fidélitas
INGENIERÍA EN SISTEMAS DE COMPUTACIÓN

APRENDAMOS A PROGRAMAR

Programación Básica

Clase 02

Python

Contenido

Conceptos fundamentales

En esta primera parte de la clase se abordarán los temas fundamentales para la construcción de programas en Python. Estudiaremos los tipos de datos, identificadores, constantes, variables, las operaciones aritméticas y lógicas para terminar con los bloques de asignación de valores.

Identificadores, constantes y variables

Todas las soluciones programadas requerirán almacenar datos durante la ejecución del programa. Esta información será de utilidad para realizar cálculos, contar eventos, ser mostrada posteriormente, etc. Los programas en general utilizan dos elementos llamados **variables** y **constantes** con ese fin.

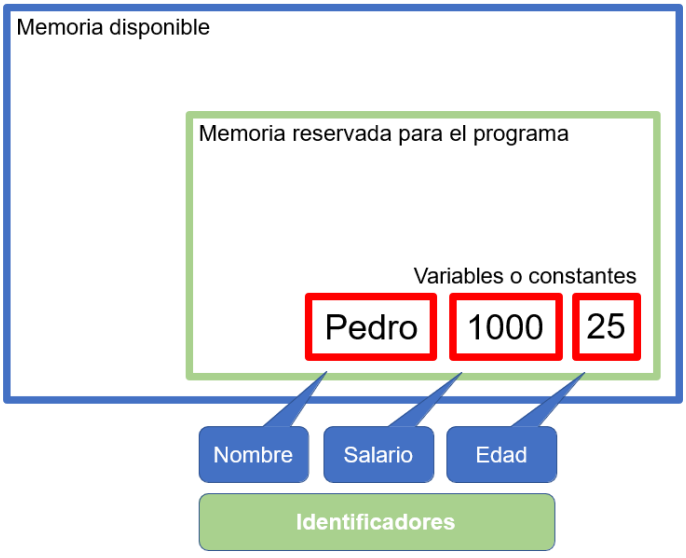
Las variables son espacios en la memoria del dispositivo electrónico en donde se almacenará algún dato. El contenido de las variables podría cambiar.

Por ejemplo, se crea una variable a la que llamaremos **total** e inicialmente tiene un cero, pero luego, como resultado de una operación aritmética podría contener el valor de 245.

A estos espacios en memoria se les asigna un nombre conocido como **identificador**, que permite reconocer este espacio dentro del programa y así poder utilizar lo que contiene.

Las constantes también son espacios en la memoria del dispositivo electrónico en donde se almacenará algún dato, a diferencia de las variables, el contenido no se cambia durante la ejecución del programa.

Por ejemplo, creamos una constante que llamaremos iva y se le asigna el valor de 0.13 y ese valor no se modifica.



Identificadores

Es importante aclarar que no se utilizan tildes ni ñ en los identificadores pero que, si en los mensajes que se van a desplegar por pantalla, este es un error muy regular.

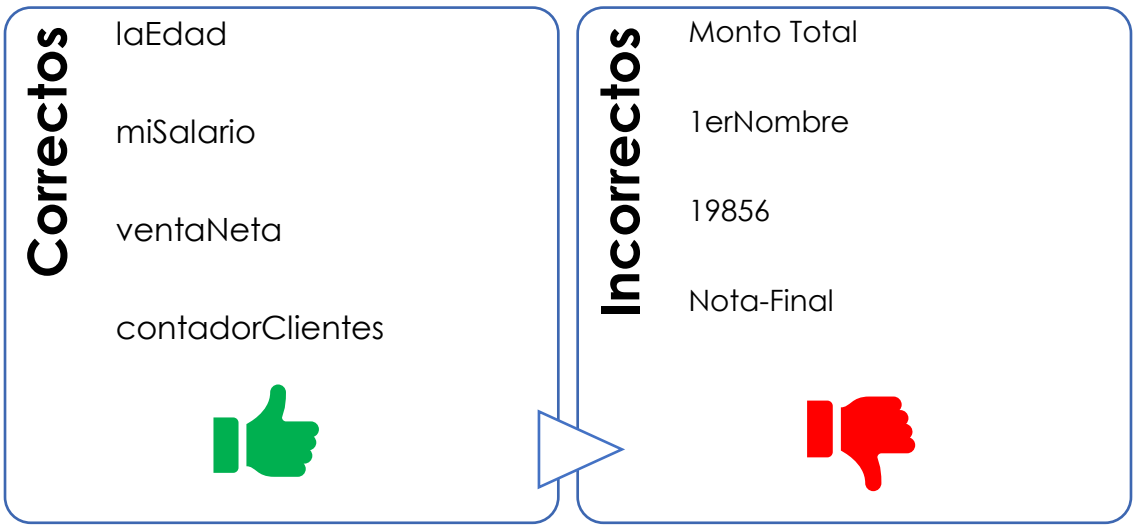
Para la creación de identificadores es necesario respetar algunas **reglas**.

- No deben contener espacios en blanco
- No deben iniciar con un número
- No deben utilizarse palabras reservadas del lenguaje.

La escuela de Ingeniería en Sistemas de Computación ha adoptado el estándar Camel Case para la creación de identificadores, por lo cual, ten presente lo siguiente:

- El identificador debe representar lo que va a contener. Ejemplos: total, cantidadProducto, notaExamen,
- Se escriben en minúscula
- Si se utilizan dos o más palabras la segunda y subsiguientes inician con mayúscula, ejemplos: precioCosto, precioVenta, nombreCliente.

Algunos ejemplos para definir variables / constantes de acuerdo con las reglas y recomendaciones



Tipos de Datos

Los tipos de datos son categorías que se han definido para cada dato que se utiliza en un programa. Dos grandes categorías son las siguientes:

Simples

Puede almacenar un valor a la vez.

Una edad, un salario, un correo, etc.

Son los tipos que utilizaremos en los primeros programas que crearemos.

Estructurados

Una sola estructura puede almacenar múltiples datos de manera simultánea.

Una lista de edades, todos los datos de matrícula, etc.

Los estudiaremos más adelante.

Tipos de Datos Simples

En los lenguajes de programación, existen varios de tipos de datos, utilizados para diferentes propósitos, estos se dividen de manera general en:

Numéricos

Pueden almacenar únicamente números, enteros, con decimales, etc.

Son utilizados para realizar operaciones aritméticas.

Cadena de caracteres

Almacenan caracteres, que pueden ser letras, números y caracteres especiales

Aunque pueden contener números, estos no pueden ser utilizados para operaciones aritméticas.

¿Qué tipo de dato podríamos utilizar para almacenar…?

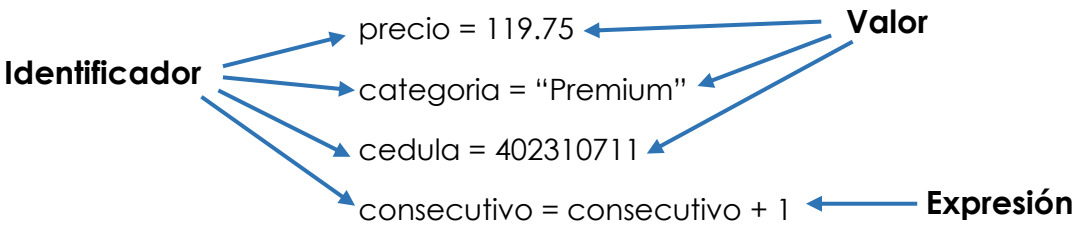
Salarios	○	**Numéricos**
Número de cédula	○	
Correo electrónico	○	**Cadena de caracteres o String**
Número de teléfono	○	

Une el dato de la izquierda con el tipo de datos que consideras conveniente

Operador de asignación

El operador de asignación se utiliza para almacenar un valor dentro de una variable. La asignación es una operación destructiva, esto significa que si la variable tenía un valor almacenado este se perderá.

Ejemplo:



Al ejecutar la instrucción precio = 119.75, se almacena el valor 119.75 en un espacio en la memoria, y podemos saber lo que hay allí utilizando el identificador precio.

Luego podríamos utilizarlo en otra operación como total = precio * 5

Operaciones aritméticas

En los programas se pueden realizar todas las operaciones aritméticas necesarias para resolver un problema de la vida real.

Operador	Nombre	Ejemplo	Resultado
+	Suma	10 + 5	15
-	Resta	10 – 5	5
-	Negación	-10	-10
*	Multiplicación	10 * 5	50
/	División exacta	10 / 5	2
%	Módulo	10 % 5	0
**	Exponente	10 ** 5	100 000
//	División entera	9 // 4	2

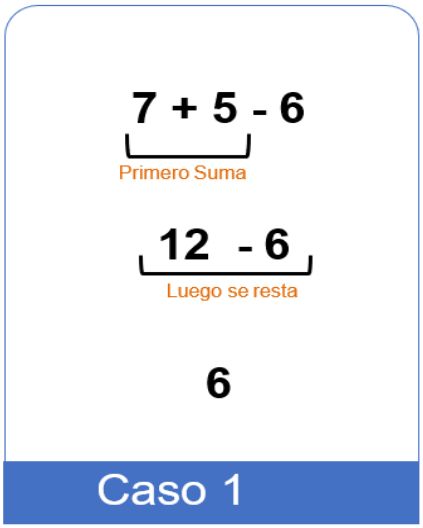
Las operaciones aritméticas pueden contener más de un operador, lo que llamamos una expresión, en estos casos se respeta la precedencia de los operadores. Esta prioridad es modificada solamente cuando se utilizan los paréntesis.

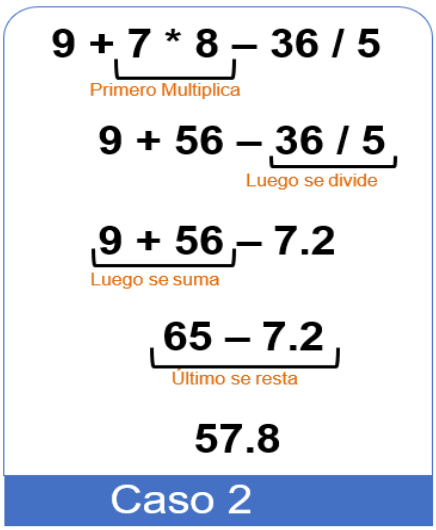
En el caso de que se encuentren operadores con el mismo nivel de prioridad, se resuelve de izquierda a derecha.

Orden de Precedencia

( )				Mayor
**				
*	/	//	%	
+		-		Menor

En los ejemplos presentados hay operadores con el mismo nivel de precedencia, por lo tanto, se resuelven primero las operaciones que estén más a la izquierda.





Probá los resultados en Python y experimentá con otras operaciones o cambiando prioridades

En los ejemplos anteriores solo se utilizaron números para ilustrar el orden de precedencia, pero al programar se utilizan variables, veamos algunos ejemplos.

1. Se requiere calcular el total a pagar en colones por un servicio de transporte, tomando en cuenta que el primer kilometro tiene un precio diferente a los kilómetros restantes.

```
kmsRecorridos = 6

precio1erKm = 800

precioKmAdicional = 450

totalPagar = (kmsRecorridos -1) * precioKmAdicional + precio1erKm
```

Con los valores actuales en las variables el resultado es 3050 colones

¿Por qué se incluyeron paréntesis?

Si no se usan los paréntesis, la operación devolvería un resultado incorrecto de 356 colones.

```
totalPagar = kmsRecorridos -1 * precioKmAdicional + precio1erKm
```

La multiplicación tiene mayor precedencia que la resta por lo cual se realizaría primero, y la operación sería algo así:

```
kmsRecorridos – precioKmAdicional + precio1erKm
```

2. Se requiere calcular la nota final de cada estudiante, dado que

Proyecto	40%
4 quices	15%
2 Tareas	20%
Examen	25%
Total	100%

Para cada rubro se requiere multiplicar la nota obtenida por el porcentaje correspondiente e ir sumando con los demás rubros.

```
notaFinal = notaProyecto * 0.40 + (quiz1+ quiz2 + quiz3 + quiz4) / 4 * 0.15 +
(tarea1 + tarea2) * 0.20 + examen * 0.25
```

Existen otros operadores de asignación compuestos que realizan operaciones aritméticas en el proceso, tales como los siguientes.

Operador	Ejemplo	Equivalencia
+=	X += 2	X = X + 2
-=	X -= 2	X = X – 2
*=	X *= 2	X = X * 2
/=	X /= 2	X = X / 2
%=	X %= 2	X = X % 2
//=	X //=2	X = X // 2
**=	X **= 2	X = X ** 2

Operaciones lógicas

Son operaciones que nos brindan por resultado un valor verdadero o falso (booleano), estas operaciones tienen como uno de sus principales objetivos la toma de decisiones en nuestras soluciones.

También tenemos operadores que se utilizan en este tipo de operaciones.

Operador	Operación	Ejemplo	Resultado
==	Igual que	“hola” == “lola”	FALSO
!=	Diferente a	“a”! = “b”	VERDADERO
<	Menor que	7 < 15	VERDADERO
<=	Menor o igual que	22 <= 15	FALSO
>	Mayor que	5 > -21	VERDADERO
>=	Mayor o igual que	8 >= 9	FALSO

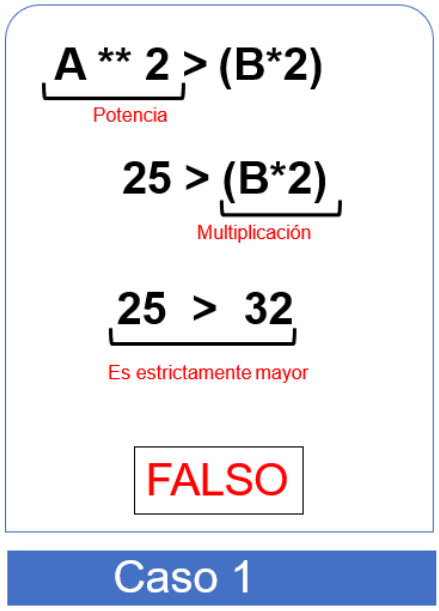
Precedencia de Operadores

( )						Mayor
**						
*	/	//	%			
+		-				
==	!=	>	>=	<	<=	Menor

Los operadores lógicos podemos resolverlos dentro de operaciones aritméticas, siempre considerando que la presencia de un operador lógico nos dará como resultado un verdadero o un falso.

Asumiendo los valores A y para B

A=5 **B=16**



Caso 1

Ejercicio

-Resolver la siguiente operación, X tiene un valor de 4 y B tiene un valor de 2.

(X * 5 + B ** 3 / 4) >= (X ** 3 – 1)

Resultado:

En algunas ocasiones nos encontraremos con situaciones en donde debemos obtener un valor lógico a partir de múltiples comparaciones, en estos casos utilizaremos las **Tablas de Verdad**, una herramienta de agrupación de operaciones lógicas que nos ayudan a obtener un único resultado lógico a partir de múltiples comparaciones.

Operador	Operación	Operador	Resultado
Verdadero	AND	Verdadero	Verdadero
Verdadero		Falso	Falso
Falso		Falso	Falso
Verdadero	OR	Verdadero	Verdadero
Verdadero		Falso	Verdadero
Falso		Falso	Falso

Operador	Operación	Resultado
Verdadero	NOT	Falso
Falso		Verdadero

Prioridad de Operaciones

( )						Mayor
**						
*	/	//	%			
+		-				
==	!=	>	>=	<	<=	
NOT						
AND						
OR						Menor

Operadores lógicos (ejemplos)

NOT (5 > 18 AND 18 > 5)

FALSO VERDADERO

FALSO

VERDADERO

NOT (5 > 18 OR 18 > 5)

FALSO VERDADERO

VERDADERO

FALSO

Ejercicio

Resolver la siguiente operación

NOT (5 > 18 OR 18 > 5 AND (3+2) >= 5)

Documentación / Comentarios

Desde los inicios de la programación, siempre ha sido necesario escribir comentarios dentro de los programas con el fin de explicar al mismo programador (o a otros) cuál es el objetivo de alguna funcionalidad.

En Python podemos definir comentarios con el carácter numeral de la siguiente manera.

```
#Esto es un comentario
print("Hola Mundo")
#Esto es una prueba
```

Resuelva los siguientes ejercicios

1. Construya un programa tal que, dados los datos para las variables A, B, C y D que representan números enteros, los imprima en orden inverso.
2. Elabore un programa que solicite al usuario la edad y calcule cuántos años tendrá la persona en 5 años, al finalizar se imprime el siguiente mensaje "Dentro de 5 años, tendrá:" y se muestra el valor de la edad proyectada.

   Nota: en clase su profesor le explicará como solicitarle al usuario un valor
3. Construya un programa que le solicite al usuario los datos enteros A y B y muestre el resultado de la siguiente expresión.

$$\frac{(A+B)^2}{3}$$

4. Desarrolle un programa que le solicite un número al usuario y calcule el cuadrado y el cubo de este.
5. Realice un programa que, dada la base y la altura de un rectángulo, calcule el área y el perímetro de este.
6. Desarrolle un programa que solicite la distancia de su casa a la Universidad, el costo por kilómetro, la cantidad de días a la semana que viaja a la Universidad y que calcule el costo total de trasladarse por

cuatrimestre. Asuma que cada visite implica ida y vuelta y que el cuatrimestre tiene 15 semanas.

7. Desarrolle un programa que solicite al usuario la edad de 5 personas y le muestre cuál es la edad promedio.
8. Desarrolle un programa que solicite al usuario la cantidad de horas semanales trabajadas, el precio que se le paga por hora y que calcule el salario mensual. Considere que se debe aplicar una deducción del 10.5% por cargas sociales y 5% por asociación solidarista. Asuma que cada mes cuenta con 4.2 semanas.
9. Desarrolle un programa que le solicite al usuario sus ingresos y sus gastos mensuales por alimentación. Con esta información el programa debe mostrar el porcentaje que gasto que corresponde al rubro de alimentación y el porcentaje que queda disponible para otros rubros.

-Esta semana hemos experimentado con nuestros primeros programas en Python, el docente ha construido con usted las soluciones.

-Experimente nuevas funcionalidades y operaciones que podría implementar en los temas diarios.

Lecturas adicionales:

eLibro.net

https://elibro.net/es/ereader/ufidelitas/106404?page=23

https://elibro.net/es/ereader/ufidelitas/106404?page=30