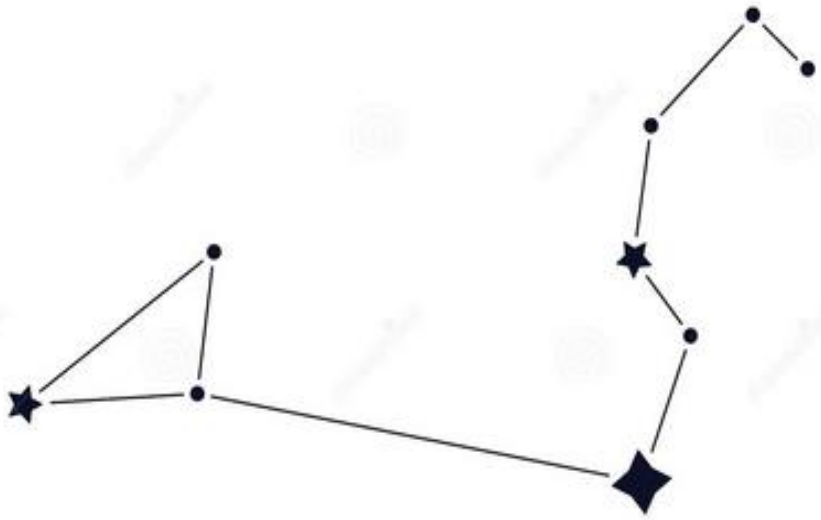
En el siguiente ejemplo podremos ver el código escrito usando las funciones que el módulo turtle nos ofrece, pudiendo crear un código más corto, eficiente, pero, que requiere un grado mayor descomprensión sobre las funciones que tenemos disponibles.

Leo

```
from turtle import *
from commands import *

c_moveup(275,150)
dot(20)
seth(180)
c_right(45,50,30)
c_left(80,100,20)
c_left(45,50,30)
pos = pos()
c_left(75,75,20)
c_right(45,85,50)
seth(180)
c_left(5,400,20)
c_left(10,100,50)
seth(90)
c_right(35,130,20)
goto(pos)
:
```

Obtendremos la forma.



Recomendaciones

- Al utilizar una versión de Python inferior a 3.0, deberemos de instalar TK(tkinter) manualmente para poder utilizar el módulo turtle.

- Para instalar Python 3.6 en un equipo con el sistema operativo Windows 7, 8, 8.1 o 10 utilizar una cuenta de usuario con privilegios de administrador, o bien, la propia cuenta del administrador local. Por seguridad, se puede agregar -temporalmente- para este proceso de instalación la cuenta del usuario actual al grupo local Administradores. Para ello, iniciar la aplicación de Administración de equipos, acceder a la herramienta del sistema Usuarios y grupos locales y agregar la cuenta actual a dicho grupo.

- En los sistemas con arquitectura 64 bit se puede instalar tanto la versión de Python para 32 bit como para 64 bit. En general, trabajar con la versión para 64 bit mejora el rendimiento y permite que un programa pueda utilizar más de 4 Gb de RAM, memoria límite en sistemas 32 bit. Normalmente, trabajando por debajo de este límite no se observan diferencias de rendimiento importantes entre ambas versiones, pero se recomienda instalar la versión de 64 bit. De todas formas, si se van a utilizar módulos de terceros, conviene comprobar si están disponibles para la versión a instalar

- Para poder usar el módulo de turtle, primero debemos importarlo en Python **from turtle import**

Conclusiones

•Conocimos más a profundidad sobre los orígenes de turtle.

•Aprendimos a diferenciar sobre lo que es un lenguaje de programación y lo que es un módulo en programación.

•Ahora podemos instalar Python e importar o instalar el módulo turtle.

•Aprendimos que se puede ejecutar código de turtle fuera del Idle de Python. Utilizando entornos en la web, como, por ejemplo: Google Colab,Replit,Jupyter Notebook ,etc.

•Aprendimos cuales son las principales funciones del módulo turtle. Es por ello que podemos ejecutar código de una manera básica creado por nosotros mismos.

Anexos

Comandos del módulo Turtle (Tortuga)

En las tablas adjuntas se relacionan los principales comandos del módulo Turtle y en forma concisa, la función que cumple cada uno de ellos:

Dibujo y movimiento

forward () |fd()	Mueve la tortuga en línea recta hacia donde apunta la tortuga tantas unidades como se indique.
back () bk()	Mueve la tortuga en línea recta hacia la dirección opuesta a la cabeza de la tortuga (o punta de flecha) tantas unidades como se indique.
right()|rt()	Gira a la tortuga a la derecha (en el sentido de las manecillas del reloj) tantas unidades angulares como se indique. Se puede establecer la medida en grados o radianes.
left()|lt()	Gira la tortuga en el sentido opuesto a las manecillas del reloj, tantas unidades angulares como se indique.
Pen up()|pu()|up()	Levanta el lápiz (no dibuja cuando se mueve).
down()	Ubica el lápiz sobre el “lienzo” para continuar con el dibujo.

speed ()	Determina la velocidad de la tortuga. Su parámetro es un entero en el rango de 0 a 10.
Goto() setpos() setposition()	Mueve la tortuga a una posición absoluta. Si el lápiz esta abajo, traza la línea, pero no cambia su orientación.
home()	Mueve la tortuga al origen; es decir, a las coordenadas (0,0).
setx()	Establece la coordenada en **x** manteniendo la coordenada en **y** invariable.
sety()	Establece la coordenada en **y** manteniendo la coordenada en **x** invariable.
setheading() seth()	Define la orientación de la tortuga según el ángulo, en el modo standard los valores predeterminados son 0=este,90=norte, 180 oeste y 270=sur.
circle()	Dibuja un círculo con un radio determinado. También dibuja arcos de circunferencia teniendo como parámetros el radio y el ángulo que por defecto este dado en grados.
dot()	Dibuja un círculo con un diámetro definido como un entero mayor o igual a 1.
stamp()	Estampa una copia de la forma de tortuga en el lienzo en la posición actual de la tortuga.
clearstamps()	Elimina la estampa de la tortuga en el lienzo.
undo()	Deshace la última acción (puede usarse repetidamente).

Conocer el estado de Turtle

towards()	Devuelve el angulo entre la linea desde la posicion de la tortuga hasta la posición definida por un vector u otra tortuga. Esto depende de la orientación inicial de la tortuga, en otras palabras del modo en el que se encuentre (standard, world o logo).
position() pos()	Devuelve la posición actual de la tortuga como un par ordenado.
xcor()	Devuelve la coordenada x de la tortuga.
ycor()	Devuelve la coordenada y de la tortuga.
heading()	Devuelve el valor del ángulo actual de la tortuga con respecto al eje x.
distance()	Devuelve la distancia desde la tortuga hasta las coordenadas (x,y), el vector o la tortuga dados; en “pasos de tortuga”.

Otros

help()	Despliega la ventana de ayuda del módulo que incluye un índice de temas. Su argumento puede ser un tema determinado del tipo string.

Sistema de medición angular

degrees()	Establece las unidades de medida angular en radianes. Equivale al comando degrees(2*math.pi).

radians()	Establece las unidades de medida angular en grados. Se puede establecer el sistema de medida con el argumento 360 o 400 grados.

Estado del lápiz

pendown() | pd() | down	Coloca el lápiz abajo, es decir sobre el lienzo. De esta forma dibuja cuando se mueve el lápiz.
penup() | pu() | up ()	Retira el lápiz del lienzo, entonces al moverse el lápiz no dibuja.
pensize() | width()	Define el ancho del trazo o devuelve el valor correspondiente. Su parámetro es un numero positivo.
pen()	Define o devuelve los atributos del lapiz (shown: True/False, pendown: True/False, pencolor : color- string or color-tuple, fillcolor : color-string or color- tuple, pensize: positive number, speed : number in range 0..10, resizemode: auto, user or noresize, stretchfactor : (positive number, positive number), outline: positive number, tilt : number.
isdown()	Responde True o False si el lápiz esta abajo o arriba respectivamente.

Control de color

color()	Devuelve o define el color del lápiz y del relleno.

pencolor()	Devuelve o define el color del lápiz. Su argumento se escribe de la forma “color”, tal como “red”, “yellow”... o como una tupla de la forma (r, g, b).
fillcolor()	Devuelve o define el color de relleno. Su argumento es tiene la misma forma del argumento del comando pen-color.
fill()	Devuelve o define el color del lápiz y del relleno.
begin_fill()	Se utiliza antes de dibujar una forma.
end_fill()	Colorea una forma después de ser dibujada.

Otros controles de dibujo

reset()	Borra lo dibujado en la pantalla hasta el momento y la tortuga se sitúa de nuevo en el origen.
clear()	Elimina de la pantalla los dibujos hechos por la tortuga. No mueve la tortuga. No se ven afectados el estado, la posición de la tortuga ni los dibujos de otras tortugas.
write()	Escribe el texto como la representación de una cadena tipo arg en la posición actual de la tortuga con la alineación y fuente determinadas.

Visibilidad de la tortuga

showturtle() | st()	Hace visible a la tortuga.
hideturtle() | ht()	Hace invisible a la tortuga. Es util para elaborar dibujos complejos, ademas de acelerar el dibujo observable.

invisible()	Regresa los valores verdadero o falso, dependiendo de si la tortuga es visible o no, respectivamente.

Apariencia de la tortuga

shape()	Establece la forma de la tortuga: “flecha”, “tortuga”, “cırculo”, “cuadrado”, “triangulo”, “clasico”.
resizemode()	Redimensiona la tortuga a uno de los valores: Auto: Adapta la apariencia de la tortuga al valor de pensize. Usuario: adapta la apariencia de la tortuga de acuerdo a los valores del factor de extension (stretchFactor) y esquema (outlinewidth), que son fijados mediante el comando shapesize () Noresize: No se altera la apariencia de la tortuga.
shapesize() | turtlesize()	Define el tamaño de la forma de la tortuga.
settiltangle()	Gira la forma de la tortuga de manera que apunte en la dirección especificada por el ángulo, independiente-mente de su ángulo de inclinación actual. No cambia dirección del movimiento de la tortuga.
tiltangle()	Establece o devuelve la inclinación actual. Es decir que funciona como contrainstrucci´on del comando tilt.
tilt()	Gira o inclina la forma de la tortuga de acuerdo al ángulo indicado, sin cambiar la dirección del movimiento de la tortuga.

Ajustar a un evento

onclick() | onscreenclick	Permite definir una actividad o evento para la tortuga al dar click sobre ella. Por ejemplo, girar (Usando como parámetro turn) o cambiar su color.
onrelease()	Inhabilita la instrucción oneclick.
ondrag()	Establece un evento de movimiento para la tortuga.

Métodos especiales de la tortuga

begin_poly()	Inicia el trazado de un polígono tomando como primer vértice el origen.
end_poly()	Traza el último segmento del polígono.
get_poly()	Devuelve el ultimo polígono registrado.
clone()	Crea y devuelve un clon de la tortuga con la misma posición encabezado y propiedades de la tortuga.
getturtle() getpen()	Devuelve las propiedades de tortuga como objeto para ser usado como una función que devuelve una “tortuga anónima”.
getscreen()	Devuelve los atributos de la pantalla sobre la cual se está dibujando.
setundobuffer()	Su parámetro es el tamaño (size). Establece las propiedades de control de la acción deshacer.
undobufferentries()	Devuelve el número de entradas en el control de la acción deshacer.
window_width()	Devuelve el ancho de la ventana del módulo turtle.
window_height()	Devuelve la altura de la ventana del módulo turtle.

Control de ventana

bgcolor()	Define el color de fondo de la ventana de la pantalla del módulo. Su argumento es un color (tipo string)o tres números en el rango 0... o una 3-tupla de tales números.
bgpic()	Coloca una imagen de fondo o devuelve el nombre de la actual. Se puede usar la ubicación de la imagen como (picname=ubicación) o si picname se define como“nopic” borra la imagen de fondo.
clear() | clearscreen()	Borra todos los dibujos y todas las tortugas de la pantalla. Resetea la pantalla a sus ajustes originales.
reset() | resetscreen()	Resetea todas las tortugas en la pantalla a su posición inicial.
screensize()	Redimensiona el lienzo con los argumentos que definen el ancho y alto del lienzo en pixeles canvwidth=entero positivo, canvheight=Entero positivo y bg=color tipo string.
setworldcoordinates()	Modifica el sistema de coordenadas al igual que los dibujos que aparecen en pantalla.
mode()	Establece el sistema coordenado como: standard que es usado por defecto (la punta de la flecha apunta hacia la derecha y los ángulos positivos van en contra del movimiento de las manecillas del reloj); logo (la punta de flecha apunta hacia arriba y los ángulos positivos van en el sentido de las manecillas del reloj) y world, que es el definido por el usuario.

colormode()	Fija el modo en 1.0 o 255. Posteriormente las tripletas de colores r, g, b se definirán en el rango definido por esos valores.
getcanvas()	Se utiliza normalmente en combinación con el módulo Tkinter para obtener un lienzo.
getshapes()	Devuelve una lista de las formas que actualmente se encuentran disponibles.
register_shape()|addshape()	Permite agregar una forma desde la ubicación indi-cada. Sus argumentos son: nombre de un archivo tipo gif y shape = None. Para registrar la forma: screen.register_shape(nombre y tupla con pares de coordenadas que definan el polıgono).
turtles()	Devuelve la lista de tortugas en pantalla.
window_height()	Devuelve la altura de la ventana del módulo.
window_width()	Devuelve el ancho de la ventana del módulo.

Control de animaciones

delay()	Coloca o devuelve el retraso del trazo en milisegundos (Este es aproximadamente el intervalo de tiempo entre 2 actualizaciones consecutivas del lienzo).
tracer()	Activa o inactiva la animación de la tortuga y retrasa la actualización de los gráficos. Puede ser usado para acelerar el trazado de graficas complejas. Su primer argumento es un entero positivo que define la actualización de la nesima pantalla, mientras que el segundo argumento (delay) asigna el valor de retraso.

update()	Actualiza la pantalla del módulo. Se usa cuando el comando tracer se encuentra en off.

Eventos en pantalla

listen()	Establece el orden de enlace entre eventos.
onkey()	Enlaza funciones a una clave de liberación. Si la función (fun) es none los eventos enlazados son removidos.
onclick() | onscreenclick()	Enlaza una función (fun) a la acción click en la pantalla. Parámetros: Una función que será llamada con las coordenadas del punto en el lienzo; número del botón del mouse (por defecto se asigna el numero 1 al izquierdo) y agregar (add) que genera nuevos enlaces que pueden reemplazar a los precedentes (usando True/False).
ontimer()	Instala un temporizador que llama a una función (fun) después de t milisegundos.

Métodos específicos para la pantalla

bye()	Cierra la ventana de gráficos del módulo turtle.
exitonclick()	Asigna el comando bye() a la acción click.
setup()	Establece el tamaño y posición de la ventana principal.
title()	Muestra el texto definido como parámetro, en la pantalla del módulo tortuga.

Referencias adicionales

if_stmt ::= "if"	The if statement is used for conditional execution
while_stmt ::= "while"	The while statement is used for repeated execution as long as an expression is true
for_stmt ::= "for"	The for statement is used to iterate over the elements of a sequence (such as a string, tuple or list) or other iterable object.
"elif" expression	"elif" sirve para enlazar varios "else if", sin tener que aumentar las tabulaciones en cada nueva comparación.

Bibliografía

Sitio oficial de Python

[Sitio en Internet]. Disponible en:

https://www.python.org/

Documentación oficial de Python sobre módulo turtle

[Sitio en Internet]. Disponible en:

https://docs.python.org/es/3.9/library/turtle.html

Trabajo de investigación sobre el módulo turtle por PROGRA USM (año 2016)

[Sitio en Internet]. Disponible en:

http://progra.usm.cl/apunte/tareas/2016-2/turtle.html

Entornos de lenguajes de programación en la web

[Sitio en Internet]. Disponible en:

https://colab.research.google.com

https://replit.com