Introducción al lenguaje de programación Python

M.C.I. Victor Manuel Mora Romo

1

Modulo III
Tipos de Datos Avanzados o Complejos

2

V.-Tipos de datos complejos

Python, posee además de los tipos ya vistos, 3 tipos más complejos, que admiten una **colección de datos**. Estos tipos son:

- Tuplas
- Listas
- Diccionarios

Estos tres tipos, pueden almacenar colecciones de datos de diversos tipos y se diferencian por su sintaxis y por la forma en la cual los datos pueden ser manipulados.

3

A)Tuplas

Una tupla **es una variable que permite almacenar varios datos inmutables** (no pueden ser modificados una vez creados) de tipos diferentes:

```
mi_tupla = ('cadena de texto', 15, 2.8, 'otro dato', 25)
```

Se puede acceder a cada uno de los datos mediante su índice correspondiente, siendo 0 (cero), el índice del primer elemento:

```
print mi_tupla[1] # Salida: 15
```

4

También se puede acceder a una porción de la tupla, indicando (opcionalmente) desde el índice de inicio hasta el índice de fin:

```
print mi_tupla[1:4] # Devuelve: (15, 2.8, 'otro dato')
print mi_tupla[3:] # Devuelve: ('otro dato', 25)
print mi_tupla[:2] # Devuelve: ('cadena de texto', 15)
```

Otra forma de acceder a la tupla de forma inversa (de atrás hacia adelante), es colocando un índice negativo:

```
print mi_tupla[-1] # Salida: 25
print mi_tupla[-2] # Salida: otro dato
```

5

Listas

Una lista es similar a una tupla con la diferencia fundamental de que permite modificar los datos una vez creados:

```
mi_lista = ['cadena de texto', 15, 2.8, 'otro dato', 25]
```

A las listas se accede igual que a las tuplas, por su número de índice:

```
print mi_lista[1] # Salida: 15
print mi_lista[1:4] # Devuelve: [15, 2.8, 'otro dato']
print mi_lista[-2] # Salida: otro dato
```

Las lista NO son inmutables: permiten modificar los datos una vez creados:

```
mi_lista[2] = 3.8 # el tercer elemento ahora es 3.8
```

Las listas, a diferencia de las tuplas, permiten agregar nuevos valores:

```
mi_lista.append('Nuevo Dato')
```


6

Diccionarios

Mientras que a las listas y tuplas se accede solo y únicamente por un número de índice, los diccionarios permiten utilizar una clave para declarar y acceder a un valor:

```
mi_diccionario = {'clave_1': valor_1, 'clave_2': valor_2, \
                  'clave_7': valor_7}
print mi_diccionario['clave_2'] # Salida: valor_2
```

Un diccionario permite eliminar cualquier entrada:

```
del(mi_diccionario['clave_2'])
```

Al igual que las listas, el diccionario permite modificar los valores

```
mi_diccionario['clave_1'] = 'Nuevo Valor'
```

7

7

Fecha y Hora

Referencia de los códigos de formato de la función datetime.strftime()

Código	Descripción	Ejemplo
%a	Día de la semana, versión corta	Wed
%A	Día de la semana, versión completa	Wednesday
%w	Día de la semana como número [0-6]	3
%d	Día del mes [01-31]	31
%b	Nombre del mes, versión corta	Dec
%B	Nombre del mes, versión completa	December
%m	Mes como número [01-12]	12
%y	Año, versión corta sin centuria	18
%Y	Año, versión completa	2018
%H	Hora [00-23]	17
%I	Hora [00-12]	05
%p	AM/PM	PM
%M	Minuto [00-59]	41
%S	Segundo [00-59]	08
%f	Microsegundo [000000-999999]	548513
%z	Huso horario	+0100
%Z	Zona horaria	CST
%j	Número del día del año [001-366]	365
%U	Número de semana del año, comenzando en domingo [00-53]	52
%W	Número de semana del año, comenzando en lunes [00-53]	52
%c	Versión local de fecha y hora	Mon Dec 31 17:41:00 2018
%x	Versión local de fecha	12/31/18
%X	Versión local de hora	17:41:00
%%	El carácter '%'	%

8

8

Restar fechas no produce fechas

En la clase se realiza la operación siguiente:

```
ahora = datetime.datetime.now()
nacimiento = datetime.datetime(1982, 30, 9)
edad = ahora - nacimiento
```

Es importante destacar que la variable `edad` de este ejemplo no contiene una fecha, o sea no es de tipo *datetime.datetime* sino de tipo *datetime.timedelta* que es un tipo de datos que también viene incluido en el módulo *datetime* y que se usa para referirse no a fechas sino a diferencias respecto a fechas.

9

9

015_Colecciones_listas
016_Colecciones_tuplas
017_Colecciones_sets
018_Colecciones_Diccionarios
020_Nontype

10

10

021_Ejercicio_004_Variables_Avanzadas

Área de Círculos

Crear un programa que calcule el área de un circulo a partir de su diámetro.

El usuario pasará el diámetro del círculo por la entrada, pudiendo usar decimales.

11

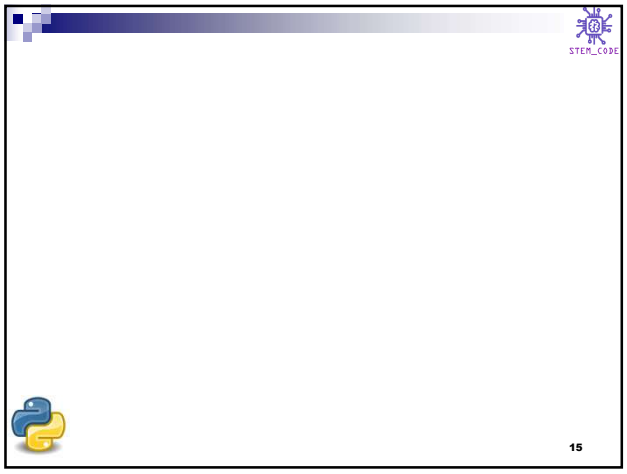
11

12

12

13

14



15

16