	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:	Ing Adrian Ulises Mercado Martínez
Asignatura:	Algoritmos y Estructuras de Datos I
Grupo:	13
No de Práctica(s):	10
Integrante(s):	Hernández Rojas Mara Alexandra
No. de Equipo de cómputo empleado:	No Aplica
No. de Lista o Brigada:	7
Semestre:	2020-2
Fecha de entrega:	07/06/2020
Observaciones:	|

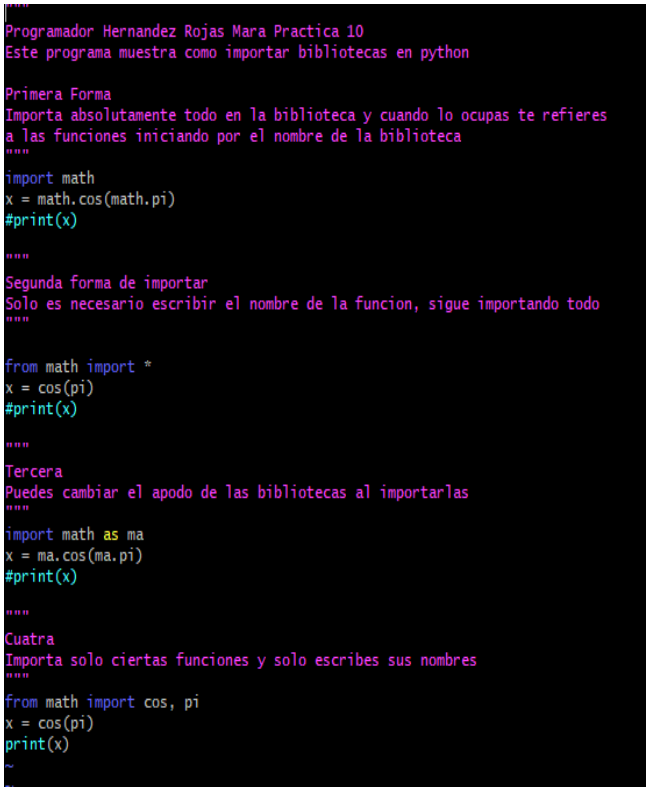
CALIFICACIÓN: _____

Introducción

El objetivo de ésta práctica es aplicar las bases del lenguaje Phyton (estructuras de selección) para crear diversos programas desde el entorno Cygwin.

Desarrollo con actividades

Actividad 1



```
"""
Programador Hernandez Rojas Mara Practica 10
Este programa muestra como importar bibliotecas en python

Primera Forma
Importa absolutamente todo en la biblioteca y cuando lo ocupas te refieres
a las funciones iniciando por el nombre de la biblioteca
"""
import math
x = math.cos(math.pi)
#print(x)

"""
Segunda forma de importar
Solo es necesario escribir el nombre de la funcion, sigue importando todo
"""

from math import *
x = cos(pi)
#print(x)

"""
Tercera
Puedes cambiar el apodo de las bibliotecas al importarlas
"""
import math as ma
x = ma.cos(ma.pi)
#print(x)

"""
Cuatra
Importa solo ciertas funciones y solo escribes sus nombres
"""
from math import cos, pi
x = cos(pi)
print(x)
~
```

Este programa no smuestra cuatro formas de inserter bibliotecas para trabajar con Python:
La primera de ella nos ayuda a importer la biblioteca junto con todas sus operaciones, se invocan así:
import biblioteca
biblioteca.funcion(argumentos)
La segunda de ella nos ayuda a importer la biblioteca junto con todas sus operaciones, se invocan así
from biblioteca import *
funcion(argumentos)
La tercera de ella nos ayuda a importer la biblioteca junto con todas sus operaciones, mientras cambiamos el apodo de la biblioteca se invocan así
import biblioteca as apodo
apodo.funcion(argumentos)
La tercera de ella nos ayuda a importer solo ciertas funciones mientras la biblioteca conserva sun ombre original y se invocan así
from biblioteca import funcion1, funcion2, funcionN
funcion(argumentos)

Actividad 2

Esta function nos indica como debe usarse la estructura de contro **for** para recorrer distintas estructuras en el lenguae Python:

Para una lista: **for auxiliar in [lista]:**
Utilizando rango: **for auxiliar in range(fin)** inicia en 0 termina en fin-1 incrementa 1
for auxiliar in range(incio, fin): incrementa 1
for auxiliar in range(inicio, fin, incremento): Defines el incremento
Para un diccionario: **for clave, valor in diccionario.items():** recorre claves y valores
for clave in diccionario.keys(): recorre claves
for valor in diccionario.values(): recorre valores
for idx, x in enumerate(diccionario): recorre los indices
Utilizando elsefor: **for auxiliar in range(fin):**
Acciones
else:
Acciones

En esta variante una ve que termina con el ciclo for entra a un else, si el ciclo se ropme antes de terminarlo no entra al else.

```
"""
Programador Hernandez Rojas Mara Alexandra Practica 10
Este programa ejemplifica como usar la estructura de control for en phyton

For para listas
"""
def forlist():
    for x in [1, 2, 3, 4, 5]:
        print(x)

"""
For para rangos
"""
def forrange():
    for x in range(5):
        print(x)

    for y in range(-3, 3):
        print(y)

    for z in range(-4, 2, 1):
        print(z)

    for i in range(5, 0, -1):
        print(i)

"""
For para diccionarios
"""
def fordic():
    diccionario = {'manzana': 1, 'pera':3, 'uva':10}
    for clave, valor in diccionario.items():
        print(clave," - ",valor)

    for clave in diccionario.keys():
        print(clave)

    for valor in diccionario.values():
        print(valor)

    for idx, x in enumerate(diccionario):
        print("El indice {} del elemento {}".format(idx, x))
```

```
"""
Else de for
"""

def elsefor():
    for x in range(5):
        print(x)
    else:
        print("La cuenta se termino")


def elsefor2():
    for x in range(5):
        print(x)
        if x== 2:
            break
    else:
        print("la cuenta se termino")
    print("\nNunca entro a elsefor2\n")

"""
En else for 2
Se rompio el ciclo y nunca entra al eslse
"""


if __name__ == "__main__":
    print("for de lista")
    forlist()
    print("for de rango")
    forrange()
    print("for de diccionario")
    fordic()
    print("elsefor")
    elsefor()
    print("elsefor2")
    elsefor2()
```

```
maale@LAPTOP-GIEKRR9A /cygdrive/c/users/maale/
$ Python for.py
for de lista
1
2
3
4
5
for de rango
0
1
2
3
4
-3
-2
-1
0
1
2
-4
-3
-2
-1
0
1
5
4
3
2
1
for de diccionario
```

```
1
for de diccionario
manzana   -  1
pera   -  3
uva   -  10
manzana
pera
uva
1
3
10
El indice 0 del elemento manzana
El indice 1 del elemento pera
El indice 2 del elemento uva
elsefor
0
1
2
3
4
La cuenta se termino
elsefor2
0
1
2

Nunca entro a else
```

Actividad 3

Utilizamos la estructura while para calcular el factorial de un número, la estructura de while es así:
while condicion:
    acciones
    incremento
Recuerda que el incremento se tiene que escribir así: **auxiliar = auxiliar +1**

```
Programador Hernandez Rojas Mara Alexandra Practica 10
Factorial con ciclo while
"""
def fact(a):
    i = 2
    t = 1
    while i<=a:
        t = t * i
        i = i + 1
    return t

if __name__ == "__main__":

    a = int(input("Ingrese un numero: "))
    print(fact(a))
~
~
```

```
maale@LAPTOP-GIEKRR9A /
$ Python factorial.py
Ingrese un numero: 5
120
```

Actividad 4

Con este ejercicio nos devuelve el valor máximo entre un grupo de tres entradas utilizando las estructuras de selección if elif y else, que en C sería el equivalente de varios ciclos if/else anidados:

if condición:
    acciones
elif condición:
    acciones
else:
    acciones

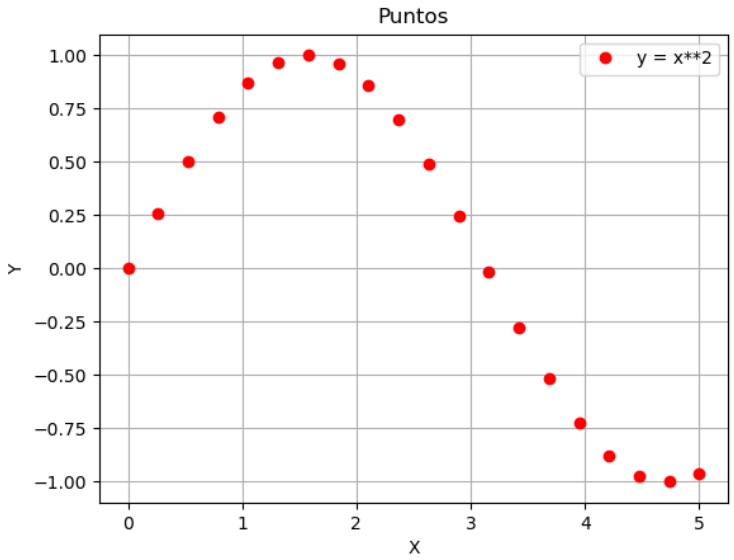
```
"""
Programador Hernandez Rojas Mara Alexandra Práctica 10
Este programa encuentra el numero mayor de tres candidatos
con las estructuras de seleccion if, elif, else en python
"""
def numeroMayor(a, b, c):
    if a > b and a > c:
        print("El numero mayor es {} ".format(a))
    elif( b > c and b > a):
        print("El numero mayor es {} ".format(b))
    else:
        print("El numero mayor es {} ".format(c))


if __name__ == "__main__":
    a = int(input("Numero: "))
    b = int(input("Numero: "))
    c = int(input("Numero: "))
    numeroMayor(a, b, c)
~
~
```

```
$ Python numeroMAyor.py
Numero: 2
Numero: 5
Numero: 8
El numero mayor es 8
```

Actividad 5

Generamos una gráfica en Python con la biblioteca **matplotlib.pyplot**

```
Programdor Hernandez Rojas Mara Alexanra Practica 10
Esta es la gráfica que viene en el manual de prácticas de la asignatura
"""
#%pylab inline esta linea genero un error

import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from numpy import *

x = linspace(0, 5, 20)

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

ax.grid(True)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.grid(True)
ax.legend(["y = x**2"])

plt.title('Puntos')
plt.show()

fig.savefig("graf.png") #El resultado se va a guardar con este nombre
```



Conclusiones

Durante la práctica pude observer que realmente no importa tanto el lenguaje de programación en el que se trabaje si el programador aprende a organizer bien sus ideas y perfecciona la lógica de los programas que implemente. Una vez que se domina la parte del diseño lo de menos es aprender la sintaxis.

En el linguae C y en phyton encontramos equivalencias; las estructuras de repetición son prácticamente las mismas, el for se maneja sitinto deacuerdo a cuantos argumentos entren en él y que tipo de datos recorra, en el while el incremento se tiene que expresar manualmente de la formacompleta y un switch o un ciclo if/else anidado se expresa con las sentencias if, elif, else.

Aquí se hace más evidente la diferencia entre un lenguaje de programación de un nivel más alto como lo es Phyton, sus instrucciones son fáciles de recordar por casi ser palabras/sentencias del idioma inglés, y uno más bajo como lo es C que maneja amstracciones de diferentes conceptos como lo serían las estructuras y los apuntadores. Claro está Python si necesita apuntadores pero el programador no los va a manejar directamente y existen entre sus tipos de datos estructuras bien definidas como los diccionarios.