DESARROLLO WEB FULLSTACK

con Python y JavaScript

polotic misiones

ESTRUCTURAS DE DATOS EN PYTHON

CONCEPTO DE VARIABLE

ALMACENAMIENTO DE INFORMACIÓN

Una aplicación requiere un acceso rápido a la información o, de lo contrario, llevará mucho tiempo completar las tareas. Como resultado, las aplicaciones almacenan información en la memoria.

Sin embargo, la memoria es temporal. Cuando apagas la máquina, la información debe almacenarse de alguna forma permanente, como en su disco duro, una unidad flash USB (Universal Serial Bus) o un Tarjeta Secure Digital (SD). Además, también debe considerar la forma de la información, por ejemplo, si es un número o un texto.

Una variable es una especie de caja de almacenamiento. Siempre que desees trabajar con la información, acceda a ella mediante la variable. Cambiar información significa acceder primero a la variable y luego almacenar el nuevo valor en la variable. Así como almacena cosas en cajas en el mundo real, también almacena cosas en variables (una especie de caja de almacenamiento) cuando trabaja con aplicaciones.

CONCEPTO DE VARIABLE

ALMACENAMIENTO DE INFORMACIÓN

Las computadoras están bastante ordenadas. Cada variable almacena solo un dato. El uso de esta técnica facilita la búsqueda de la información particular que necesita, a diferencia de lo que ocurre en su armario, donde se podrían esconder cualquier tipo de cosas, desde ropa hasta elementos de jardinería.

Además de guardar solo un tipo de dato tambien estas “cajas” (al igual que en la vida real) tienen un límite de cuanto se puede guardar.

Cada tipo de dato tendrá un limite definido por la arquitectura de sistemas operativos, hardware o demás de la computadora donde estas trabajando.

VARIABLES NUMÉRICAS

ENTEROS

Cualquier número natural es un número `entero`. Por ejemplo, el valor `1` es un número entero, por lo que es un número entero. Por otro lado, `1.0` no es un número entero; tiene una parte decimal, por lo que no es un número entero. Los enteros están representados por el tipo de datos `int`. También el `0` (cero) y los números negativos son enteros.

Al igual que con las cajas de almacenamiento, las variables tienen límites de capacidad. Si intenta guardar un valor demasiado grande en una caja de almacenamiento, se producirá un error. En la mayoría de las plataformas, puede almacenar números entre `–9223372036854775808` y `9223372036854775807` dentro de un `int` (que es el valor máximo que cabe en una variable de 64 bits). Aunque es un número realmente grande, no es infinito.

VARIABLES NUMÉRICAS

DISTINTAS BASES NUMERICAS

- ✓ `Base 2`: usa solo 0 y 1 como números.
- ✓ `Base 8:` usa los números del 0 al 7.
- ✓ `Base 10`: Utiliza el sistema numérico habitual.
- ✓ `Base 16:` Se denomina Hexadecimal y utiliza los números del 0 al 9 y las letras de la A a la F para crear 16 valores posibles diferentes.

Para decirle a Python cuándo usar bases distintas a la base 10, agrega un 0 y una letra especial al número.
Por ejemplo, 0b100 es el valor uno-cero-cero en base 2.

Estas son las letras que usa normalmente:

- ✓ `b: Base 2`
- ✓ `o: Base 8`
- ✓ `x: Base 16`

VARIABLES NUMÉRICAS

DISTINTAS BASES NUMERICAS

También es posible convertir valores numéricos a otras bases usando los comandos

- `bin()`
- `oct()`
- `hex()`

El uso de una base diferente en realidad facilita las cosas en muchas situaciones particulares.
Por ahora, todo lo que necesita saber es que los números enteros admiten diferentes bases numéricas.

VARIABLES NUMÉRICAS

PUNTO FLOTANTE

Cualquier número que incluya una parte decimal es un valor de punto flotante.

Por ejemplo, `1.0` tiene una parte decimal, por lo que es un valor de punto flotante.

Python almacena valores de punto flotante en el tipo de datos `float`.

Los valores de punto flotante tienen una ventaja sobre los valores enteros en el sentido de que puede almacenar valores inmensamente grandes o increíblemente pequeños en ellos.

Al igual que con las variables enteras, las variables de punto flotante tienen limitaciones en almacenamiento. El valor máximo que puede contener una variable es $\pm 1.7976931348623157 \times 10^{308}$ y el valor mínimo que puede contener una variable es $\pm 2.2250738585072014 \times 10^{-308}$ en la mayoría de plataformas.

VARIABLES NUMÉRICAS

PUNTO FLOTANTE

Al trabajar con valores de punto flotante, puedes asignar la información a la variable de varias formas. Los dos métodos más comunes son *proporcionar el número directamente* y utilizar la *notación científica*. Cuando se usa la notación científica, una *e* separa el número de su exponente.

- `Test = 255.05`
- `Test = 2.55e2`
- `Test = 2.55e-2`

VARIABLES NUMÉRICAS

NUMEROS COMPLEJOS

Puede que recuerdes o no los números complejos de la escuela. Un número complejo consta de un número ***real*** y un número ***imaginario*** que están emparejados. Python es uno de los pocos lenguajes que proporciona un tipo de datos integrado para admitirlos.

En caso de que se haya olvidado por completo de los números complejos, puede leer sobre ellos en: http://www.mathsisfun.com/numbers/complex-numbers.html.
Los usos del mundo real para números complejos incluyen:

- ✓ Ingeniería eléctrica
- ✓ Dinámica de fluidos
- ✓ Mecánica cuántica
- ✓ Gráficos de computadora
- ✓ Sistemas dinámicos

VARIABLES NUMÉRICAS

NUMEROS COMPLEJOS

La parte imaginaria de un número complejo siempre aparece con una *j* después.
Entonces, si desea crear un número complejo con 3 como parte real y 4 como parte imaginaria, puedes hacer una asignación como esta:

```
miComplejo = 3 + 4j
```

Si deseas ver la parte real de la variable, simplemente escribes `miComplejo.real` en el interprete de Python y presionas Enter. Del mismo modo, si deseas ver la parte imaginaria de la variable, escribes `miComplejo.imag` en el interprete de Python y presionas Enter.

COMPRENDIENDO VALORES LOGICOS

BOOLEAN

Puede parecer increíble, ¡pero las computadoras siempre te dan una respuesta directa!
Una computadora nunca proporcionará "tal vez" como salida.

Cada respuesta que obtienes es Verdadera (`True`) o Falsa (`False`).

De hecho, existe una rama completa de las matemáticas llamada álgebra de Boole que originalmente fue definida por George Boole (un super-geek de su tiempo) en la que se basan las computadoras para tomar decisiones. Contrariamente a la creencia común, el álgebra de Boole existe desde 1854, mucho antes de la época de las computadoras.

COMPRENDIENDO VALORES LOGICOS

BOOLEAN

CONCEPTO IMPORTANTE

Cuando usas un valor booleano en Python, lo defines mediante el tipo `bool`.

Una variable de este tipo solo puede contener dos valores: True o False.
Puedes asignar un valor utilizando las palabras reservadas `True` o `False`, o puedes crear una expresión que defina una idea lógica que equivale a verdadero o falso.

Por ejemplo, podría decir `miBoolean = 1 > 2`, lo que equivaldría a `False` porque definitivamente 1 no es mayor que 2.

Verás que el tipo bool se usa ampliamente durante todo el transcurso del curso, así que debes tenerlo en cuenta.

COMPRENDIENDO CADENAS

STRINGS

De todos los tipos de datos, las cadenas de caracteres (strings) son las que los humanos comprenden con mayor facilidad y las computadoras no las comprenden en absoluto.

Una cadena es simplemente cualquier grupo de caracteres que coloques entre comillas dobles. Por ejemplo, `miString = "Python es un gran lenguaje"` asigna una cadena de caracteres a `miString`.

La computadora no ve letras en absoluto. Cada letra que usa está representada por un número en la memoria. Por ejemplo, la letra A es en realidad el número 65. Para ver esto por ti mismo, puedes escribir `ord ("A")` en el interprete de Python y presiona Enter.

COMPRENDIENDO CADENAS

STRINGS

Debido a que la computadora no comprende realmente los strings, pero son tan útiles para escribir aplicaciones, a veces es necesario convertir un string en un número.

Puedes utilizar los comandos `int()` y `float()` para realizar esta conversión.

Por ejemplo, si escribes `miEntero = int("123")` y presionas Enter en el interprete de Python, creas un `int` llamado `miEntero` que contiene el valor `123`.

También puede convertir números en un string utilizando el comando `str()`. Por ejemplo, si escribes `miString = str(1234.56)` y presionas Enter, creas un string que contiene el valor `"1234.56"` y la asignas a `miString`.

El punto es que puedes ir y venir entre strings y números con gran facilidad.

COMPRENDIENDO CADENAS

METODOS ÚTILES

Mayusculas
```
>>> my_string.upper()
```
Minúsculas
```
>>> my_string.lower()
```
Cantidad de ocurrencia de un elemento
```
>>> my_string.count('w')
```
Reemplazar un elemento con otro
```
>>> my_string.replace('e', 'i')
```
Eliminar espacios en blanco
```
>>> my string.strip()
```

DETERMINANDO TIPOS

¿Cómo saber de que tipo es una variable?

A veces, es posible que desees conocer el tipo de variable.

Quizás el tipo no sea obvio en el código o hayas recibido la información de una fuente cuyo código no es accesible.

Siempre que desees ver el tipo de una variable, utilizas el método `type()`. Por ejemplo, si comienzas colocando un valor de `5` en `miEntero` escribiendo `miEntero = 5` y presionando Enter, puedes encontrar el tipo de `miEntero` escribiendo `type(miEntero)` y presionando Enter.
La salida será `<class 'int'>,` lo que significa que `miEntero` contiene un valor de tipo `int`.

PERDIDOS EN EL TIEMPO

TRABAJANDO CON FECHAS Y HORAS

Las fechas y horas son elementos con los que la mayoría de la gente trabaja bastante. La sociedad basa casi todo en la fecha y hora en que una tarea debe estar o se completó.

Hacemos citas y planificamos eventos para fechas y horarios específicos. La mayor parte de nuestro día gira las 24 horas del día.

Debido a la naturaleza de los humanos orientada al tiempo, es una buena idea observar cómo Python maneja la interacción con fechas y horas (especialmente almacenando estos valores para su uso posterior). Como ocurre con todo lo demás, las computadoras solo comprenden números: la fecha y la hora no existen realmente.

PERDIDOS EN EL TIEMPO

TRABAJANDO CON FECHAS Y HORAS

Para usar fechas y horas, debes escribir un comando especial denominado `import datetime`. Técnicamente, este acto se llama importar un módulo (reusar código de un tercero).

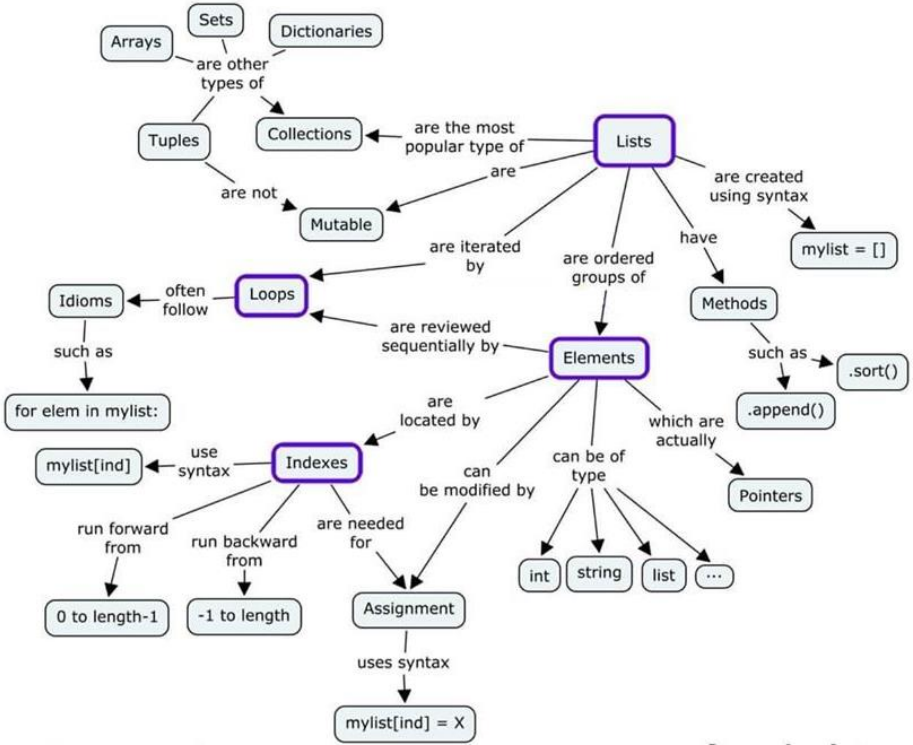
Para obtener la hora actual, simplemente escribes `datetime.datetime.now()` y presione Enter.

Verás la información completa de fecha y hora que se encuentra en el reloj de tu computadora.

Es posible que hayas notado que la fecha y la hora son un poco difíciles de leer en el formato existente. Diga que desea obtener solo la fecha actual, en un formato legible.

Es hora de combinar algunas cosas que descubrimos anteriormente para lograr esta tarea. Escribes `str(datetime.datetime.now().date())` y presione Enter.

¡HAY MUCHO, MUCHO MÁS!



OPERACIONES

OPERACIONES ARITMÉTICAS

Suma

```
>>> x+2
 7
```

Resta

```
>>> x-2
 3
```

Multiplicación

```
>>> x*2
 10
```

Potencia/Exponente

```
>>> x**2
 25
```

Modulo de una división

```
>>> x%2
 1
```

División

```
>>> x/float(2)
 2.5
```

TRABAJEMOS JUNTOS

1. Escribe un programa Python que acepte el radio de un círculo del usuario y calcule el área.
2. Escribe un programa Python que acepte un número entero (n) y calcule el valor de n + nn + nnn
3. Escribe un programa en Python que acepte una cadena de caracteres y cuente el tamaño de la cadena y cuantas veces aparece la letra A (mayuscula y minúscula)
4. Escribe un programa en Python que muestre la hora actual con una suma de dos horas adicionales

¡HASTA LA *próxima!*

www.polotic.misiones.gob.ar

/poloticmisiones

polotic misiones

Gobierno de Misiones