

Clase Número: 5

Procesar variables de tipo cadenas de caracteres

Hasta este momento hemos visto como definir variables enteras y flotantes. Realizar su carga por asignación y por teclado.

Para iniciarlas por asignación utilizamos el operador =

```
#definición de una variable entera
cantidad=20
#definición de una variable flotante
altura=1.92
```

Como vemos el intérprete de Python diferencia una variable flotante de una variable entera por la presencia del caracter punto.

Para realizar la carga por teclado utilizando la función input debemos llamar a la función int o float para convertir el dato devuelto por input:

```
cantidad=int(input("Ingresar la cantidad de personas:"))
altura=float(input("Ingresar la altura de la persona en metros ej:1.70:"))
```

A estos dos tipos de datos fundamentales (int y float) se suma un tipo de dato muy utilizado que son las cadenas de caracteres.

Una cadena de caracteres está compuesta por uno o más caracteres. También podemos iniciar una cadena de caracteres por asignación o ingresarla por teclado.

Inicialización de una cadena por asignación:

```
#definición e inicio de una cadena de caracteres
dia="lunes"
```

Igual resultado obtenemos si utilizamos la comilla simple:

```
#definición e inicio de una cadena de caracteres
dia='lunes'
```

Para la carga por teclado de una cadena de caracteres utilizamos la función input que retorna una cadena de caracteres:

```
nombre=input("ingrese su nombre:")
```

Problema 1:

Realizar la carga por teclado del nombre, edad y altura de dos personas. Mostrar por pantalla el nombre de la persona con mayor altura.

Programa:

```
print("Datos de la primer persona")
nombre1=input("Ingrese nombre:")
edad1=int(input("Ingrese la edad:"))
altura1=float(input("Ingrese la altura Ej 1.75:"))
print("Datos de la segunda persona")
nombre2=input("Ingrese nombre:")
edad2=int(input("Ingrese la edad:"))
altura2=float(input("Ingrese la altura Ej 1.75:"))
print("La persona mas alta es:")
if altura1>altura2:
    print(nombre1)
else:
    print(nombre2)
```

Es importante notar que cuando cargamos un entero el dato devuelto por la función input se lo pasamos a la función int que tiene por objetivo convertirlo a entero:

```
edad1=int(input("Ingrese la edad:"))
```

Cuando cargamos un valor con decimal el dato devuelto por la función input se lo pasamos a la función float que tiene por objetivo convertirlo a float:

```
altura1=float(input("Ingrese la altura Ej 1.75:"))
```

Finalmente cuando cargamos una cadena de caracteres como es en este caso el nombre de una persona la función input retorna directamente una cadena de caracteres.

```
nombre1=input("Ingrese nombre:")
```

Problema 1

Realizar la carga de dos nombres por teclado. Mostrar cual de los dos es mayor alfabéticamente o si son iguales.

Programa:

```
nombre1=input("Ingrese el primer nombre:")
nombre2=input("Ingrese el segundo nombre:")
if nombre1==nombre2:
    print("Ingreso dos nombre iguales")
else:
    if nombre1>nombre2:
        print(nombre1)
        print("es mayor alfabeticamente")
    else:
        print(nombre2)
        print("es mayor alfabeticamente")
```

Cuando trabajamos con cadenas de caracteres al utilizar el operador > estamos verificando si una cadena es mayor alfabéticamente a otra (esto es distinto a cuando trabajamos con enteros o flotantes)

Por ejemplo 'luis' es mayor a 'carlos' porque la 'l' se encuentra más adelante en el abecedario que la 'c'.

Problema 3:

Realizar la carga de enteros por teclado. Preguntar después que ingresa el valor si desea cargar otro valor debiendo el operador ingresar la cadena 'si' o 'no' por teclado. Mostrar la suma de los valores ingresados.

Programa:

```
opcion="si"
suma=0
while opcion=="si":
    valor=int(input("Ingrese un valor:"))
    suma=suma+valor
    opcion=input("Desea cargar otro numero (si/no):")
print("La suma de valores ingresados es")
print(suma)
```

Para resolver este problema hemos inicializado una variable de tipo cadena de caracteres (también se las llama variables de tipo string) con el valor "si", esto hace que la condición del while se verifique verdadera la primera vez. Dentro del while luego de cargar el valor entero se pide la carga por teclado que confirme si desea cargar otro valor, en caso que cargue el string "si" el ciclo repetitivo while se vuelve a repetir.

El ciclo se corta cuando el operador carga un string distinto a "si".

Es importante notar que el string "si" es distinto al string "Si", es decir las mayúsculas no tienen el mismo valor alfabético que las minúsculas (después veremos que podemos convertir mayúsculas a minúsculas y viceversa)

Problema propuesto

1 - Realizar la carga de dos nombres de personas distintos. Mostrar por pantalla luego ordenados en forma alfabética.

Solución al problema

```
nombre1=input("Ingrese el primer nombre:")
nombre2=input("Ingrese el segundo nombre:")
print("Listado alfabetico:")
if nombre1<nombre2:
    print(nombre1)
    print(nombre2)
else:
    print(nombre2)
    print(nombre1)
```

Variables enteras, flotantes y cadenas de caracteres

Ya hemos visto que podemos cargar una cadena de caracteres por asignación:

```
#con doble comillas
cadena1="juan"
#el resultado es igual con simple comillas
cadena2='ana'
```

También podemos cargarla por teclado:

```
nombre=input("Ingrese su nombre:")
```

Podemos utilizar los operadores relacionales para identificar si dos cadenas son iguales, distintas o cual es la mayor alfabética:

```
== Igualdad
!= Desigualdad
< menor
<= menor o igual
> mayor
>= mayor o igual
```

Como su nombre lo indica una cadena de caracteres está formada generalmente por varios caracteres (de todos modos podría tener solo un caracter o ser una cadena vacía)

Podemos acceder en forma individual a cada caracter del string mediante un subíndice:

```
nombre='juan'
print(nombre[0])    #se imprime una j
if nombre[0]=="j": #verificamos si el primer caracter del string es una j
    print(nombre)
    print("comienza con la letra j")
```

Los subíndices comienzan a numerarse a partir del cero.

Si queremos conocer la longitud de un string en Python disponemos de una función llamada len que retorna la cantidad de caracteres que contiene:

```
nombre='juan'
print(len(nombre))
```

El programa anterior imprime un 4 ya que la cadena nombre almacena 'juan' que tiene cuatro caracteres.

Problema 1:

Realizar la carga del nombre de una persona y luego mostrar el primer caracter del nombre y la cantidad de letras que lo componen.

Programa:

```
nombre=input("Ingrese su nombre:")
print("Primer caracter")
print(nombre[0])
print("Cantidad de letras del nombre:")
print(len(nombre))
```

Problema 2:

Solicitar la carga del nombre de una persona en minúsculas. Mostrar un mensaje si comienza con vocal dicho nombre.

Programa:

```
nombre=input("Ingrese su nombre:")
if nombre[0]=="a" or nombre[0]=="e" or nombre[0]=="i" or nombre[0]=="o" or
nombre[0]=="u":
    print("El nombre ingresado comienza con vocal")
else:
    print("El nombre ingresado no comienza con vocal")
```

Con que una de las condiciones del if sea verdadera luego se ejecuta el bloque del verdadero.

Problema 3:

Ingresar un mail por teclado. Verificar si el string ingresado contiene solo un caracter "@".

Programa:

```
mail=input("Ingrese un mail:")
cantidad=0
x=0
while x<len(mail):
    if mail[x]=="@":
        cantidad=cantidad+1
    x=x+1
if cantidad==1:
    print("Contiene solo un caracter @ el mail ingresado")
else:
    print("Incorrecto")
```

Para analizar cada caracter del string ingresado disponemos una estructura while utilizando un contador llamado x que comienza con el valor cero y se repetirá tantas veces como caracteres tenga la cadena (mediante la función len obtenemos la cantidad de caracteres):

```
while x<len(mail):
```

Dentro del ciclo while verificamos cada caracter mediante un if y contamos la cantidad de caracteres "@":

```
    if mail[x]=="@":
```

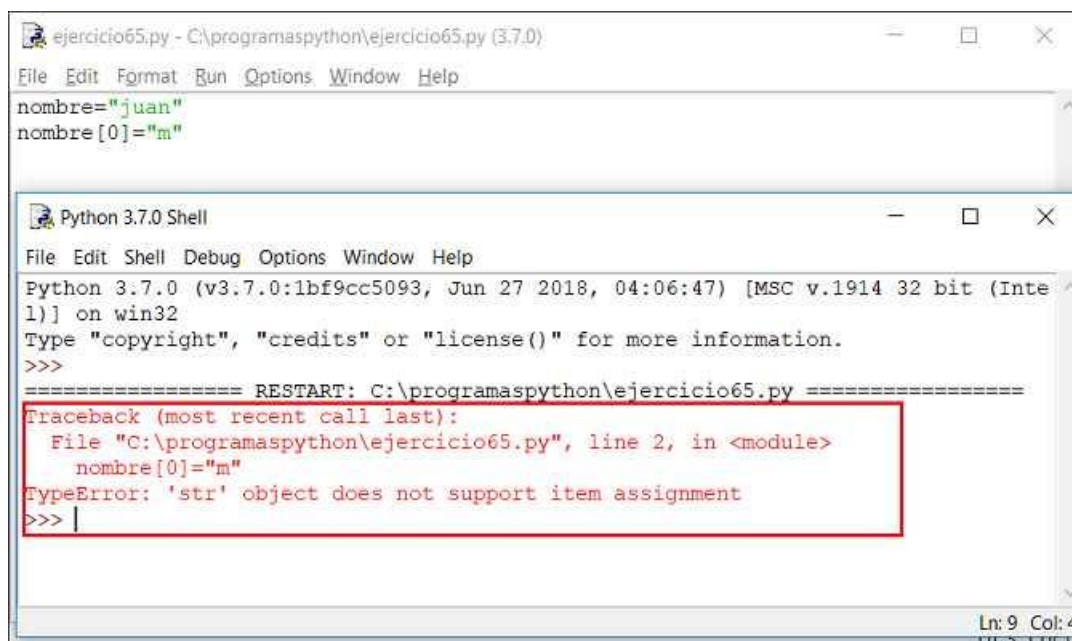
```
cantidad=cantidad+1
```

Cuando sale del ciclo while procedemos a verificar si el contador tiene almacenado el valor 1 y mostramos el mensaje respectivo:

```
if cantidad==1:  
    print("Contiene solo un caracter @ el mail ingresado")  
else:  
    print("Incorrecto")
```

Los string en Python son inmutables, esto quiere decir que una vez que los inicializamos no podemos modificar su contenido:

```
nombre="juan"  
nombre[0]="m" #esto no se puede
```



No hay que confundir cambiar parte del string con realizar la asignación de otro string a la misma variable, luego si es correcto asignar otro valor a un string:

```
nombre="juan"  
print(nombre)  
nombre="ana"  
print(nombre)
```



The screenshot shows a Python 3.7.0 IDE window titled 'ejercicio65.py - C:\programaspython\ejercicio65.py (3.7.0)'. The script contains the following code:

```
nombre="juan"
print(nombre)
nombre="ana"
print(nombre)
```

The Python 3.7.0 Shell window shows the execution output:

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\programaspython\ejercicio65.py =====
juan
ana
>>> |
```

The status bar at the bottom right indicates 'Ln: 7 Col: 4'.

Métodos propios de las cadenas de caracteres.

Los string tienen una serie de métodos (funciones aplicables solo a los string) que nos facilitan la creación de nuestros programas.

Los primeros tres métodos que veremos se llaman: lower, upper y capitalize.

- upper() : devuelve una cadena de caracteres convertida todos sus caracteres a mayúsculas.
- lower() : devuelve una cadena de caracteres convertida todos sus caracteres a minúsculas.
- capitalize() : devuelve una cadena de caracteres convertida a mayúscula solo su primer caracter y todos los demás a minúsculas.

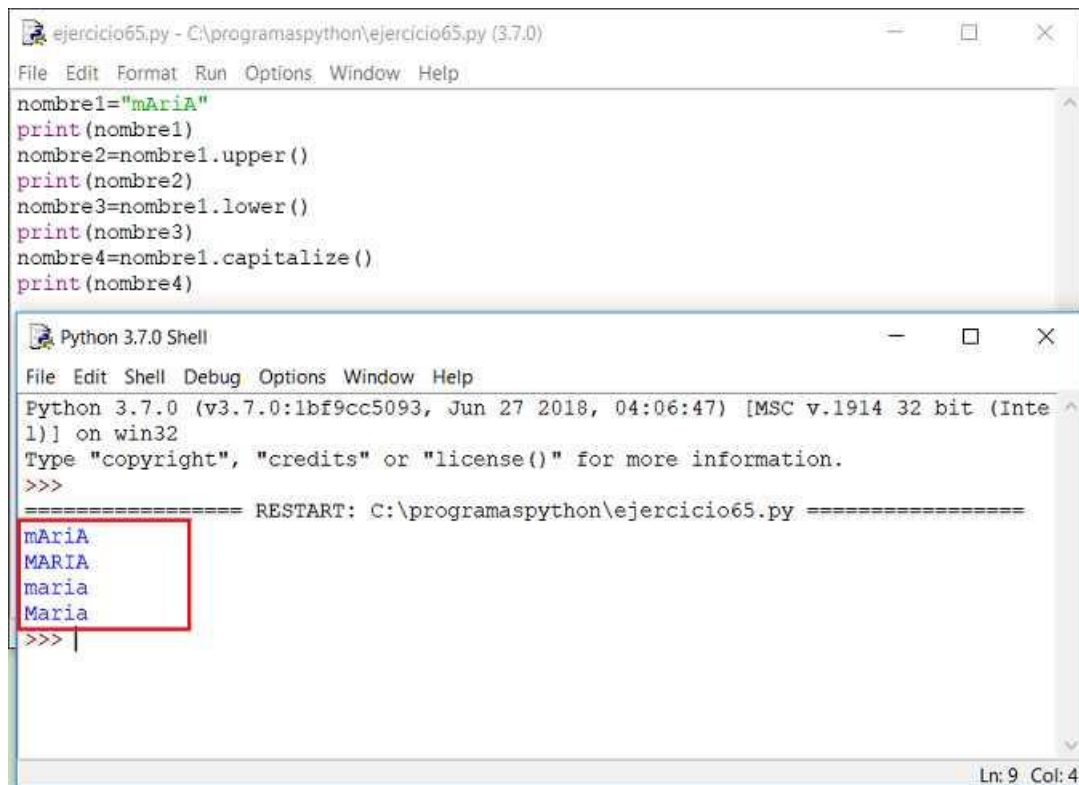
Problema 4:

Inicializar un string con la cadena "mAriA" luego llamar a sus métodos upper(), lower() y capitalize(), guardar los datos retornados en otros string y mostrarlos por pantalla.

Programa:

```
nombre1="mAriA"
print(nombre1)
nombre2=nombre1.upper()
print(nombre2)
nombre3=nombre1.lower()
print(nombre3)
nombre4=nombre1.capitalize()
print(nombre4)
```

El resultado de ejecutar este programa es:



The image shows a screenshot of a Python 3.7.0 IDE. The top window, titled 'ejercicio65.py - C:\programaspython\ejercicio65.py (3.7.0)', contains the following code:

```
nombre1="mArIA"
print(nombre1)
nombre2=nombre1.upper()
print(nombre2)
nombre3=nombre1.lower()
print(nombre3)
nombre4=nombre1.capitalize()
print(nombre4)
```

The bottom window, titled 'Python 3.7.0 Shell', shows the output of the script. The first four lines of output are highlighted with a red box:

```
>>>
===== RESTART: C:\programaspython\ejercicio65.py =====
mArIA
MARIA
maria
Maria
>>> |
```

The status bar at the bottom right of the shell window indicates 'Ln: 9 Col: 4'.

Para llamar a un método del string debemos disponer entre el nombre del string y el método el caracter punto:

```
nombre2=nombre1.upper()
```

Es importante decir que el string nombre1 no se modifica su contenido (recordar que un string es inmutable) pero el método upper() retorna el contenido de la variable nombre1 para convertida a mayúsculas. El dato devuelto se almacena en la variable nombre2.

Problemas propuestos

Ha llegado nuevamente la parte fundamental, que es el momento donde uno desarrolla individualmente un algoritmo para la resolución de un problema.

1- Cargar una oración por teclado. Mostrar luego cuantos espacios en blanco se ingresaron. Tener en cuenta que un espacio en blanco es igual a " ", en cambio una cadena vacía es ""

2 - Ingresar una oración que pueden tener letras tanto en mayúsculas como minúsculas. Contar la cantidad de vocales. Crear un segundo string con toda la oración en minúsculas para que sea más fácil disponer la condición que verifica que es una vocal.

3 - Solicitar el ingreso de una clave por teclado y almacenarla en una cadena de caracteres. Controlar que el string ingresado tenga entre 10 y 20 caracteres para que sea válido, en caso contrario mostrar un mensaje de error.

Soluciones a los problemas

Esta sección solo se debería leer luego de haber intentado por un largo tiempo la resolución en forma personal de los problemas propuestos.

```
oracion=input("Ingrese una oracion:")
cantidad=0
x=0
while x<len(oracion):
    if oracion[x]==" ":
        cantidad=cantidad+1
    x=x+1
print("La cantidad de espacios en blanco ingresado son")
print(cantidad)
```

```
oracion=input("Ingrese una oracion:")
oracionmin=oracion.lower()
cantidad=0
x=0
while x<len(oracionmin):
    if oracionmin[x]=="a" or oracionmin[x]=="e" or oracionmin[x]=="i" or
oracionmin[x]=="o" or oracionmin[x]=="u":
        cantidad=cantidad+1
    x=x+1
print("La cantidad de vocales de la oracion son")
print(cantidad)
```

```
clave=input("Ingrese una clave que tenga entre 10 y 20 caracteres:")
if len(clave)>=10 and len(clave)<=20:
    print("Largo correcto")
else:
    print("Largo incorrecto")
```