



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Adrian Ulises Mercado

Profesor:

Asignatura:

Estructura de datos y Algoritmos
I

Grupo:

13

No de Práctica(s):

#10

Integrante(s):

Jose Abraham Hernandez Vargas

No. de Equipo de cómputo empleado:

No. de Lista o Brigada:

Brigada 5

Semestre:

2020-2

Fecha de entrega:

07/06/2020

Observaciones:

CALIFICACIÓN: _____

INTRODUCCION.

Práctica introducción a Python II veremos cómo se manejan las estructuras de datos , en este caso veremos un ejemplo de pila, posteriormente se revisará las declaraciones y usos de bibliotecas, también usaremos funciones repetitivas como es el caso de for y while y veremos cómo se podrá graficar un programa utilizando las librerías matplotlib.

DESARROLLO

Comenzaremos con el desarrollo de la práctica, primeramente tenemos un programa que utiliza las declaraciones if else, elif, aplicadas a un programa que ordena números por cantidad(<,>), y se utilizara la función input para pedir datos por terminal y teclado.

Código practica10.py

```
def numeroMayor(a,b, c):
    if a > b and a > c:
        print("el numero es {}".format(a))
    elif(b > c and b > a):
        print("el numero es {}".format(b))
    else:
        print("el numero es {}".format(c))

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

- A continuación, veremos cómo usar las bibliotecas y como de declaran y usaremos las funciones coseno y seno para obtenerlo de un valor pi u otro Angulo.

Código practica10bib.py

```
import math
from math import *
from math import cos, pi
x= math.cos(math.pi)
x = cos(pi)
print(x)
```

Código practica10bib2.py

```
import math

print(dir(math))
#como usar las funciones
hel(math.log)

#se puede importar una biblioteca y asignarle un alias
x = ma.cos(ma.pi)
```

- Ahora veremos cómo utilizar las funciones repetitivas, para este caso el for donde lo aplicaremos a listas, rangos y en diccionarios de Python

Código practica10for.py

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

    for x in ["uno", "dos", "tres", "cuatro", "cinco"]
        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, 2):
        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")

if __name__ == "__main__":
    forlist()
    forrange()

```

Ahora veremos cómo hacer un ciclo repetitivo con la función while, donde lo aplicaremos para sacar el factorial de un numero que se solicitara por terminal

Practica10while.py

```

def factorial(n):
    #espaios en blanco causan problemas
    i = 2

```

```

temp = 1
while i <= n:
    temp = temp*i
    i = i+1
return temp

if __name__ == "__main__":
    a = int(input("ingresa un numero"))
    print(factorial(a))

```

Ahora veremos cómo aplicar lo anterior a diferentes tipos de programa y aplicaciones, primeramente veremos el caso de aplicar las funciones para hacer un pila, donde las funciones esenciales de la pila ya están definidas en el lenguaje de Python, a diferencia del lenguaje C.

pila2.py

```

def insertar(lista,dato):
    lista.append(dato)#agregar al final

def borrar(lista):
    dato=lista.pop()#elimina al final de la lista
    return dato

def imprimir_pila(lista):
    lista.reverse()
    for x in lista:
        print(x)
    print()
    lista.reverse()

def main():
    pila = [0]
    insertar(pila, "lista1")
    insertar(pila, 2)
    imprimir_pila(pila)
    print(borrar(pila))
    print()
    imprimir_pila(pila)

if __name__ == "__main__":
    main()

```

Ahora se hará la instalación de la biblioteca para graficar

Código practica10graf.py

```
#para instalar python -m pip install -U matplotlib
#python -m pip3 install -U matplotlib

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

x=linspace
```

Ejercicio de tarea para implementar el enfoque mergesort para ordenar una serie de números o elementos en forma de lista con la función merge sort

mergesortE.c

```
/* programa en c merge sort*/
#include<stdlib.h>
#include<stdio.h>

// Merges 2 subarrays de array[].
// primer array[l..m]
// Segundo[m+1..r]
void merge(int arr[], int l, int m, int r)
{
    int i, j, k;
    int n1 = m - l + 1;
    int n2 = r - m;

    int L[n1], R[n2];

    for (i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (j = 0; j < n2; j++)
        R[j] = arr[m + 1 + j];

    i = 0;
    j = 0;
    k = l;
    while (i < n1 && j < n2)
    {
        if (L[i] <= R[j])
        {
            arr[k] = L[i];
            i++;
        }
        else
        {
            arr[k] = R[j];
            j++;
        }
    }
}
```

```

        }
        k++;
    }

    while (i < n1)
    {
        arr[k] = L[i];
        i++;
        k++;
    }

    while (j < n2)
    {
        arr[k] = R[j];
        j++;
        k++;
    }
}

void mergeSort(int arr[], int l, int r)
{
    if (l < r)
    {
        int m = l+(r-1)/2;

        mergeSort(arr, l, m);
        mergeSort(arr, m+1, r);

        merge(arr, l, m, r);
    }
}

void printArray(int A[], int size)
{
    int i;
    for (i=0; i < size; i++)
        printf("%d ", A[i]);
    printf("\n");
}

int main()
{

```

```
int arr[] = {12, 11, 13, 5, 6, 7};
int arr_size = sizeof(arr)/sizeof(arr[0]);

printf("Given array is \n");
printArray(arr, arr_size);

mergeSort(arr, 0, arr_size - 1);

printf("\nSorted array is \n");
printArray(arr, arr_size);
return 0;
}
```

EJECUCION

Mergesort

```
[Running] cd "c:\Users\moonw\Desktop\EDA1\p10\" && gcc ejercicio13may.c -o ejercicio13may && "c:\Users\moonw\Desktop\EDA1\p10\"ejercicio13may.exe
array dado
12 11 13 5 6 7

array mezclado y ordenado is
5 6 7 11 12 13

[Done] exited with code=0 in 0.715 seconds
```

Backtracking

```
[Running] cd "c:\Users\moonw\Desktop\EDA1\p10\"  
Elementos en la mochila optima [(2, 1), (2, 2), (15, 12)], con beneficio 15, y con un peso de 15  
  
[Done] exited with code=0 in 0.715 seconds
```

pd.py o mochila2.py

```
[Running] cd "c:\Users\moonw\Desktop\EDA1\p10\" && gcc ejercicio13may.c -o ejercicio13may && "c:\Users\moonw\Desktop\EDA1\p10\"ejercicio13may.exe
[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 28, 28, 28, 28, 28, 28, 28, 28, 28, 28, 28, 28], [0, 0, 28, 28, 28, 33, 33, 61, 61, 61, 61, 61, 61, 61], [0, 0, 28, 28, 28, 33, 33, 61, 61, 61, 66, 66, 66, 66], [0, 0, 28, 28, 28, 33, 33, 61, 61, 61, 66, 66, 66, 73, 73, 73], [0, 20, 28, 48, 48, 48, 53, 61, 81, 81, 81, 86, 86, 86, 93, 93]]
93
-----
[Done] exited with code=0 in 0.715 seconds
```


CONCLUSION

En esta practica nos podemos dar cuenta de las diferencias que tiene Python con lenguaje C, cada lenguaje tiene su ventaja pero sabiendo dominar lenguaje C puedes dominar varios lenguajes de programacion

COMENTARIOS

Me parece muy funcional la opción de graficar, pues te sirve para verificar varias cosas y aparte tiene su chiste.

BIBLIOGRAFIA

Tutorial oficial de Python: <https://docs.python.org/3/tutorial/>

Galería de notebooks: <https://wakari.io/gallery>

Matplotlib: <http://matplotlib.org/>