Carátula para entrega de prácticas	
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:	Edgar Tista García
Asignatura:	Estructuras de Datos y Algoritmos I
Grupo:	1
No. de Práctica(s):	10
Integrante(s):	Ugalde Velasco Armando
No. de Equipo de cómputo empleado:	
No. de Lista:	38
Semestre:	2020-2
Fecha de entrega:	5 de abril de 2020
Observaciones:	

CALIFICACIÓN: _____

OBJETIVO

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

ACTIVIDAD 1

I) EJECUTA LA APLICACIÓN JUPYTER NOTEBOOK Y ABRE EL ARCHIVO CORRESPONDIENTE A LA PRACTICA 10. SIGUE LAS INSTRUCCIONES QUE AHÍ SE MENCIONAN PARA LA EJECUCIÓN DEL CONTENIDO DE LAS CELDAS. AL IGUAL QUE EN LA PRÁCTICA ANTERIOR, PARA CADA UNA DE LAS SECCIONES ESCRIBE COMENTARIOS CON RESPECTO A PYTHON Y SU DIFERENCIA CON EL LENGUAJE C.

1. Estructuras de control selectivas

- En Python no es necesario colocar paréntesis en las condiciones, a diferencia del lenguaje C.
- En Python es posible encadenar más de una condición sin tener que agregar un operador booleano, a diferencia del lenguaje C.
- **Elif** en Python es equivalente a **else if** en C.
- En Python no es necesario colocar corchetes para indicar el bloque de código a ejecutar si se cumple la condición, únicamente se colocan dos puntos **(:)** y se indenta el código correctamente.
- En Python no es posible utilizar el operador ternario.

2. Estructuras de control repetitivas

- El ciclo **while** tiene el mismo funcionamiento en ambos lenguajes. Sin embargo, en Python, como en el caso de las estructuras de control selectivas, no es necesario colocar corchetes para delimitar el bloque de código a ejecutar.
- En C no existe un ciclo **for in** como tal. En Python no existe un ciclo **for** como el del lenguaje C, donde se utilice un contador explícitamente.
- En Python, es posible emular un ciclo **for** como el del lenguaje C, utilizando el ciclo **for in** y la función **range**, que genera una lista con los números especificados.

3. Bibliotecas

- Al igual que en el lenguaje C, Python cuenta con una biblioteca estándar, donde se encuentran las funcionalidades básicas del lenguaje.

- En ambos lenguajes se encuentran disponibles un gran número de bibliotecas para distintos fines.

4. Graficación

- En ambos lenguajes existen bibliotecas utilizadas para la graficación. Cabe mencionar que las bibliotecas de Python para este fin son más claras y accesibles en mi opinión, debido a la naturaleza del lenguaje.

5. Ejecución desde ventana de comandos.

- Utilizando el lenguaje C, es posible ejecutar el programa directamente en la línea de comandos. En cambio, en un programa realizado en Python, es necesario ejecutarlo mediante el intérprete, es decir, con el comando **python**.

- La entrada de datos en Python siempre se almacena como una cadena, por lo que es necesario convertir posteriormente los datos obtenidos conforme nuestro objetivo.

ACTIVIDAD 2

EN LA PRÁCTICA ANTERIOR SE TRABAJÓ CON UNA ALTERNATIVA LA NOTEBOOK LA CUAL CONSISTE EN CREAR LOS ARCHIVOS CON EXTENSIÓN .PY Y EJECUTARLOS DESDE CONSOLA. UNA TERCER ALTERNATIVA PARA HACER USO DEL LENGUAJE DE PROGRAMACIÓN ES MEDIANTE UN IDE.

VERIFICA QUE SE ENCUENTRA INSTALADO EL EDITOR SPYDER. Y UTILIZANDO ESTE EDITOR REALIZA UN ARCHIVO PARA CADA UNO DE LOS EJERCICIOS SIGUIENTES.

1. Escribir un programa que contenga una contraseña inventada (números y letras), que le pregunte al usuario la contraseña, y no le permita continuar hasta que la haya ingresado correctamente. (Ejercicio 2.1)

Para realizar el programa únicamente se utilizaron las siguientes líneas de código. Nos podemos percatar de que la programación en Python simplifica en gran forma la escritura de nuestro código, a diferencia de C.

Primero, se solicitó al usuario la contraseña utilizando la función **input**, y el valor obtenido se asignó a la variable **password**. Posteriormente, se ejecutó el ciclo **while** mostrado, que se encargaba de comprobar si la contraseña era correcta. De no ser así, el ciclo continuaba ejecutándose, y ésta se solicitaba de nueva cuenta. Finalmente, cuando el usuario ingresaba una contraseña correcta, se imprimía el mensaje “Has logrado ingresar”.

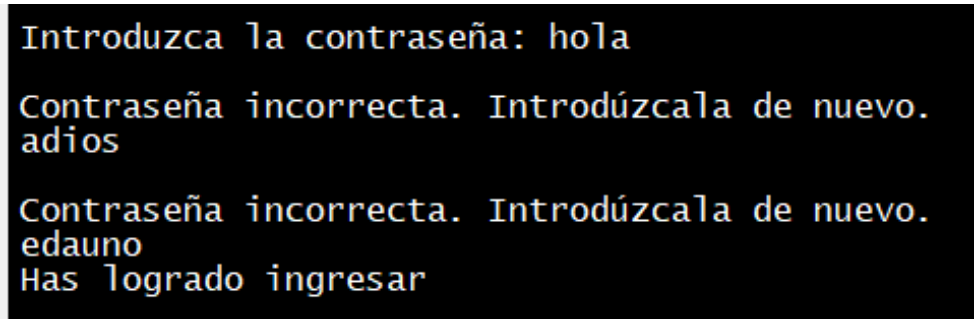
```
password = input("Introduzca la contraseña: ")

while (password != "edauno"):
    password = input("Contraseña incorrecta. Introdúzcala de nuevo.\n")

print("Has logrado ingresar")
```

CÓDIGO DEL PROGRAMA

A continuación, se muestra una captura de pantalla de la ejecución del programa:



EJECUCIÓN DEL PROGRAMA

2. Modificar el programa anterior para que solamente permita una cantidad fija de intentos y después finalice. (Ejercicio 2.2).

Primero, se creó la variable **numIntentos**, para establecer el número de intentos posibles. Se solicitó la contraseña de la misma forma que en el programa anterior, pero se agregó la impresión del valor de **numIntentos**. Después, se realizó el decremento pertinente a la variable **numIntentos**:

```
numIntentos = 5
password = input("Introduzca la contraseña (" + str(numIntentos) + " intento
numIntentos -= 1
```

SE SOLICITA LA CONTRASEÑA Y SE ESTABLECE EL NÚMERO DE INTENTOS DISPONIBLES

Al igual que en el programa anterior, se utilizó un ciclo **while** para solicitar la contraseña mientras ésta no fuera correcta. Sin embargo, se agregó una segunda condición a éste: que el número de intentos disponibles fuera mayor a cero.

```
while password != "edauno" and numIntentos > 0:
    password = input("Contraseña incorrecta. Introdúzcala de nuevo. (" + str(numIntentos)
    numIntentos -= 1
```

CICLO PARA SOLICITAR LA CONTRASEÑA EN CASO DE QUE NO SEA CORECCTA

Finalmente, cuando alguna de las dos condiciones no se cumpliera, es decir, cuando la contraseña fuera correcta, o bien, el número de intentos se hubiera agotado, el ciclo terminaría su ejecución, y se ejecutarían las siguientes líneas:

```
if numIntentos > 0:
    print("Has logrado ingresar")

else:
    print("Se ha agotado el número de intentos")
```

MENSAJE FINAL DEL PROGRAMA

Es decir, si aún hay intentos disponibles, quiere decir que el ciclo terminó su ejecución porque ingresó la contraseña correcta, por lo que se imprime el mensaje *"Has logrado ingresar".* De lo contrario, el ciclo terminó su ejecución porque el número de intentos se agotó, por lo que se imprime el mensaje **"Se ha agotado el número de intentos".**

A continuación, se muestran capturas de pantalla de la ejecución del programa:

```
Introduzca la contraseña (5 intentos restantes): fasdg

Contraseña incorrecta. Introdúzcala de nuevo. (4 intentos restantes)
gf

Contraseña incorrecta. Introdúzcala de nuevo. (3 intentos restantes)
hgds

Contraseña incorrecta. Introdúzcala de nuevo. (2 intentos restantes)
afsd

Contraseña incorrecta. Introdúzcala de nuevo. (1 intentos restantes)
ghdfs
Se ha agotado el número de intentos
```

SE AGOTA EL NÚMERO DE INTENTOS DISPONIBLES

```
Introduzca la contraseña (5 intentos restantes): fs

Contraseña incorrecta. Introdúzcala de nuevo. (4 intentos restantes)
edauno
Has logrado ingresar
```

SE INTRODUCE CORRECTAMENTE LA CONTRASEÑA

ACTIVIDAD 3

ELABORA UN PROGRAMA EN EL CUAL EL ALUMNO INGRESE LAS CALIFICACIONES DE SUS TAREAS (5), POSTERIORMENTE LAS CALIFICACIONES DE SUS EXÁMENES (4) Y LAS CALIFICACIONES DE SUS PRÁCTICAS DE LABORATORIO (5) (SE SUGIERE HACER UNA LISTA PARA CADA UNO).

Para cumplir con el objetivo, el programa se realizó de la siguiente forma:

Primero, se crearon tres listas para almacenar los datos correspondientes a los promedios de tareas, exámenes y prácticas. Posteriormente, se ejecutaron 3 ciclos, para solicitar al usuario los datos e ingresarlos a las listas correspondientes mediante el método **append**. También fue necesario convertir el valor retornado por **input** a un dato de tipo **float**, como se puede observar en la captura.

```
tareas = []
examenes = []
practicas = []

for i in range(5):
    tareas.append( float(input("Introduzca la calificación de la tarea " + str(

for i in range(4):
    examenes.append( float(input("Introduzca la calificación del examen " + str

for i in range(5):
    practicas.append( float(input("Introduzca la calificación de la practica "
```

SE SOLICITAN LOS DATOS Y SE INGRESAN A LAS LISTAS RESPECTIVAS

Se definieron dos funciones auxiliares en el programa: **promedio** y **toStr**. La primera retorna el promedio de los números en una lista, utilizando los métodos **sum** y **len**, y la segunda devuelve un número redondeado a dos decimales, convertido a una cadena.

```
def promedio(lista):
    return sum(lista) / len(lista)

def toStr(num):
    return str(round(num, 2))
```

FUNCIONES AUXILIARES

Con ayuda de las funciones anteriores, se calcularon los promedios respectivos y se almacenaron en variables con los nombres pertinentes, como se observa en la siguiente captura. Se calculó también la calificación inicial, sumando el valor respectivo de los promedios de exámenes y prácticas. Además, se creó una variable adicional para almacenar la calificación final, y que la calificación inicial no fuera modificada directamente por la condición del promedio de las prácticas.

```
promedioTareas = promedio(tareas)
promedioExamenes = promedio(examenes)
promedioPracticas = promedio(practicas)
califInicial = promedioExamenes*0.6 + promedioPracticas*0.4
califFinal = califInicial
```

PROMEDIOS

Se expresó la condición respecto a la calificación del laboratorio utilizando las condicionales **if** y **elif**: si ésta era mayor a 8.5, la calificación final aumentaba 5 décimas, y si era menor o igual a 7, disminuía por la misma cantidad. Si ninguna de estas condiciones se cumplía, la calificación final permanecía igual. A continuación, se muestra el bloque de código donde se realizó lo anterior:

```
if promedioTareas > 8.5:
    califFinal += 0.5

elif promedioTareas <= 7:
    califFinal -= 0.5
```

PROMEDIOS

Finalmente, se imprimieron los resultados, utilizando la función **print** y la función auxiliar antes mencionada: **toStr**.

A continuación, se muestra una captura de pantalla de la ejecución del programa:

```
Introduzca la calificación de la tarea 1: 10
Introduzca la calificación de la tarea 2: 10
Introduzca la calificación de la tarea 3: 9
Introduzca la calificación de la tarea 4: 9
Introduzca la calificación de la tarea 5: 10
Introduzca la calificación del examen 1: 9
Introduzca la calificación del examen 2: 10
Introduzca la calificación del examen 3: 9
Introduzca la calificación del examen 4: 9
Introduzca la calificación de la practica 1: 9
Introduzca la calificación de la practica 2: 8.6
Introduzca la calificación de la practica 3: 9.2
Introduzca la calificación de la practica 4: 8.5
Introduzca la calificación de la practica 5: 8.5
Tu promedio de exámenes fue de 9.25
Tu promedio de laboratorio fue de 8.76
Tu calificación inicial es de 9.05
Tu calificación de tareas es de 9.6, por lo tanto tu calificación final es de 9.55
```

EJECUCIÓN DEL PROGRAMA

CONCLUSIONES

Gracias a la realización de la práctica se obtuvieron más conocimientos acerca del lenguaje Python. Se presentaron más conceptos fundamentales propios del lenguaje, es decir, el funcionamiento y utilización de las estructuras de control repetitivas y selectivas, y la existencia de bibliotecas para distintos fines. El lenguaje presenta diferencias y similitudes con el lenguaje C, las cuales se analizaron a profundidad en la actividad 1. Algunas de las más importantes son:

- Las estructuras de control selectivas son muy similares a las del lenguaje C. La sentencia **elif** es equivalente a **else if** en C.
- En Python no es necesario colocar corchetes para delimitar bloques de código.
- Ambos lenguajes cuentan con una biblioteca estándar, además de un gran número de bibliotecas disponibles para otros fines, como la graficación.
- Es posible ejecutar programas de ambos lenguajes desde la ventana de comandos, pero para ejecutar un programa escrito en Python, es necesario hacerlo mediante el intérprete.

Para la realización de los programas se utilizó el IDE **Spyder**, el cual facilitó en gran medida la escritura del código. En mi opinión, éste es más completo que el entorno de **Jupyter Notebook**, ya que cuenta con más funcionalidades, propias de un IDE; por lo tanto, considero que provocó que la escritura de los programas fuera más cómoda y amena.

Se realizaron programas simples para familiarizarse con el lenguaje. Nos podemos dar cuenta de que, a diferencia del lenguaje C, Python nos permite realizar programas de una forma más expresiva y con menos líneas de código, en muchas ocasiones.

Considero que se cumplió el objetivo de la práctica, ya que logré familiarizarme en mayor medida con el lenguaje Python mediante la realización de las actividades antes mencionadas. Sin embargo, considero que algo más útil y acertado en cuanto al objetivo principal de la materia se refiere, podría ser analizar el modelo de costo de algunas funciones y estructuras de datos presentes en el lenguaje.