	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:	M. I. Marco Antonio Martínez Quintana
Asignatura:	Estructura de Datos y Algoritmos
Grupo:	17
No de Práctica(s):	10
Integrante(s):	Díaz Segura, Mauricio Iván
No. de Equipo de cómputo empleado:	
No. de Lista o Brigada:	
Semestre:	2020-2
Fecha de entrega:	14 - IV - 2020
Observaciones:	

CALIFICACIÓN: _____

OBJETIVO

Aplicar las bases del lenguaje de programación Python en el ambiente de Jupyter notebook.

INTRODUCCIÓN

Python es un lenguaje de programación de alto nivel; debido a su potencia muchas empresas e institutos lo emplean actualmente.

Al igual que otros lenguajes de programación, Python posee distintos tipos de estructuras de control con el fin de analizar la información con la que se desee trabajar. Las estructuras de control o de repetición son parecidas a las que hemos aprendido en C, sin embargo, no son completamente iguales; las diferencias radican en otro tipo de palabras reservadas, la estructura del código, entre otras cosas. Otro de los rasgos importantes de Python son las librerías que se pueden manejar, con las cuales se pueden hacer una gran variedad de actividades diversas, como el estudio del lenguaje, graficar, etc. La instalación de estas librerías depende mucho del ambiente que se emplee para trabajar; por ejemplo, el distribuidor de Anaconda instala todo este material para que pueda ser empleado desde el primer uso.[2] En cambio, si se maneja Python desde la consola del sistema, habrá que instalar cada librería.

DESARROLLO Y RESULTADOS

Python también maneja distintas estructuras de control y de repetición por medio de palabras reservadas comúnes, con las que resulta sencillo realizar un programa eficaz y las cuales tienen una amplia gamad de aplicaciones.

La estructura de control `if-elif-else` se llama así porque esas son las palabras reservadas útiles para esta función. La primera, `if`, selecciona una opción contenida en una condición. La segunda y la tercera sólo se pueden escribir cuando se haya escrito un `if` con anterioridad; la diferencia radica en que `elif` propone otra condición, mientras que `else` representa el final de la estructura, en caso de que ninguna opción anterior haya resultado verdadera. Se pueden omitir las dos últimas cuando no hay otra acción a ejecutar, sin embargo, en ocasiones se usa `pass` para concluir la sintaxis de la estructura. No hay que olvidar que, ante cualquier estructura, hay que escribir dos puntos; de otra manera, el programa regresará un error.

```
def numeros(num):
  if num == 1:
    print("tu numero es 1")
  elif num == 2:
    print("tu numero es 2")
  elif num == 3:
    print("tu numero es 3")
  elif num == 4:
    print("tu numero es 4")
  else:
    pass

numeros(2)
numeros(4)
numeros(5)
```

```
[Out] tu numero es 2

      tu numero es 4
```

Las estructuras de control permiten hacer repeticiones tal y como lo hacen también en C. De igual forma, las estructuras de control son `while` y `for`. La primera se utiliza cuando no se conoce el número de iteraciones que van a ocurrir, mientras que el segundo tiene un límite.

Ambas estructuras pueden detenerse con `break` si se llega a una condición deseada.

```
def cuenta(limite);
  i = limite
  while True:
    print(i)
    i = i - 1
    if i == 0:
      break

cuenta(5)
```

```
[Out] 5

[Out] 4

[Out] 3

[Out] 2
```

```
[Out] 1
```

Por su lado, las iteraciones con `for`, una vez que terminan, pueden tener una acción final que se ejecuta una sola vez; esto se hace por medio de `else`, la misma palabra reservada usada en las estructuras `if`.

```
for i in range(2, 0, -1):
    print(i)
  else:
    print("Cuenta finalizada")
```

```
[Out] 2

[Out] 1

[Out] Cuenta finalizada
```

Una de las herramientas de trabajo de Python más importantes son las librerías, las cuales contienen una gran cantidad de funciones. Antes de usar una librería, hay que importarla por medio de `import` más el nombre de dicha librería. No es necesario hacerlo desde el inicio del programa, se puede hacer en el momento en que se necesite.

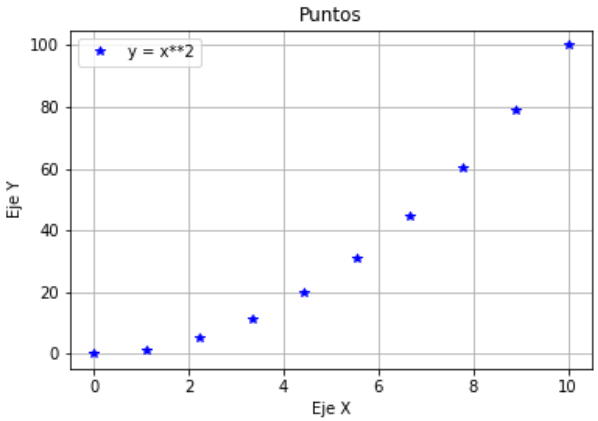
```
import matlib
import numpy
from nltk import *
```

Por último, se puede graficar por medio de Python y una de las librerías que incluye, como se muestra en el siguiente código.

```
%pylab inline

import mathplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

x = linspace(0, 10, 10)

fig, ax = plt.subplots(facecolor='w', edgecolor='k')
ax.plot(x, sin(x), marker='*', color='b', linestyle='None')

ax.grid(True)
ax.set_xlabel('Eje X')
ax.set_ylabel('Eje Y')
ax.grid(True)
```

```
15 ax.legend(["y = x**2"])
16
17 plt.title('Puntos')
18 plt.show()
19
20 fig.savefig("grafica.png")
```

Así se obtiene la gráfica que se muestra a continuación.



CONCLUSIONES

Las herramientas que Python ofrece para trabajar son excepcionales al momento de tratar con información y datos, ya que provee bastante organización y, con su filosofía, es posible elaborar un código legible.

REFERENCES

1. *Tutorial de Python.* Fecha de consulta: 10 de abril de 2020. Recuperado de http://docs.python.org.ar/tutorial/3/real-index.html

2. Distribuidor de Anaconda. Fecha de consulta: 10 de abril de 2020. Recuperado de https://www.anaconda.com/why-anaconda/