Universidad De San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Lenguajes Formales y de Programación
Sección “B -”


# “MANUAL DE USUARIO”

Luis Angel Barrera Velásquez

Carné: 202010223

# Índice

# Objetivos

## General:

Brindar al lector información acerca del desarrollo de esta práctica y como se le dio solución para una futura actualización o corrección de errores técnicos.

## Específicos:

- Detallar con palabras la solución presentada en el código paso a paso para seguir un orden y lógica del programa.

- Proporcionar los conceptos que fueron empleados para la solución de esta práctica con teoría para que sea más entendible la solución expresada en palabras.

# Introducción

Este manual técnico esta elaborado con el propósito de detallar el funcionamiento del código fuente de la practica y en caso de que lo lea alguien con conocimientos sobre programación sea de ayuda para realizar futuras actualizaciones sobre el programa o corrección de algún error que suceda durante la ejecución. Es importante recalcar que su solución de esta practica 1 es bastante sencilla abarcando conceptos bastante básicos de programación.

# Descripción de la Solución

Para la solución de esta practica fue necesario utilizar la lectura de archivos de texto plano y guardarlos en un buffer para realizar el respectivo análisis del archivo. Pues para encontrar cada uno de los datos dentro del archivo fue realizando varios recorridos al texto entero con varios for y con diferentes condiciones if para capturar los datos de la siguiente manera:

- Primero se realizo un for recorriendo todo el texto eliminando espacios en blanco innecesarios y sin relevancia en caso de que hubiesen también tabulaciones (\t) y saltos de línea (\n).

- Luego se recorrió el archivo dividiendo en tres el contenido. Antes del signo = esta el nombre del curso después del signo = hasta el } están los datos de los estudiantes y por ultimo están los parámetros.

- Luego se realizó un recorrido a la parte de los datos de los estudiantes extrayendo sus nombres interpretando desde donde empieza las " y hasta donde vuelve a aparecer las comillas " después su nota que esta después de el punto y coma ; hasta antes de > teniendo así los datos de cada estudiante y guardándolos en dos arreglos globales uno con los nombres y otro con las notas exactamente en el mismo orden de cómo se fueron leyendo.

- después se recorrido la parte de los parámetros guardando cada parámetro en otro arreglo global ya que también serán usados para la generación de reportes.

- Para la opción de generación de reportes se crearon dos copias de los arreglos anteriores para no modificar los originales, estas copias se le fueron aplicados los ordenamientos según lo solicitado en los parámetros y generando tanto para los reportes de consola como para los reportes en html.

# Conceptos utilizados para la solución

## Lectura de archivos de texto plano

La entrada y salida de ficheros (comúnmente llamada File I/O) es una herramienta que Python, como la mayoría de los lenguajes, nos brinda de forma estándar. Su utilización es similar a la del lenguaje C, aunque añade mayores características.

El primer paso para introducirnos es saber cómo abrir un determinado archivo. Para esto se utiliza la función open, que toma como primer argumento el nombre de un fichero.

```
1.  open("archivo.txt")
```

No está de más aclarar que esto no significa ejecutar el programa por defecto para editar el fichero, sino cargarlo en la memoria para poder realizar operaciones de lectura y escritura.

En caso de no especificarse la ruta completa, se utilizará la actual (donde se encuentre el script).

Como segundo argumento, open espera el modo (como una cadena) en el que se abrirá el fichero. Esto indica si se utilizará para lectura, escritura, ambas, entre otras.

```
1.  open("archivo.txt", "r")
```

La r indica el modo lectura. Si se intentara utilizar la función write para escribir algo, se lanzaría la excepción IOError. A continuación, los distintos modos.

- r – Lectura únicamente.
- w – Escritura únicamente, reemplazando el contenido actual del archivo o bien creándolo si es inexistente.
- a – Escritura únicamente, manteniendo el contenido actual y añadiendo los datos al final del archivo.

- w+, r+ o a+ – Lectura y escritura.

## Escritura de archivos

Este concepto fue utilizado para la escritura del reporte HTML ya que fue escrito como si escribiéramos un txt pero se uso la sintaxis de HTML y guardarlo como *.html como fue solicitado en el enunciado.

```
1.  f = open("archivo.txt", "w")
```

En este ejemplo he abierto el fichero archivo.txt que, en caso de no existir, será automáticamente creado en el directorio actual. A continuación escribiré algunos datos utilizando la función file.write.

```
1.  f.write("Esto es un texto.")
```

Esta función fue utilizada hasta de ultimo ya que mientras la creación del html se fue concatenando una cadena bastante larga agregando cada etiqueta que se necesitara como por ejemplo:

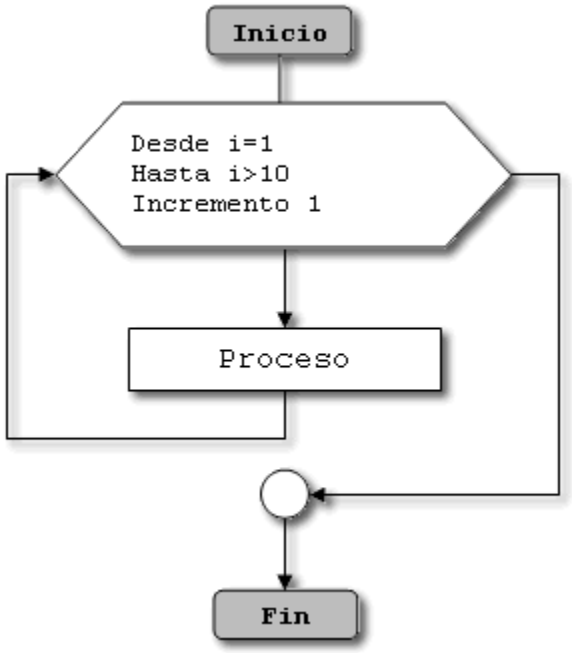
```
1.  cadena+=”<h1>Notas de los estudiantes ordenada ascendentemente</h1>”
```

por ultimo se cerro el archivo leído o de escritura por medio de la función close

```
1.  f.close()
```

## Ciclos for

El bucle for se utiliza para recorrer los elementos de un objeto iterable (lista, tupla, conjunto, diccionario, …) y ejecutar un bloque de código. En cada paso de la iteración se tiene en cuenta a un único elemento del objeto iterable, sobre el cuál se pueden aplicar una serie de operaciones.




## Condicional if

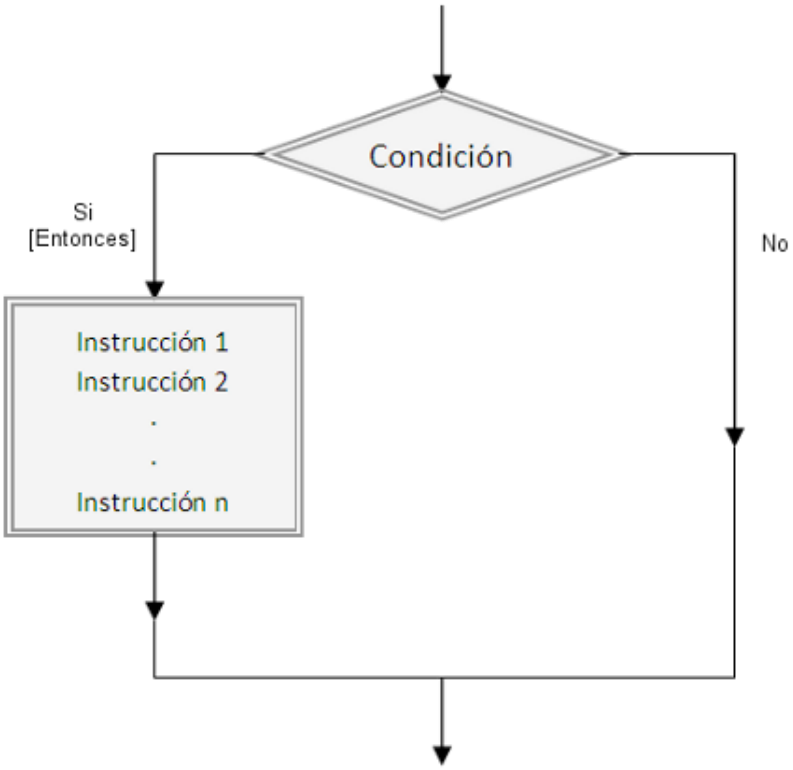
La estructura de control if ... permite que un programa ejecute unas instrucciones cuando se cumplan una condición. En inglés "if" significa "si" (condición).

La sintaxis de la construcción if es la siguiente:

```
if condición:
    aquí van las órdenes que se ejecutan si la condición es cierta
    y que pueden ocupar varias líneas
```

La ejecución de esta construcción es la siguiente:

- La condición se evalúa siempre.
    - Si el resultado es True se ejecuta el bloque de sentencias
    - Si el resultado es False no se ejecuta el bloque de sentencias.




## Condicional if.. else

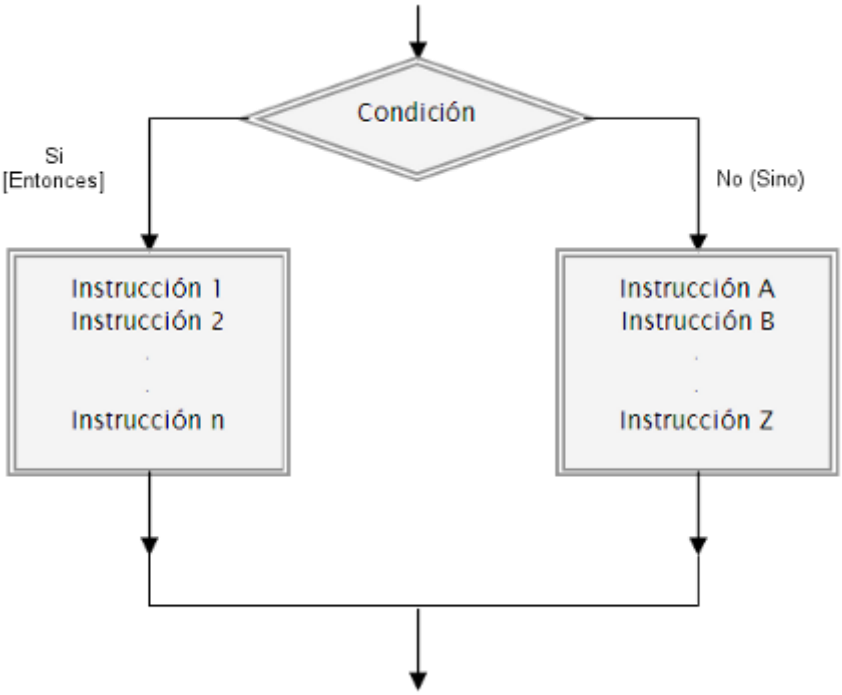
La estructura de control if ... else ... permite que un programa ejecute unas instrucciones cuando se cumple una condición y otras instrucciones cuando no se cumple esa condición. En inglés "if" significa "si" (condición) y "else" significa "si no". La orden en Python se escribe así:

La sintaxis de la construcción if ... else ... es la siguiente:

```
if condición:
    aquí van las órdenes que se ejecutan si la condición es cierta
    y que pueden ocupar varias líneas
else:
    y aquí van las órdenes que se ejecutan si la condición es
    falsa y que también pueden ocupar varias líneas
```

La ejecución de esta construcción es la siguiente:

- La condición se evalúa siempre.
    - Si el resultado es True se ejecuta solamente el bloque de sentencias 1
    - Si el resultado es False se ejecuta solamente el bloque de sentencias 2.




## Variables globales

Las variables globales son las variables con alcance global. El alcance global significa que la variable es accesible desde cualquier lugar del programa. Las variables globales se declaran fuera de las funciones ya que su alcance no se limita a ninguna función. La vida útil de la variable global es igual al tiempo de ejecución del programa.

No tenemos que declarar explícitamente las variables antes de usarlas, por lo tanto, para diferenciar entre una variable local y una global, necesitamos especificar que la variable a la que estamos accediendo es la variable global o no. Podemos especificar una variable como global en Python usando la palabra clave global.

Si pasamos el valor a la variable global dentro de una función sin declararla variable global, el valor se pasará a la nueva variable con el mismo nombre. Y el alcance de la nueva variable estará restringido al alcance de la función.

Por ejemplo:

```
date = "17-06-2002"



def update_date():

  global date #declaracion de la variable global hace referencia a:

  date = "12-03-2021"



update_date()

print(date)
```