# Trabajo práctico nro. 1

| | | |
|---|---|---|
| UTN FACULTAD REGIONAL MENDOZA | **Asignatura: Programación I** | |
| | | |
| | **Cursado:**Primer Trimestre | **Horas semanales**: |
| | | **Horas semestrales:** *Cantidad estimada de horas semestrales/anuales.* |
| | **Carrera**: *Tecnicatura Universitaria en Programación* | **Nivel (Año):** |
| | **Ciclo Lectivo: 2023** | ☐ 1° ☐ 2° ☐ 3° |

**Integrantes de la Cátedra:**

- **DOCENTES:**

| Nombre del Profesor | Periodo | Cantidad horas materia |
|---|---|---|
| **Rigoni Cinthia** | | 6 horas |

1. Indica si los siguientes identificadores son válidos en Python. En el caso de que el identificador no sea válido, explica el motivo.

a) alumno1
b) 1alumno
c) primerNombre
d) /apellido
e) tamaño_máximo
f) for
g) _$nombre
h) global
i) primer_nombre
j) num_mayor
k) menor-num
l) dni@alumno
m) 5var
n) with
o) Auto-seleccionado
p) %aumento
q) _123
r) ValorTotal
s) DESCUENTO
t) año
u) mes_actual
v) apellido&nombre
w) 89GW5
x) valido?

Debemos tener en cuenta que no pueden usarse palabras reservadas ni caracteres especiales en la declaración de variables.

a) Es válido pero es una mala práctica utilizar números en el identificador de una variable
b) No es válido porque inicia con números
c) Es válido pero no es recomendable utilizar camelCase en Python si no snake_case
d) No es válido porque utiliza /
e) No es válido porque utiliza "ñ"
f) No es válido porque es una palabra reservada
g) No es válido porque utiliza carácter especial ($) y no es buena práctica utilizar (_) al comienzo de un identificador, esto se utiliza para indicar variables auxiliares.
h) No es válido porque es una palabra reservada
i) Válido
j) Válido
k) No es válido porque utiliza un signo reservado, el de resta (-)
l) No es válido porque utiliza carácter especial (@)
m) No es válido porque comienza con número
n) No es válido porque utiliza una palabra reservada
o) No es válido porque involucra el símbolo de resta y es una mala práctica porque comienza con mayúscula
p) No es válido porque utiliza el símbolo de módulo.
q) Es válido pero no es una buena práctica utilizar números en el identificador y tampoco un guión bajo (_) en variables no auxiliares.
r) Es válido pero no es una buena práctica utilizar CamelCase en Python
s) Es válido pero no es una buena práctica utilizar mayúsculas en variables que no son constantes.
t) No es válido porque utiliza "ñ"
u) Es válido
v) No es válido porque usa símbolo reservado (&)
w) No es válido porque inicia con números
x) No es válido porque usa caracteres especiales (?)

2. Indica qué dato se guarda en la variable **x** en cada caso, suponiendo una ejecución secuencial del programa.

a) x=46
x=15
x=30

b) x=46
x=15
x=30

c) x=25
x+10

d) x=10-2
10+2

e) y=3*(4+2)
x=y+2
z=5
x=y-z

f) x=3
y=x+6
x=y-1

a) x = 30 porque es la última asignación de x
b) x = 30 porque es la última asignación de x
c) x= 24 porque no se vuelve a asignar
d) x = 8 porque no la operación siguiente no se asigna a x
e) x = 13 porque es la última asignación de x
f) x = 8 porque es la última asignación de x

3. Indica qué tipo de dato se guarda en cada variable.

a) var1 = 100/5
b) var2 = 7/2
c) var3 = 7//2
d) var4 = 7%2
e) var5 = 'a'
f) var6 = "casa"+"s"
g) var7 = "automóvil"[1+1]
h) var8 = len("carpeta")
i) var9 = int("748")
j) var10 = float("832")
k) var11 = float(321)
l) var12 = str(65)
m) var13 = 1+5!=3
n) var14 = 177%2==0
o) var15 = len("ola")<=12

a) float
b) float
c) int
d) int
e) str
f) str
g) str
h) int
i) int
j) float
k) float
l) str
m) bool
n) bool
o) bool

4. Indica cuáles de las siguientes operaciones no son válidas.

a) 11-(4%2+10)
b) “30”+”2”
c) “30”+2
d) “hola”[len(“hola”)]
e) len(456)
f) “hola”[len(“fin”)]
g) int(“4”)
h) int(4)
i) int(“z”)
j) int(“4.”)
k) 4<”f”
l) “palabra”=”rama”

c) No es válida porque no puedo concatenar un número con una string

d) No es válida porque el índice de la string queda fuera de rango

e) No es válida porque no puedo utilizar el método len con un integer

i) No es válida porque el método int espera un parámetro de base 10

j) No es válida porque el método int espera un parámetro de base 10

k)No es válido porque no se soportan comparaciones entre integer y string

l) No es válido porque no puedo nombrar una variable con string

5. Declara una variable de cada tipo de dato y asígnale un valor.

- int
- float
- complex
- string
- bool
- list
- tuple
- dict
- **null**

Sabías que en Python al momento de declarar una variable es necesario darle un valor. Por lo que para que esta sea ‘vacia’, podemos declarar de la siguiente forma:

```
var_nula = None
```

int

```
var_entera= 5
```

list

```
var_lista=[9,8,6,5,4,3,2]
```

| | | | |
|---|---|---|---|
| float | `var_decimal = 5.5` | tuple | `var_tupla=("colores","animales","deportes")` |
| complex | `var_compleja = 3j` | dict | `var_diccionario={"clave": "valor"}` |
| string | `var_string= "hola"` | null | `var_nula=None` |
| bool | `var_booleana=5<8` | | |

6. Teniendo la variable de tipo **string:** frase = “Caminante, no hay camino, se hace camino al andar.”, indica qué obtendremos si aplicáramos:

   a) frase[5]
   “a”
   b) frase[-1]
   “.”
   c) frase[0:8]
   “Caminant”
   d) frase[::3]

   “Cin,oaci,ea molnr”

7. Usando la variable del ejercicio anterior:

   a) ¿Cómo obtenemos la cadena al revés? “.radna la onimac ecah es ,onimac yah on ,etnanimaC”
   Podemos dar vuelta la cadena utilizando:
   var = ".radna la onimac ecah es ,onimac yah on ,etnanimaC"
   print(var[::-1])

   b) ¿Cómo obtenemos la subcadena ‘hace’?

   Primero tenemos que asignar el valor del ejercicio anterior a una variable auxiliar o reasignar la variable actual y luego podemos utilizar un método para sacar una subcadena:

print(var[29:33])

8. Métodos upper(), lower() y title().

El método **title()** cambia la primera letra de cada palabra a mayúscula.

   a) Pon en mayúsculas la primera letra de cada palabra del siguiente nombre: ‘lucas mauricio barros’.
   palabra="lucas mauricio barros"
   print(palabra.title())
   b) Deja esta frase totalmente en letras minúsculas: ‘El qUe No arRiesGa, nO gANa.’
   palabra= "El qUe No arRiesGa, nO gANa."
   palabra = palabra.lower()

print(palabra)

c) Deja esta frase totalmente en letras mayúsculas: 'El qUe No arRiesGa, nO gANa.'

```
palabra= "El qUe No arRiesGa, nO gANa."
palabra = palabra.upper()
print(palabra)
```

9. Convierte en expresiones algorítmicas las siguientes expresiones algebraicas. Coloca paréntesis solamente donde sean necesarios.

a) $\frac{b}{2} - 4ac$

b) $3xy - 5x + 12x - 17$

c) $\frac{b+d}{c+4}$

d) $\frac{xy}{y} + 2$

e) $\frac{1}{y} + \frac{3x}{z} + 1$

f) $\frac{1}{y+3} + \frac{x}{y} + 1$

g) $a^2 + b^2$

h) $(a+b)^2$

i) $\sqrt[3]{b} + 34$

j) $\frac{x}{y}(z+w)\pi$

k) $\frac{x+y}{u+\frac{w}{b}}$

a) b/2 - 4 * a * c
b) 3*x y - 5*x + 12*x - 17
c) (b+d)/(c+4)
d) (x*y)/y + 2
e) 1/y + (3*x)/z + 1
f) 1/(y+3) + (3*x)/z + 1
g) a**2 + b**2
h) (a+b)**2
i) b**(⅓) + 34
j) (x/y) * (z+w) * π
k) (x+y)/ (u + w/b)

10. Convierte en expresiones algebraicas las siguientes expresiones algorítmicas. Coloca paréntesis solamente donde sean necesarios.

a) x=(-b+(b**2-4*a*c)**(1/2))/(2*a)

b) (x**2+y**2)/(z**2)

c) 4*x**2-2*x+7

d) (b**2)**(1/2)-4*a*c

e) (a-b)**2+(c-d)**3

f) (x+y)/y-(3*x)/5

g) (a**2+b**2)**(1/3)=c

h) 3*x**2/(3*x**3/(4*y+6))**(1/2)

a) $x= \sqrt{(-b + b^2 - 4ac)} / 2a$
b) $(x^2 + y^2 ) / z^2$
c) $4x^2 - 2x+7$
d) $\sqrt{(b^2)} - 4ac$
e) $(a-b)^2 + (c-d)^3$
f) $(x+y)/y - (3x)/5$
g) $\sqrt[3]{(a^2 + b^2)} = c$
h) $3x^2 / \sqrt{(3x^3/ (4y+6))}$

Nota: tuve que usar paréntesis donde normalmente no usaría, por ejemplo en las raíces, para que se entienda mejor donde termina la raíz.

11. Dada la siguiente expresión aritmética:

$$a + b * \left(5 - \frac{c}{2}\right) + (7 - x)/(y + 4)$$

Determinar qué resultado obtendremos si a=5, b=2, c=6, x=(-6) y y=4.

a=5

b=2

c=6

x=(-6)

y=4

resultado = a + b*(5-(c/2)) + (7-x)/(y+4)

print(resultado)

resultado arrojado por consola: 10.625

12. Escribe las expresiones algorítmicas equivalentes a los siguientes enunciados:

a) Suma los números 5 y 3.

b) Calcula el promedio de los números 4, 7 y 9.

c) Calcula el área de un rectángulo con base 8 y altura 5.

d) Verifica si un número es par.

e) El doble de 16.

f) Seis veces la diferencia de 8 y 3.

g) La diferencia entre el producto de 2 por 6 y la suma de 4 y 3.

h) Comprobar si un número entero N es múltiplo de 2 y de 3.

i) Comprobar si el contenido de la variable precio es igual o mayor que 15 y menor que 90.

j) Modificar el valor de la variable entera N incrementándolo en 12.

k) Modificar el valor de la variable entera N disminuyéndolo en 5.

l) Modificar el valor de la variable entera N triplicando su valor.

m) Modificar el valor de la variable entera N por su mitad.

a) sumando = 5+3

b) promedio = (4 + 7 + 9) / 3

c) base = 8

altura = 5

area_rectangulo = base * altura

d) num_par = num%2 == 0

e) num = 16

doble_numero = num * 2

f) diferencia_por_seis = (8-3) * 6

g) num = (2*6) - (4+3)

h)
```
if (N % 2 == 0) and (N % 3 == 0) {
        print(N , “es múltiplo de 2 y 3”)
}
```

i)
```
if (precio >= 15) and (precio < 90){
   return True
}
```

j)
```
for n in range(0, 100, 12):
   print(n)
```

k)
```
for n in range(100, 0, -5):
   print(n)
```

l) n = 3

```
for i in range(3):
  n = n *3
  print(n)
```

resultados arrojados por consola: 9, 27, 81

m)
```
n = 80
for i in range(4):
  n = n / 2
  print(int(n))
```

Resultados arrojados por consola: 40, 20, 10, 5

13. ¿Qué resultado (True/False) dan las siguientes operaciones?

a) not true

b) not(1+2 != 3)

c) x = (len('jugar') > 5) and (len('jugar') < 10)

d) 'alto'[2] == 't' and x

e) 842913%10 != 3 and len('café') == 3

f) 0 != 0 or 'a' < 'y'

g) True or int('50') >= 50

h)
```
edad = 20
not(x) or edad%2 == 0
```

i)
```
es_cliente = False
not(es_cliente and not(edad < 18))
```

a) False
b) True
c) False
d) False
e) False
f) True
g) True
h) True
i) True

14. Siendo x una variable de tipo entera, con valor 5, determine qué se mostrará por pantalla en cada caso.

a) print(x += 1)

b) print(x -= 2)

c) print(x *= 5)

d) print(x /= 5)

Sabías que en Python no existen los operadores de incremento y decremento, por lo que si queremos aumentar en 1 una variable usamos el operador +=1. En caso de querer disminuir en 1 una variable usamos el operador -=1. Este operador es válido para sumar o restar cualquier valor.

Podemos aplicar esta misma lógica con la multiplicación y la división usando los operadores *= y /= respectivamente.

En teoría funciona pero me saltaba error. Para poder imprimir el resultado en cada una de las operaciones tuve que declarar x, asignarle la operación y luego imprimir la x.

a) 6
b) 3
c) 25
d) 1

15. Tipos *list*, *tuple* y *dict*.

Una **lista** es una variable con múltiples valores. Pueden contener cualquier tipo de dato soportado por Python en cualquier orden.

Por ejemplo: **lista = ['texto', 10, 15.6, 'texto']**

a) De la siguiente lista, ¿qué color está en la posición 3?, ¿cómo accedemos a esta posición?

```
colores = ["rojo", "azul", "verde", "amarillo", "marrón", "lila", "negro", "rosa"]
```

Amarillo

b) ¿En qué posición se encuentra el color 'rojo'? ¿Y el 'rosa'?
colores[0] ="rojo"
colores[7] = "rosa"

c) Crea una lista que contenga los siguientes valores en las posiciones indicadas.

- 'uno' en la posición 4.
- 'dos' en la posición 1.
- 'tres' en la posición 0.
- 'cuatro' en la posición 3.
- 'cinco' en la posición 2.

numeros= ["tres", "dos", "cinco", "cuatro", "uno"]

Las **tuplas** son como las listas pero con dos diferencias clave. La primer diferencia es que las tuplas se escriben con paréntesis () y las listas con corchetes []. La segunda diferencia es que las tuplas son inmutables y las listas no.

Las listas son capaces de variar, podemos introducir datos, ordenarlos, eliminarlos, etc. En cambio, las tuplas no pueden, son como listas constantes que no se pueden modificar.

Aquí tienes un ejemplo de como se ve una tupla: **tupla = ('texto', 10, 15.6, 'texto')**

d) Imprime la segunda posición de esta tupla.

```
colores = ('rojo', 'azul', 'verde', 'amarillo', 'marrón', 'lila', 'negro', 'rosa', 'blanco', 'naranja')
```

colores[2] = "verde"

e) Utiliza los símbolos de suma y resta para obtener el resultado 25 a partir de los elementos de la siguiente tupla en una variable llamada operacion.

```
numeros = (10, 1, 5, 11)
```

veinticinco=numero[0]+ numero[3]+ numero[2] - numero[1]

print(veinticinco)

resultado arrojado por consola: 25

Un **diccionario** en Python es una estructura de datos que permite almacenar cualquier tipo de información. Los valores de un diccionario se guardan utilizando un par de valores que siempre van enlazados. Una es la denominada como Key o *Clave*, que es la que nos permite encontrar un dato dentro del diccionario. Cada clave está acompañada por el dato o *valor* al que representa.

Veamos un ejemplo para comprender mejor: **diccionario = {'nombre':'Antonio', 'apellido':'López', 'edad':35, 'peso':72.6}**

f) Cuenta la cantidad de elementos del siguiente diccionario.

```
diccionario = {"a": 1, "b": 2, "c": 3, "d": 4}
```

diccionario tiene 4 elementos

g) Accede al valor de la clave 'c' en el diccionario.

print(diccionario["c"])

16. Vamos a practicar el uso de las funciones **input()** y **print()**.

Ejemplo: Solicita el nombre de una persona e imprime un mensaje de bienvenida.

```
nombre = input("Ingresa tu nombre: ")
print("¡Hola", nombre + "! Bienvenido(a).")
```

a) Solicita dos números al usuario, súmalos e imprime el resultado.

```
num1= int(input("Ingresa un numero: "))
num2 = int(input("Ingresa otro numero: "))
suma = num1 + num2
print(f"La suma de los numeros ingresados es de {suma}")
```

b) Solicita la edad de una persona, calcula cuántos años faltan para que cumpla 100 años e imprime el resultado.

```
edad_persona = int(input("Ingrese su edad: "))

falta_para_cien = 100 - edad_persona

print(f"Faltan {falta_para_cien} para cumplir 100 años")
```

17. Operadores ternarios.

Los operadores ternarios son más conocidos en Python como expresiones condicionales. Estos operadores evalúan si una expresión es verdadera o no.

Estructura: `condition_if_true if condition else condition_if_false`

Un ejemplo:

```
es_bonito = True
estado = "Es bonito" if es_bonito else "No es bonito"
```

*¡Practiquemos!* Crear las variables necesarias para realizar la ejercitación.

a) Comprobar si un número es par o impar.

```
numero = int(input("Ingrese un número: "))
numero_par = True if (numero%2==0) else False
print(numero_par)
```

b) Obtener el valor absoluto de un número.

```
numero = int(input("Ingrese un número: "))
valor_absoluto = numero if (numero > 0) else numero*-1
print(f"El valor absoluto del número ingresado es: {valor_absoluto}")
```

c) Comparar dos números y obtener el mayor.

```
numero_uno = int(input("Ingrese un número: "))

numero_dos = int(input("Ingrese un segundo número: "))

mayor = "El primer número es mayor" if (numero_uno > numero_dos) else "El segundo número es mayor"

print(mayor)
```