**Módulo 2:**

**Python.org** tiene un consejo directivo que ordena y guia el desarrollo del interprete Python

Por otro lado hay una **comunidad de programadores** que aporta y genera reglas para el trabajo en común.

**➢** **Guardar datos en memoria.**

En Python, un objeto es en parte un contenedor que se utiliza para almacenar datos.

Se puede pensar como una caja etiquetada donde puedes guardar diferentes tipos de información, como números, cadenas de texto o listas.

Algunos conceptos clave sobre los objetos variables en Python son los siguientes:

No es necesario declarar explícitamente el tipo de una variable antes de usarla.

Simplemente puedes asignar un valor a una variable utilizando el operador de asignación "="esto guarda en memoria un dato asociado al nombre del objeto y el interprete chequea el tipo de dato y determina que tipo de objeto es.

edad = 25 guarda en memoria el valor 25 asignándole una etiqueta con el nombre edad, este objeto sera del tipo int - entero

pi = 3.14159 guarda en memoria el valor 3.14159 asignándole una etiqueta con el nombre pi, este objeto sera del tipo float – con punto decimal

Cada objeto tiene un nombre único, dependiendo del tipo de objeto su asociación a un dato en guardado en memoria puede variar en el transcurso del script. Esto permite que el programa pueda acceder y manipular los valores y los tipos durante su ejecución.

|  |
| --- |
| Los objetos variables NO son espacios de memoria reservados para almacenar datos. |

Por ejemplo, para declarar una variable entera llamada "edad", se utilizaría la siguiente línea de **código:**

|  |
| --- |
| Se recomienda el uso de print (f"descripción : {objeto} ") |

·

|  |
| --- |
| # no se requiere declarar int edad como en C  edad = 25  print (f"La edad guardada es de {edad} años") |

Salida esperada por consola

|  |
| --- |
| La edad guardada es de 25 años |

En Python algunos objetos con un dato son de los siguientes tipos:

int: Representa números enteros.

dato = 8

float: Representa números de punto flotante (números con decimales).

Dato = 8.5

str: Representa una cadena de caracteres. Cada Eslabón es un carácter aunque

puede haber strings con un solo carácter e incluso sin ninguno.

Siempre se coloca entre comillas (simples, dobles o tres juegos de alguna

para multilineas)

dato = "Hola mundo IT"

dato = """ esto

es

un

texto

de varias

lineas """

dato = "T"

dato = ""

bool: Representa valores booleanos (True verdadero o False falso).

dato = True

dato = False

Formato antiguo:

|  |
| --- |
| objeto = "hola mundo Python"  print (objeto) |

Salida esperada por consola

|  |
| --- |
| hola mundo Python |

·

Formato moderno:

objeto imprime el contenido en memoria guardado con la etiqueta objeto

|  |
| --- |
| objeto = "hola mundo Python"  print (f"el contenido del objeto es {objeto} ") |

Salida esperada por consola

|  |
| --- |
| el contenido del objeto es hola mundo Python |

·

objeto= imprime el nombre del objeto y el contenido en memoria guardado con la etiqueta objeto

|  |
| --- |
| objeto = "hola mundo Python"  print (f"el contenido del objeto es {objeto=} ") |

Salida esperada por consola

|  |
| --- |
| el contenido del objeto es objeto='hola mundo Python' |

·

Mas adelante veremos algunos objetos con múltiples datos: listas (`list`), tuplas (`tuple`), diccionarios (`dict`), entre otros.

Después de declarar una variable, se puede asignar un valor utilizando el operador de asignación =

**➢ Operador de asignación igual (=)**

Para guardar un dato en memoria debemos previamente haber declarado el nombre de la variable con el propósito que tendrá. Ahora veremos como se asigna y que reglas hay

|  |
| --- |
| Luego veremos el resto de los operadores de asignación el la Módulo de bucles |

lado\_recibe = lado\_dato

En el lado que recibe debemos poner algún nombre de fantasía que represente el dato a ingresar.

Las reglas del lenguaje exijan que inicie con letras, puede usarse el signo guin de abajo (\_) y números pero nunca al inicio.

La comunidad te sugiere que los nombres se relacionen con el dato a guardar.

En Python, una variable es un contenedor que se utiliza para almacenar datos. Puedes pensar en una variable como una caja etiquetada donde puedes guardar diferentes tipos de información, como números, cadenas de texto o listas.

**➢ Variables y Constantes en Python Mmmmmmm………..**

Algunos conceptos clave sobre los objetos variables en Python son los siguientes:

**➢ Declaración de objetos variables:** En Python, NO es necesario declarar explícitamente el tipo de una variable antes de usarla.

**➢ Asignación de valores:** Para asignar un valor a una variable, utilizas el operador de asignación "=" seguido del valor que deseas asignar.

edad = 25

nombre = "Juan"

**➢ Nombres de objetos:**

Los nombres de los objetos en Python deben seguir ciertas reglas para el interprete Python

Solo pueden contener letras, dígitos y guiones bajos.

Deben comenzar con una letra o un guion bajo (₎

Python distingue entre mayúsculas y minúsculas en los nombres de los objetos.

Los nombres de los objetos en Python deben seguir ciertas reglas para la comunidad Python

**Estilo general**

* snake\_case, donde las palabras se separan con guiones bajos
* minúsculas para objetos mutables, variables, que cambian en el trascurso del script.
* mayúsculas para objetos que si bien pueden mutables, variables, que cambian en el trascurso del script el programador marca en caracteres mayúsculas para marcar su contenido como constante y que no debería ser modificado.

**Estilo de módulos**

* pascal\_case Primer carácter en mayúsculas o Capitalize
* Es importante seguir estas convenciones para que el código sea más legible y coherente.

Ejemplos validos para el interprete

nombre = "Pedro"# Valido para el interprete Python y la comunidad Python

nombre\_1 = "Pedro"# Valido para el interprete Python y la comunidad Python

nombre\_alumno = "Pedro"# Valido para el interprete Python y la comunidad Python

nombreAlumno = "Pedro"# Valido para el interprete Python y la comunidad Python

\_nombre = "María"# Valido para el interprete Python pero NO la comunidad Python

Nombre = "Pedro"# Valido para el interprete Python pero NO la comunidad Python

xdcfvgb = 9# Valido para el interprete Python pero NO la comunidad Python

En Python, por lo general, los valores se pisan, se sobre escriben o la etiqueta o nombre se asigna a otro dato

**- Reasignación de valores:** Puedes cambiar el valor de una variable simplemente asignándole un nuevo valor. Python permite reasignar una variable a diferentes tipos de datos a lo largo del programa.

valor = 25

valor = 30 # Cambio el valor de la objeto variable

valor = 3.55 # Cambio el valor y tipo de la objeto variable

Puedes utilizar objetos variables en expresiones matemáticas o concatenarlas con cadenas de texto utilizando los operadores correspondientes.

x = 10

y = 5

suma = x + y

mensaje = "El resultado es: " + str(suma) # modo antiguo cambiando

el entero (10+5=15)

a str(15) = "15"

mensaje = f"El resultado es: {suma}" # modo actual

Las objetos variables en Python te permiten almacenar y manipular datos de manera flexible en tus programas.

Puedes usarlas para realizar cálculos, almacenar resultados intermedios, guardar información del usuario y mucho más.

|  |
| --- |
| nombre = "Juan"  apellido = "Pérez"  nombre\_completo = nombre + apellido  print (f " el nombre completo es {nombre\_completo } ")  nombre\_completo\_con\_espacio = nombre + " " +apellido  print (f " el nombre completo con espacios es {nombre\_completo\_con\_espacio} ") |

Salida esperada por consola

|  |
| --- |
| el nombre completo es JuanPérez  el nombre completo con espacios es Juan Pérez |

|  |
| --- |
| Se recomienda el uso de  print (f"descripción : {**objeto**} ") print (f"tipo de objeto : {**type(objeto)**} ") |

Es importante tener en cuenta que en Python los objetos deben existir antes de que se utilicen en el código. Esto significa no puedo imprimir un objeto antes de que este sea creado.  
Para crearlo debo asignar al nombre un valor (y por ende un tipo)

En relación a los objetos numéricos int, float o complex no se tendrá en cuenta el tamaño, valor o grado de exactitud o definición de cantidad de decimales.

Noten que dije punto decimal y no coma. En Argentina rige una ley (SiMeLA) sistema métrico legal argentino donde se declara que la parte entera de la decimal se separa por una coma.  
A los lenguajes de programación no les importa mucho esta ley y para separar la parte decimal de la entera se usa un punto. NO hay separador de miles sino que se escriben de corrido.

cien\_mil= 100000; # no ~~100.000~~

PI=3.14159; # no 3,14159

**➢ Ámbitos de los objetos variables:**

Las variables en Python tienen un alcance, es decir, la parte del programa donde son accesibles y válidas.

|  |
| --- |
| Los ámbitos de los distintos objetos se vera luego de ver funciones propias.  La variable declarada dentro de una función solo es accesible dentro de esa función (a menos que se utilicen mecanismos como global o nonlocal para ampliar su alcance).  Las variables declaradas fuera de una función tienen un alcance global y son accesibles en todo el programa. |

**➢ Expresiones:**

En el lenguaje de programación Python, los operadores son símbolos que se utilizan para realizar operaciones matemáticas, de comparación y de asignación.

**➢ Operadores aritméticos en Python:**

Estos operadores se utilizan para realizar operaciones matemáticas básicas en valores numéricos.

|  |  |
| --- | --- |
| **operadores** | **Funciónes estándar entre operandos** |
| **+** | Suma - adiciona dos operandos. |
| **-** | Resta - sustracción al valor del operando de la izquierda el valor del de la derecha.  Cambia el signo sobre un único operador. |
| **\*** | Producto - Multiplicación de dos operandos. |
| **/** | Divide el operando de la izquierda por el de la derecha.  La salida siempre es un float (genera un casting que veremos mas adelante) |
| **//** | División entera se obtiene el cociente 'entero' de dividir el operando de la izquierda por el de la derecha. |
| **%** | Módulo es el residuo que se obtiene el resto de dividir el entero del operando de la izquierda por el de la derecha. |
| **operadores** | **Funciónes Python entre operandos** |
| **\*\*** | Potencia - Exponenciación eleva el operando de la izquierda a la potencia del operador del de la derecha. |
| **\*\***(1/n) | Radicación eleva el operando de la izquierda a la potencia de 1 (uno) dividido el valor operador del de la derecha. |

* En Python ademas de los estándar suma(+), resta (-), división estándar (/), división entera (//) , resto de división (%) , multiplicación (\*)
* ademas

Potenciación o exponenciación (\*\*) x\*\*2 x al cuadrado

Radicación (\*\*(1/r) x\*\*(1/2) raiz cuadrada de x

**➢ Comentarios en Python**

Una linea # comentario

Multilineas – un objeto string sin nombre por lo que no se asigna a memoria

""" comentario

multi

linea """

**➢Tipos de datos que se deben declarar en las variables a utilizar .**

|  |
| --- |
| nota =9;  print (f"la nota es {nota} ")  puts("Felicitaciones!") |

Salida esperada por consola

|  |
| --- |
| la nota es 9  Felicitaciones! |

·

|  |
| --- |
| Podremos generar varios ejemplos para cálculos de volumen, áreas, y diversas formulas que se requieren en el uso diario.  Calcule el iva de un producto de ferretería 21%, elementos de electrónica o computación 10.5%, y sin iva (0%) como algunos elementos de la canasta básica. |

·

|  |
| --- |
| nota\_1er\_p = 9  nota\_2do\_p = 8  nota\_3er\_p = 7  nota\_4to\_p = 6  suma = nota\_1er\_p + nota\_2do\_p + nota\_3er\_p + nota\_4to\_p  media= suma/4  print (f"la media de las notas es {media} puntos / 10") |

Salida esperada por consola

|  |
| --- |
| la media de las notas es nota es 7.5 puntos / 10 |

·

|  |
| --- |
| Se recomienda agregar al uso de  print (f"descripción : {**objeto=**} ")  # descripción permite una comprensión del contenido del objeto print (f"tipo de objeto : {**type(objeto)=**} ")  # type muestra el tipo asignado por el interprete según dato |

·

|  |
| --- |
| nombre="Juan Pérez"  nota\_1er\_p = 9  nota\_2do\_p = 8  nota\_3er\_p = 7  nota\_4to\_p = 6  suma = nota\_1er\_p + nota\_2do\_p + nota\_3er\_p + nota\_4to\_p  media= suma/4  print (f"nombre la media de las notas es {media} puntos / 10")  print (f"suma= tipo de objeto :{ type(suma)=} ") print (f"media= tipo de objeto :{ type(media)=} ") |

Salida esperada por consola

|  |
| --- |
| Juan Pérez la media de las notas es 7.5 puntos / 10  suma=30 tipo de objeto : type(suma) =<class 'int'>  media=7.5 tipo de objeto : type(media)=<class 'float'> |

|  |
| --- |
| Para poder mostrar que es un método en un objeto y que los distintos objetos tienen en general distintos métodos se agrega dir  Se recomienda agregar al uso de  print (f"descripción : {**objeto=**} ")  # descripción permite una comprensión del contenido del objeto print (f"tipo de objeto : {**type(objeto)=**} ")  # type muestra el tipo asignado por el interprete según dato  print (f"métodos del tipo de objeto : {**dir(objeto)=**} ")  # muestra los métodos del tipo de objeto asignado por el interprete según dato |

**·**

|  |
| --- |
| **Atributos:** Los atributos son las variables que pertenecen a un objeto. Pueden ser variables de instancia (definidas dentro de un método) o variables de Módulo (definidas en la Módulo pero fuera de los métodos). Cada instancia de una Módulo puede tener diferentes valores para los atributos.  **Métodos:** Los métodos son las funciones asociadas a una módulo. Pueden realizar operaciones en los datos del objeto y pueden acceder y modificar los atributos. Los métodos pueden ser llamados en las instancias de la Módulo y actúan en contexto a los datos específicos de la instancia. |

**Métodos:**

En los siguientes ejemplos empezaremos a usar algunos métodos de los objetos.

|  |
| --- |
| nombre="Juan Pérez"  nota\_1er\_p = 9  nota\_2do\_p = 8  nota\_3er\_p = 7  nota\_4to\_p = 6  suma = nota\_1er\_p + nota\_2do\_p + nota\_3er\_p + nota\_4to\_p  media= suma/4  print (f"nombre la media de las notas es {media puntos / 10")  print (f"{nombre=} tipo de objeto : {type(nombre)= ")  print (f"{suma=} tipo de objeto : {type(suma)= ")  print (f"{media= } tipo de objeto : {type(media)= ")  print (f"{type(nombre)=} \n métodos de objeto : {dir(nombre)=} ")  print (f"{type(suma)=} \n métodos de objeto : {dir(suma)=} ")  print (f"{type(media)=} \n métodos de objeto : {dir(media)=} ") |

Salida esperada por consola

|  |
| --- |
| Juan Pérez la media de las notas es 7.5 puntos / 10  nombre='Juan Pérez' tipo de objeto : type(nombre)=<class 'str'>  suma=30 tipo de objeto : type(suma)=<class 'int'>  media=7.5 tipo de objeto : type(media)=<class 'float'>  type(nombre)=<class 'str'>  metodos de objeto : dir(nombre)=['\_\_add\_\_', '\_\_class\_\_', '\_\_contains\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getitem\_\_', '\_\_getnewargs\_\_', '\_\_getstate\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_iter\_\_', '\_\_le\_\_', '\_\_len\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasshook\_\_', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format\_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'removeprefix', 'removesuffix', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']  type(suma)=<class 'int'>  metodos de objeto : dir(suma)=['\_\_abs\_\_', '\_\_add\_\_', '\_\_and\_\_', '\_\_bool\_\_', '\_\_ceil\_\_', '\_\_class\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_divmod\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_float\_\_', '\_\_floor\_\_', '\_\_floordiv\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getnewargs\_\_', '\_\_getstate\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_index\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_int\_\_', '\_\_invert\_\_', '\_\_le\_\_', '\_\_lshift\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_neg\_\_', '\_\_new\_\_', '\_\_or\_\_', '\_\_pos\_\_', '\_\_pow\_\_', '\_\_radd\_\_', '\_\_rand\_\_', '\_\_rdivmod\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rfloordiv\_\_', '\_\_rlshift\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_ror\_\_', '\_\_round\_\_', '\_\_rpow\_\_', '\_\_rrshift\_\_', '\_\_rshift\_\_', '\_\_rsub\_\_', '\_\_rtruediv\_\_', '\_\_rxor\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_sub\_\_', '\_\_subclasshook\_\_', '\_\_truediv\_\_', '\_\_trunc\_\_', '\_\_xor\_\_', 'as\_integer\_ratio', 'bit\_count', 'bit\_length', 'conjugate', 'denominator', 'from\_bytes', 'imag', 'numerator', 'real', 'to\_bytes']  type(media)=<class 'float'>  metodos de objeto : dir(media)=['\_\_abs\_\_', '\_\_add\_\_', '\_\_bool\_\_', '\_\_ceil\_\_', '\_\_class\_\_', '\_\_delattr\_\_', '\_\_dir\_\_', '\_\_divmod\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_float\_\_', '\_\_floor\_\_', '\_\_floordiv\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_getformat\_\_', '\_\_getnewargs\_\_', '\_\_getstate\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_init\_subclass\_\_', '\_\_int\_\_', '\_\_le\_\_', '\_\_lt\_\_', '\_\_mod\_\_', '\_\_mul\_\_', '\_\_ne\_\_', '\_\_neg\_\_', '\_\_new\_\_', '\_\_pos\_\_', '\_\_pow\_\_', '\_\_radd\_\_', '\_\_rdivmod\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_rfloordiv\_\_', '\_\_rmod\_\_', '\_\_rmul\_\_', '\_\_round\_\_', '\_\_rpow\_\_', '\_\_rsub\_\_', '\_\_rtruediv\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_sub\_\_', '\_\_subclasshook\_\_', '\_\_truediv\_\_', '\_\_trunc\_\_', 'as\_integer\_ratio', 'conjugate', 'fromhex', 'hex', 'imag', 'is\_integer', 'real'] |

<class 'str'> =

['capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format\_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'removeprefix', 'removesuffix', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']

<class 'int'> =

['as\_integer\_ratio', 'bit\_count', 'bit\_length', 'conjugate', 'denominator', 'from\_bytes', 'imag', 'numerator', 'real', 'to\_bytes']

<class 'float'>=

['as\_integer\_ratio', 'conjugate', 'fromhex', 'hex', 'imag', 'is\_integer', 'real']

<class 'bool'>=

['as\_integer\_ratio', 'bit\_count', 'bit\_length', 'conjugate', 'denominator', 'from\_bytes', 'imag', 'numerator', 'real', 'to\_bytes']

<class 'str'> =

['capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format\_map', 'index', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'removeprefix', 'removesuffix', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']

**Ejercicio I**