Empezando a programar

Ejercicios para aprender practicando

Herramientas

Podés resolver los ejercicios descargando Python 3 y generando, por cada ejercicio, un archivo con extensión *.py* y ejecutándolo mediante el intérprete de Python. Sin embargo, también es posible resolver los ejercicios en cualquier intérprete de Python online, sin necesidad de descargar nada, por ejemplo: https://repl.it/languages/python3. Cada ejercicio de este documento constituye un programa.

Importante

Cuando resuelvas los ejercicios estarás actuando como el "programador" (la persona que crea el programa). Pero luego tendrás que ejecutar tu código y probar si funciona correctamente, y en este caso actuarás como el "usuario" (una persona que utiliza el programa). No siempre el programador y el usuario son la misma persona y, generalmente, los usuarios no conocen los detalles técnicos ni pueden ver el código. Sólo verán la salida en pantalla que les muestre el programa.
Considerá que, en cada ejecución del programa, el usuario podría ingresar datos diferentes. El programa debería funcionar de forma consistente y sin errores en todos los casos.

Sobre los ejercicios

Los ejercicios están separados en tres secciones, correspondientes al video en https://youtu.be/IV_OxGH8ZYE, el cual contiene explicaciones sobre los conceptos necesarios para resolver los ejercicios.
Junto a los ejercicios se encuentran algunos ejemplos con explicaciones, a modo de guía para su resolución. Éstos están ubicados en recuadros de color gris.
Además, por cada ejercicio se muestra un ejemplo de salida en pantalla indicando cómo debería quedar el programa luego de su ejecución. El texto en negrita representa la salida del programa y el texto en cursiva representa el texto ingresado por el usuario.

Soluciones

Al final del documento se encuentra una sección con las soluciones a los ejercicios. Es importante tener en cuenta que, en algunos casos, es posible obtener el mismo resultado utilizando diferentes estrategias o algoritmos. En este caso se plantea una solución posible para cada ejercicio.

Sección 1

Entrada/salida de datos - Variables - Tipos de datos

```
Variable=5
variable=3
VaRiAbLe=8
```

Son todas variables distintas porque *Python* (como muchos otros lenguajes) distingue mayúsculas y minúsculas.

```
a=5
a=18
```

a tomará el último valor asignado (lo que tuviera guardado anteriormente la variable, se pierde).

1. Escribí un programa que solicite al usuario que ingrese su nombre. El nombre se debe almacenar en una variable llamada *nombre*.
A continuación se debe mostrar en pantalla el texto "Ahora estás en la matrix, *[usuario]*", donde "*[usuario]*" se reemplazará por el nombre que el usuario haya ingresado.

Ejemplo de ejecución:

Tu nombre: *Patricia*
Ahora estás en la matrix, Patricia

```
1variable=23.95
```

Arroja error porque los nombres de variables sólo pueden comenzar con letras o guiones bajos (_).

```
n1=int(input())
n2=float(input())
```

Sabemos que **input**() lee lo que el usuario escribe en el programa, pero el tipo de eso que lee será siempre *string*. Si necesitamos que sea un número debemos convertir lo que **input**() devuelve. Para convertir a número entero usamos **int**(**input**()) y para convertir a número con decimales usamos **float**(**input**()).

2. Escribí un programa que solicite al usuario ingresar un número con decimales y almacenalo en una variable. A continuación, el programa debe solicitar al usuario que ingrese un número entero y guardarlo en otra variable. En una tercera variable se deberá guardar el resultado de la suma de los dos números ingresados por el usuario. Por último, se debe mostrar en pantalla el texto "El resultado de la suma es *[suma]*", donde "*[suma]*" se reemplazará por el resultado de la operación.

Ejemplo de ejecución:

Primer número: *14.2*
Segundo número: *19*
El resultado de la suma es 33.2

3. Escribí un programa que solicite al usuario dos números y los almacene en dos variables. En otra variable, almacená el resultado de la suma de esos dos números y luego mostrá ese resultado en pantalla.
A continuación, el programa debe solicitar al usuario que ingrese un tercer número, el cual se debe almacenar en una nueva variable. Por último, mostrá en pantalla el resultado de la multiplicación de este nuevo número por el resultado de la suma anterior.

Ejemplo de ejecución:

Ingresá un número: *1*
Ingresá otro número: *2*
Suman: *3*
Ingresá un nuevo número: *3*
Multiplicación de la suma por el último número: *9*

```
a=a+1
a+=1
```

Las dos instrucciones de arriba son equivalentes (cualquiera de ellas hace lo mismo: sumar 1 al valor almacenado en la variable **a** y reemplazar el valor anterior de **a** con el nuevo resultado).

4. Escribí un programa que solicite al usuario ingresar la cantidad de kilómetros recorridos por una motocicleta y la cantidad de litros de combustible que consumió durante ese recorrido. Mostrar el consumo de combustible por kilómetro.

Ejemplo de ejecución:

Kilómetros recorridos: *260*
Litros de combustible gastados: *12.5*
El consumo por kilómetro es de 20.8

5. Escribí un programa que solicite al usuario el ingreso de una temperatura en escala Fahrenheit (debe permitir decimales) y le muestre el equivalente en grados Celsius. La fórmula de conversión que se usa para este cálculo es: *Celsius = (5/9) * (Fahrenheit-32)*

Ejemplo de ejecución:

Ingresá una temperatura expresada en Farenheit: *75*
23.88888888888889

6. Escribí un programa que solicite al usuario ingresar tres números para luego mostrarle el promedio de los tres.

Ejemplo de ejecución:

Primer número: *8.5*
Segundo número: *10*
Tercer número: *5.5*
El promedio de los tres es 8.0

7. Escribí un programa que solicite al usuario un número y le reste el 15%, almacenando todo en una única variable. A continuación, mostrar el resultado final en pantalla.

Ejemplo de ejecución:

Ingresá un número: *260*
Descontando el 15% queda: 221.0

```
cadena=""
cadena=cadena+"buen"
cadena=cadena+" día"
print(cadena)
```

Cuando se utiliza el operador + en una operación entre strings, se está realizando una concatenación (unión de strings). La instrucción print mostrada arriba imprimirá “buen día”.

8. Escribí un programa que solicite al usuario el ingreso de dos palabras, las cuales se guardarán en dos variables distintas. A continuación, almacená en una variable la concatenación de la primera palabra, más un espacio, más la segunda palabra. Mostrá este resultado en pantalla.

Ejemplo de ejecución:

Primera palabra: *derribando*
Segunda palabra: *muros*
derribando muros

```
frase="Estoy programando"
print(frase[0])
i=6
print(frase[i])
```

El operador [ ] (corchetes) permite obtener un carácter a partir de un *string*. La posición del carácter se indica entre los corchetes, ya sea ingresando directamente el número, con una variable que contenga un número o con una operación que de como resultado un número. Siempre, el primer carácter de un *string* estará ubicado en la posición 0.

```
frase="Estoy programando"
print(len(frase))
ultimo_caracter=frase[len(frase)-1]
print(ultimo_caracter)
```

Mediante **len**() podemos obtener la cantidad de caracteres que contiene un *string*. Este valor siempre será un número entero (tipo *int*) y puede guardarse en una variable, imprimirse, usarse en una operación aritmética, etc.

9. Escribí un programa que solicite al usuario el ingreso de un texto y almacene ese texto en una variable. A continuación, mostrar en pantalla la primera letra del texto ingresado.
Luego, solicitar al usuario que ingrese un número positivo menor a la cantidad de caracteres que tiene el texto que ingresó (por ejemplo, si escribió la palabra “HOLA”, tendrá que ser un número entre 0 y 4) y almacenar este número en una variable llamada *indice*.
Mostrar en pantalla el carácter del texto ubicado en la posición dada por *indice*.

Ejemplo de ejecución:

Ingresá un texto: *En un lugar de la Mancha, de cuyo nombre no quiero acordarme...*
El carácter en primer lugar es: E
Ingresá un número positivo menor a 63
7
El carácter en esa posición es: u

```
a=10
b=4
print(a != b)
```

La instrucción print imprimirá True, ya que el valor contenido en **a** es diferente del valor contenido en **b**.

10. Escribí un programa que solicite al usuario que ingrese cuántos shows musicales ha visto en el último año y almacene ese número en una variable. A continuación mostrar en pantalla un valor de verdad (True o False) que indique si el usuario ha visto más de 3 shows.

Ejemplo de ejecución:

Shows vistos en el último año: *3*
False

```
print(58273%10)
print(58273//10)
```

La primera instrucción imprimirá el número 3, ya que es el resto de la división de 58273 por 10. La segunda instrucción imprimirá 5827, ya que es la parte entera del resultado de dividir 58273 por 10. Estas operaciones matemáticas son estrategias que se pueden utilizar para obtener partes de un número.

11. Escribí un programa que le solicite al usuario ingresar una fecha formada por 8 números, donde los primeros dos representan el día, los siguientes dos el mes y los últimos cuatro el año (*DDMMAAAA*). Este dato debe guardarse en una variable con tipo *int* (número entero). Finalmente, mostrar al usuario la fecha con el formato *DD / MM / AAAA*.

Ejemplo de ejecución:

Fecha en formato DDMMAAAA: *16112017*
16 / 11 / 2017

12. Escribí un programa para solicitar al usuario el ingreso de un número entero y que luego imprima un valor de verdad dependiendo de si el número es par o no. Recordar que un número es par si el resto, al dividirlo por 2, es 0.

Ejemplo de ejecución:

Número entero: *7254*
True

```
a=int(input())
print(a>100  and  a!=1000)
```

Primero se calcularán los valores lógicos (True o False) de las dos comparaciones: **a** > **100** y **a** != **1000** (lo cual dependerá del número guardado en la variable **a**). A continuación, se utilizará la tabla de verdad de la operación **AND** para calcular el resultado.

13. Escribí un programa que le solicite al usuario su edad y la guarde en una variable. Que luego solicite la cantidad de artículos comprados en una tienda y la guarde en otra variable. Finalmente, mostrar en pantalla un valor de verdad (True o False) que indique si el usuario es mayor de 18 años de edad y además compró más de 1 artículo.

Ejemplo de ejecución:

Tu edad: *32*
Artículos comprados: *1*
False

14. Escribí un programa que, dada una cadena de texto por el usuario, imprima True si la cantidad de caracteres en la cadena es un número impar, o False si no lo es.

Ejemplo de ejecución:

Ingresá una frase: *Era el mejor de los tiempos, era el peor de los tiempos.*
True

```
"animal" > "piedra"
"bailar" > "bebida"
```

Ambas comparaciones arrojan True porque el string “animal” es menor que “piedra” y el string “bailar” es menor que “bebida”. El orden está dado por cómo aparecen las letras en el alfabeto. En el caso de “animal” y “piedra”, la “a” es menor que la “p”. En el caso de “bailar” y “bebida”, como la primera letra es la misma se evalúa la segunda, y en este caso “a” es menor que “e”.

15. Escribí un programa que le pida al usuario ingresar dos palabras y las guarde en dos variables, y que luego imprima True si la primera palabra es menor que la segunda o False si no lo es.

Ejemplo de ejecución:

Una palabra: *complejidad*
Otra palabra: *algoritmo*
False

16. Escribí un programa para pedir al usuario su nombre y luego el nombre de otra persona, almacenando cada nombre en una variable. Luego mostrar en pantalla un valor de verdad que indique si: los nombres de ambas personas comienzan con la misma letra ó si terminan con la misma letra. Por ejemplo, si los nombres ingresados son *María* y *Marcos*, se mostrará True, ya que ambos comienzan con la misma letra. Si los nombres son *Ricardo* y *Gonzalo* se mostrará True, ya que ambos terminan con la misma letra. Si los nombres son *Florencia* y *Lautaro* se mostrará False, ya que no coinciden ni la primera ni la última letra.

Ejemplo de ejecución:

```
Tu nombre: Alfredo
Otro nombre: Eduardo
True
```

Sección 2

Bloques - Selección - Repeticiones

```
x=10
if x!=0:
    print("Hola")
```

El programa anterior hará que siempre se muestre “Hola” porque la condición que tiene la instrucción if nunca puede ser **False**, ya que la variable **x** vale 10 y 10 siempre será distinto de 0. Si, en lugar de asignar un valor concreto, le solicitamos al usuario que ingrese uno y lo guardamos en **x**, dependerá de si ese valor es distinto de 0 o no para saber si se mostrará la palabra “Hola” o no.

17. Escribí un programa que, dado un número entero, muestre su valor absoluto. Recordá que, para los números positivos su valor absoluto es igual al número (el valor absoluto de 52 es 52), mientras que, para los negativos, su valor absoluto es el número multiplicado por -1 (el valor absoluto de -52 es 52).

Ejemplo de ejecución:

Número: *-12*
Valor absoluto: 12

```
x=int(input("Número:"))
if x%2==0:
    print("El número es par")
else:
    print("El número es impar")
```

La instrucción **if-else** permite que se ejecute un bloque de código u otro, pero nunca ambos. En este ejemplo, si es verdad que el número ingresado por el usuario (y almacenado en la variable **x**) es un número par (es decir, si es verdad que el resto de la división del número por 2 es 0), se imprimirá “El número es par”. Si esa condición es falsa (el resto de la división del número por 2 no es 0), se imprimirá “El número es impar”.

18. Escribí un programa que solicite al usuario el ingreso de dos números diferentes y muestre en pantalla al mayor de los dos.

Ejemplo de ejecución:

Un número: *592*
Otro número distinto: *1726*
1726 es mayor

19. Escribí un programa que solicite al usuario una letra y, si es una vocal, muestre el mensaje “Es vocal”. Verificar si el usuario ingresó un string de más de un carácter y, en ese caso, informarle que no se puede procesar el dato.

Ejemplo de ejecución:

Letra: *o*
Es vocal

20. Escribí un programa para solicitar al usuario tres números y mostrar en pantalla al menor de los tres.

Ejemplo de ejecución:

Primer número: *20*
Segundo número: *30*
Tercer número: *10*
Menor: 10

```
n=int(input("Número:"))

if n>10 and n<20:
    print("Número correcto")
else:
    print("Número incorrecto")


if n<10 or n>20:
    print("Número incorrecto")
else:
    print("Número correcto")
```

En las dos instrucciones **if-else** anteriores se toma como correcto a cualquier número entre 10 y 20, pero es necesario observar cómo los operadores < y > cambian y cómo el operador **and** cambia por **or** para lograr el mismo objetivo.

21. Escribí un programa que solicite ingresar un nombre de usuario y una contraseña. Si el nombre es "*Gwenevere*" y la contraseña es "*excalibur*", mostrar en pantalla "Usuario y contraseña correctos. Puede ingresar a Camelot". Si el nombre o la contraseña no coinciden, mostrar "Acceso denegado".

Ejemplo de ejecución:

Nombre de usuario: *gwen*
Contraseña: *excalibur*
Acceso denegado

22. Escribí un programa que permita saber si un año es bisiesto. Para que un año sea bisiesto debe ser divisible por 4 y no debe ser divisible por 100, excepto que también sea divisible por 400.

Ejemplo de ejecución:

Año: *2020*
Bisiesto

```
for x in range(0,10):
    print(x)

for x in range(10):
    print(x)
```

Las dos repeticiones anteriores imprimirán lo mismo: los números del 0 al 9. En la primera se indica que se debe comenzar en el 0 y terminar en el 9 (el rango siempre terminará en el número dado como final, menos 1). En la segunda, al obviar el número en que se debe comenzar, automáticamente se toma al 0 como inicio del rango. Si quisiéramos, podríamos indicar que el rango comience en otro número que no sea el 0.

23. Escribí un programa que le solicite al usuario un número entero y muestre todos los números correlativos entre el 1 y el número ingresado por el usuario.

Ejemplo de ejecución:

Ingresá un número: *3*
1
2
3

24. Escribí un programa que muestre la sumatoria de todos los números entre el 0 y el 100.

Ejemplo de ejecución:

Sumatoria: 5050

25. Escribí un programa que, dado un número por el usuario, muestre todos sus divisores positivos. Recordá que un divisor es aquel que divide al número de forma exacta (con resto 0).

Ejemplo de ejecución:

Número: *14*
Divisores:
1
2
7
14

```
for x in "hola":
    print(x)
```

La repetición **for** utiliza el operador **in** para recorrer una secuencia. Este operador retorna un valor lógico: True si el primer operando está contenido en el segundo, False si no es así. Es posible utilizar el operador **in** de forma independiente, para saber, por ejemplo, si un string está contenido dentro de otro: **"a" in "hola"** dará como resultado True porque el string "a" está dentro de "hola". Asimismo, se puede negar el valor lógico obtenido por la operación utilizando not: **"a" not in "hola"** dará como resultado Fase porque no es verdad que el string "a" no esté dentro de "hola".

26. Escribí un programa que, dada una frase por el usuario, muestre la cantidad total de vocales (tanto mayúsculas como minúsculas) que contiene.

Ejemplo de ejecución:

Frase: *Verde que te quiero verde*
Vocales: 11

27. Escribí un programa que muestre los primeros 10 números de la sucesión de Fibonacci. La sucesión comienza con los números 0 y 1 y, a partir de éstos, cada elemento es la suma de los dos números anteriores en la secuencia: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55…

Ejemplo de ejecución:

0
1
1

2
3
5
8
13
21
34

28. Escribí un programa que, dado un número entero positivo, calcule y muestre su factorial. El factorial de un número se obtiene multiplicando todos los números enteros positivos que hay entre el 1 y ese número. El factorial de 0 es 1.

Ejemplo de ejecución:

Número: *7*
Factorial: 5040

29. Escribí un programa que permita al usuario ingresar 6 números enteros, que pueden ser positivos o negativos. Al finalizar, mostrar la sumatoria de los números negativos y el promedio de los positivos. No olvides que no es posible dividir por cero, por lo que es necesario evitar que el programa arroje un error si no se ingresaron números positivos.

Ejemplo de ejecución:

Número: *15*
Número: *-12*
Número: *5*
Número: *10*
Número: *-3*
Número: *9*
Sumatoria de los negativos: -15
Promedio de los positivos: 9.75

```
cadena=input("Frase:")
for caracter in cadena:
    print(caracter)
```

Cuando se recorre un string mediante un for que itera por cada uno de sus caracteres, asignar un nuevo valor a la variable iteradora (en este caso, **caracter**) no modifica el string original. Esto es, la instrucción **caracter**=**"a"** dentro del **for** no modificaría en nada el string almacenado en la variable **cadena**.

30. Escribí un programa que permita al usuario ingresar una frase y luego un carácter (string de longitud 1) y luego muestre la frase ingresada, pero con todas las ocurrencias del carácter indicado por el usuario reemplazadas por “*”.

Ejemplo de ejecución:

Frase: *Aquí me pongo a cantar al compás de la vigüela*
Carácter: *o*
Aquí me p*ng* a cantar al c*mpás de la vigüela

31. Escribí un programa que, dada una frase por el usuario, la muestre invertida.

Ejemplo de ejecución:

Frase: *Sabía quién era esta mañana, pero he cambiado varias veces desde entonces*
secnotne edsed secev sairav odaibmac eh orep ,anañam atse are néiuq aíbaS

```
suma=0
numero=int(input("Número:"))
while numero != 0:
    suma=suma+numero
    numero=int(input("Número:"))
print("Suma de todos los números ingresados:", suma)
```

Una repetición **while** necesita una **condición** (en este ejemplo: **numero!**=**0**), que es un valor lógico para saber si continúa repitiendo el bloque o no. Si el valor es True, se ejecuta el bloque correspondiente al while; si es False, no se ejecuta y continúa el programa con la instrucción siguiente al bloque. El código del ejemplo permite ingresar números y cada uno se acumula sumándolo en la variable **suma**, para luego pedir al usuario un nuevo número. Cuando se ingresa el **0**, el bloque del while no se ejecuta y pasa directamente a la siguiente instrucción, que en este caso es un **print** que muestra la suma de todos los números ingresados.

```
numero=int(input("Número:"))
while numero != 0:
    print("Tu número es:", numero)
```

La repetición anterior contiene un error y es que, si el número ingresado por el usuario es diferente de cero, se producirá un bucle infinito que mostrará “Tu número es: [número del usuario]” infinitas veces, sin continuar con el resto del programa. Esto es así porque, en las repeticiones **while**, una vez ejecutado el bloque, se vuelve a evaluar la condición y, si es True, vuelve a ejecutar el bloque y nuevamente a evaluar la condición. En el caso de ingresar un valor distinto de **0** en la variable **numero**, si no se permite cambiar ese valor la condición siempre comparará el mismo número con

el 0 y siempre será True, por lo que nunca dejará de repetir la instrucción print dentro del bloque. Para evitarlo, es necesario permitir cambiar el valor de la variable **numero**:

```
numero=int(input("Número:"))
while numero != 0:
    print("Tu número es:", numero)
    numero=int(input("Número:"))
```

32. Escribí un programa que permita al usuario ingresar los montos de las compras de un cliente (se desconoce la cantidad de datos que cargará, la cual puede cambiar en cada ejecución), cortando el ingreso de datos cuando el usuario ingrese el monto 0. Si ingresa un monto negativo, no se debe procesar y se debe pedir que ingrese un nuevo monto. Al finalizar, informar el total a pagar teniendo que cuenta que, si las ventas superan el monto total de 1000, se le debe aplicar un 10% de descuento.

Ejemplo de ejecución:

Monto de una venta: $ *100*
Monto de una venta: $ *300*
Monto de una venta: $ *-1*
Monto no válido.
Monto de una venta: $ *2000*
Monto de una venta: $ *0*
Monto total a pagar: $ 2160.0

33. Escribí un programa que permita al usuario ingresar una cantidad de números positivos indefinida (la cantidad que ingresará no se conoce y puede cambiar en cada ejecución), finalizando cuando ingresa el número 0 (que no se tendrá en cuenta). Una vez terminada la lectura de números, informar cuál fue el mayor de los números ingresados.

Ejemplo de ejecución:

Número: *6*
Número: *9*
Número: *2*
Número: *12*
Número: *0*
Mayor número ingresado: 12

```
cantidad=0
caracter=input("Carácter:")
while len(caracter) == 1:
    cantidad=cantidad+1
    caracter=input("Carácter:")
```

La repetición **while** ejecutará el bloque siempre que la condición sea **True**. En el ejemplo anterior, se repetirá el bloque mientras los strings ingresados tengan longitud 1 (un único carácter) y la repetición finalizará cuando se ingrese un string con longitud diferente a 1 (ya sea un string vacío o uno que tenga más de 1 carácter).

```
cantidad=0
caracter=input("Carácter:")
while len(caracter) != 1:
    cantidad=cantidad+1
    caracter=input("Carácter:")
```

En este segundo ejemplo se repetirá el bloque mientras los strings ingresados tengan longitud diferente de 1 y la repetición finalizará cuando se ingrese un único carácter.

34. Escribí un programa que pregunte al usuario si desea analizar calificaciones de alumnos y, sólo si responde "S" comenzará el procesamiento de los datos, hasta que el usuario ingrese algo diferente de "S". Por cada alumno, permitir ingresar su calificación. Si es mayor a 4 el alumno está aprobado. Finalmente, mostrar "Porcentaje de alumnos aprobados: x %" (donde *x* es el porcentaje de aprobados sobre el total de calificaciones procesadas). También se debe imprimir "Promedio de los aprobados: y" (donde *y* es la calificación promedio, sólo de los alumnos aprobados).

Ejemplo de ejecución:

```
¿Analizar calificaciones? 'S' para 'sí': S
Calificación de un alumno: 9
¿Continuar? 'S' para 'sí': S
Calificación de un alumno: 4
¿Continuar? 'S' para 'sí': S
Calificación de un alumno: 8
¿Continuar? 'S' para 'sí': N
Porcentaje de alumnos aprobados: 66.66666666666667 %
Promedio de los aprobados: 8.5
```

```
cantidad=0
n=int(input("Número:"))
while n>0 and n%10!=0:
    cantidad=cantidad+1
    n=int(input("Número:"))
```

Una condición tiene que ser un valor lógico, sin importar qué tipo de operación se deba realizar para obtenerlo. En el ejemplo anterior, se ejecutará el bloque mientras los números ingresados sean positivos (mayores que 0) y no sean múltiplos de 10. Cuando el usuario ingrese un número que incumpla alguna de las dos condiciones (un número negativo ó un número múltiplo de 10) el bloque deja de ejecutarse.

35. Escribí un programa que solicite al usuario el ingreso de strings de longitud 1 (un solo carácter), uno por vez. La repetición finalizará cuando se ingrese un string que no tenga longitud 1, o cuando el string ingresado corresponda al dígito numérico 0. Al finalizar, mostrar el string completo que se formó con todos los caracteres ingresados y qué porcentaje de caracteres del total fueron la letra “a”.

Ejemplo de ejecución:

```
Escribí un carácter: L
Escribí un carácter: 9
Escribí un carácter: a
Escribí un carácter: 4
Escribí un carácter: A
Escribí un carácter: 0
Escribí un carácter: N
Escribí un carácter: a
Escribí un carácter: a
Escribí un carácter: 5
El string completo es: L9a4A0Naa
Porcentaje de letras 'a': 33.333333333333336
```

36. Escribí un programa que, dado un número entero por el usuario (guardado como int), muestre la suma de todos sus dígitos. Recordá que vas a necesitar obtener cada uno de los dígitos por separado para poder sumarlos entre sí.

Ejemplo de ejecución:

```
Escribí un número: 7124
Suma de los dígitos: 14
```

37. Escribí un programa para solicitar al usuario que ingrese números enteros positivos (la cantidad que ingresará no se conoce y la decide el usuario). La lectura de números finalizará cuando el usuario ingrese el número -1. Por cada número ingresado, mostrar la cantidad de dígitos pares y la cantidad de dígitos impares que tiene. Al finalizar, mostrar cuántos números múltiplos de 3 ingresó el usuario.

Ejemplo de ejecución:

```
Número (-1 para terminar el programa): 123
Dígitos pares: 1
Dígitos impares: 2
Número (-1 para terminar el programa): 44
Dígitos pares: 2
Dígitos impares: 0
Número (-1 para terminar el programa): 9
```

Dígitos pares: 0
Dígitos impares: 1
Número (-1 para terminar el programa): *-1*
Se ingresaron 2 múltiplos de 3.

38. Escribí un programa que solicite al usuario una cadena de caracteres (que puede contener letras, números o símbolos). Analizar la cadena para mostrar: cuántas letras del abecedario (minúsculas y mayúsculas) contiene, cuántos símbolos (caracteres que no son ni letras ni números), cuántos dígitos numéricos y, de los dígitos, cuántos son múltiplos de 4.

Ejemplo de ejecución:

Cadena de caracteres: *1984 (novela de George Orwell)*
Cantidad de letras: 20
Cantidad de dígitos numéricos: 4
Cantidad de símbolos: 6
Cantidad de múltiplos de 4: 2

39. Escribí un programa que permita al usuario ingresar números que serán leídos como string (no como int o float) hasta que ingrese uno que sea múltiplo de 10 ó menor que 0 (que no será procesado). Se formarán dos strings, en los cuales se concatenarán los números ingresados, según el siguiente criterio: en un string se concatenarán todos los números que el usuario ingrese cuya cantidad de dígitos sea un múltiplo de 3. En el otro, se concatenarán todos los números que contengan el dígito 0. Si un número cumple ambas condiciones, debe concatenarse en ambos strings. En cada string, después de cada número concatenado debe colocarse el carácter “-”. Al finalizar, mostrar en pantalla ambos strings.

Ejemplo de ejecución:

Número: *829*
Número: *102834*
Número: *6*
Número: *4307*
Número: *23*
Número: *1602357*
Número: 5896
Número: *720*
Números cuya cantidad de dígitos es múltiplo de 3: 829-102834-
Números que contienen el 0: 102834-4307-1602357-

40. Escribí un programa que permita al usuario ingresar títulos de libros por teclado, finalizando el ingreso al leerse el string “*” (asterisco). Cada vez que el usuario ingrese un string de longitud 1 que contenga sólo una barra “/” se considera que termina una línea. Por cada línea completa, informar cuántos dígitos numéricos (del 0 al 9) aparecieron en total (en todos los títulos de libros que componen en esa línea). Finalmente, informar cuántas líneas completas se ingresaron.

Ejemplo de ejecución:

Cadena: *Don Quijote de La Mancha*
Cadena: *Los 3 mosqueteros*
Cadena: *Historia de 2 ciudades*
Cadena: */*
Aparecen 2 dígitos en la línea
Cadena: *20000 leguas de viaje submarino*
Cadena: *El señor de los anillos*
Cadena: *Alicia en el país de las maravillas*
Cadena: *1984*
Cadena: *El hobbit*
Cadena: */*
Aparecen 9 dígitos en la línea
Cadena: *Divina comedia*
Cadena: *Drácula*
Cadena: */*
Aparecen 0 dígitos en la línea
Cadena: *20 años después*
Cadena: *Los viajes de Gulliver*
Cadena: *
Se leyeron 3 líneas completas

Sección 3

Funciones

```
def promedio(x, y, z):
    return (x+y+z)/3

n1=int(input("Primer número:"))
n2=int(input("Segundo número:"))
n2=int(input("Tercer número:"))
print("El promedio de los tres es:", promedio(n1,n2,n3))
```

La función **suma** definida en el ejemplo recibe tres parámetros y retorna el promedio.

Una función termina donde termina su bloque ó cuando se ejecuta una instrucción return, aunque esté en cualquier parte del bloque. Al retornar, el programa continúa desde el punto en donde se había llamado a la función.

Una función puede recibir 0 o más parámetros, sin límite, pero no puede retornar más de un resultado a la vez. Puede también no retornar nada (por ejemplo, una función que sólo muestra algo en pantalla).

41. Escribí una función llamada *esPar* que reciba como parámetro un número y retorne True si el número es par ó False si es impar. Utilizar esta función en un programa que solicite al usuario el ingreso de 10 números y que luego muestre, por separado, la suma de todos los pares y la suma de todos los impares.

Ejemplo de ejecución:

Número: *620*
Número: *12993*
Número: *230*
Número: *7*
Número: *18*
Número: *9234*
Número: *38*
Número: 567
Número: *8146*
Número: *32*
Suma de los pares: 18318
Suma de los impares: 13567

42. Escribí una función llamada *sumatoriaDigitos* que reciba como parámetro un número y retorne la suma de todos sus dígitos, reutilizando la estrategia utilizada en el ejercicio 36. Finalmente, escribir un programa que solicite al usuario ingresar varios números hasta que ingrese el número 100, con el cual cortará la repetición. Por cada número, mostrar la suma de sus dígitos, para lo cual se llamará a la función *sumatoriaDigitos*.

Ejemplo de ejecución:

Escribí un número: *7124*
Suma de los dígitos: 14
Escribí un número: *20*
Suma de los dígitos: 2
Escribí un número: *916*
Suma de los dígitos: 16
Escribí un número: *100*

```
def mayorDeDos(x, y):
   if x > y:
       return x
   else:
       return y

def mayorDeTres(x, y, z):
    if mayorDeDos(x,y)==x:
        if mayorDeDos(x,z)==x:
            return x
        else:
            return z
    else:
        if mayorDeDos(y,z)==y:
            return y
        else:
            return z

n1=int(input("Primer número:"))
n2=int(input("Segundo número:"))
n3=int(input("Tercer número:"))
print(mayorDeTres(n1,n2,n3))
```

Un programa puede tener una cantidad ilimitada de funciones y éstas pueden llamarse desde cualquier parte del programa, incluso desde dentro de otras funciones. También es posible llamar a una misma función más de una vez, incluso pasándole diferentes datos como argumentos.

Cada función tiene su propio espacio de memoria, por lo que las variables creadas dentro de ella no existen dentro del resto del programa. Es por esto que podrían utilizarse los mismos nombres para variables en diferentes funciones, pero representarán datos diferentes. Además, aunque el valor de los parámetros se modifique, ésto no modificará los valores de los argumentos en la llamada a la

función. Es por esto que se suele decir que los parámetros son datos “de entrada” y el valor de retorno es un dato “de salida”.

43. Escribí un programa que permita al usuario ingresar números enteros. La repetición terminará cuando el usuario ingrese un número para el cual la suma de sus dígitos sea mayor que 1000 ó múltiplo de 5. Finalmente, mostrar cuántos números impares ingresó el usuario antes de cortar la repetición. Reutilizar las funciones *esPar* y *sumatoriaDigitos* implementadas en los ejercicios anteriores.

Ejemplo de ejecución:

Escribí un número: *16*
Escribí un número: *922*
Escribí un número: *1513*
Escribí un número: *481*
Escribí un número: *90*
Cantidad de impares: 2

44. Escribí una función que reciba un string y retorne True si es un palíndromo (esto es, si se lee igual de izquierda a derecha o de derecha a izquierda), False en caso contrario. Utilizar esta función en un programa que permita al usuario ingresar palabras hasta que ingrese la palabra “fin” (suponer que todas las palabras son en minúsculas o todas en mayúsculas, de forma consistente). Al finalizar, mostrar la cantidad de palíndromos ingresados.

Ejemplo de ejecución:

Cadena: *abba*
Cadena: *m*
Cadena: *luz*
Cadena: *reconocer*
Cadena: *golondrina*
Cadena: *fin*
Cantidad de palíndromos: 3

```
def cantidadDigitos(numero):
    cantidad=0
    while numero!=0:
        cantidad=cantidad+1
        numero=numero//10
    return cantidad

mayor=-1
n=int(input("Número positivo:"))
```

```
while cantidadDigitos(n)%3 != 0:
    if n>mayor:
        mayor=n
    n=int(input("Número positivo:"))
print("Mayor número ingresado:", mayor)
```

Las funciones son también útiles para pensar los programas en partes más pequeñas. En el ejemplo de arriba, la repetición lee números hasta que se ingresa uno cuya cantidad de dígitos es múltiplo de 3. Cuando finaliza la repetición, de todos los números ingresados se muestra cuál fue el mayor. Pero para poder armar la condición de esta repetición es necesario calcular la cantidad de dígitos de un número, lo cual se delega en una función que es llamada en cada iteración, cuando se evalúa la condición.

45. Escribí un programa que permita al usuario ingresar números enteros hasta que ingrese uno cuyo dígito inicial sea el 9 (el cual no se procesará). Una vez terminada la repetición, mostrar cuántos de los números que el usuario ingresó tienen sólo dos divisores (para esto es posible reutilizar parte de la estrategia elaborada en el ejercicio 25).

Ejemplo de ejecución:

Número entero: *167*
Número entero: *11*
Número entero: *821*
Número entero: *38*
Número entero: *292*
Número entero: *3*
Número entero: *954*
Tienen sólo 2 divisores: 4 números

Soluciones a los ejercicios

Sección 1: Entrada/salida de datos - Variables - Tipos de datos

1.
```
nombre=input("Tu nombre:")
print("Ahora estás en la matrix,", nombre)
```

2.
```
a=float(input("Primer número:"))
b=int(input("Segundo número:"))
c=a+b
print("El resultado de la suma es", c)
```

3.
```
n1=int(input("Ingresá un número:"))
n2=int(input("Ingresá otro número:"))
suma=n1+n2
print("Suman:", suma)
n3=int(input("Ingresá un nuevo número:"))
print("Multiplicación de la suma por el último número:",suma*n3)
```

4.
```
kilometros=float(input("Kilómetros recorridos:"))
litros=float(input("Litros de combustible gastados:"))
print("El consumo por kilómetro es de", kilometros/litros)
```

5.
```
Fahrenheit=float(input("Ingresá una temperatura expresada en Fahrenheit:"))
print((5/9) * (Fahrenheit-32))
```

6.
```
n1=float(input("Primer número:"))
n2=float(input("Segundo número:"))
n3=float(input("Tercer número:"))
print("El promedio de los tres es", (n1+n2+n3)/3)
```

7.
```
numero=int(input("Ingresá un número:"))
print("Descontando el 15% queda:", numero-(numero*15)/100)
```

8.

```
palabra1=input("Primera palabra:")
palabra2=input("Segunda palabra:")
frase=palabra1+" "+palabra2
print(frase)
```

9.

```
cadena=input("Ingresá un texto:")
print("El carácter en primer lugar es:", cadena[0])
print("Ingresá un número positivo menor a", len(cadena))
indice=int(input())
print("El carácter en esa posición es:", cadena[indice])
```

10.

```
shows=int(input("Shows vistos en el último año:"))
print(shows>3)
```

11.

```
fecha=int(input("Fecha en formato DDMMAAAA:"))
año=fecha%10000
dia=fecha//1000000
mes=(fecha//10000)%100
print(dia,"/",mes,"/",año)
```

12.

```
numero=int(input("Número entero:"))
print((numero%2) == 0)
```

13.

```
edad=int(input("Tu edad:"))
articulos=int(input("Artículos comprados:"))
print((edad>18) and (articulos>1))
```

14.

```
cadena=input("Ingresá una frase:")
longitud=len(cadena)
print(longitud%2 == 0)
```

15.

```
palabra1=input("Una palabra:")
palabra2=input("Otra palabra:")
print(palabra1<palabra2)
```

16.

```
nombre1=input("Tu nombre:")
nombre2=input("Otro nombre:")
posicion_final_nombre1=len(nombre1)-1
posicion_final_nombre2=len(nombre2)-1
print((nombre1[0] == nombre2[0]) or (nombre1[posicion_final_nombre1] ==
nombre2[posicion_final_nombre2]))
```

Sección 2: Bloques - Selección - Repeticiones

17.

```
numero=int(input("Número:"))
if numero<0:
    numero=numero*-1
print("Valor absoluto:", numero)
```

18.

```
numero1=int(input("Un número:"))
numero2=int(input("Otro número distinto:"))
if numero1>numero2:
    print(numero1, "es mayor")
else:
    print(numero2, "es mayor")
```

19.

```
letra=input("Letra:")
if len(letra)!=1:
    print("Debe ser sólo una letra")
else:
    if letra=="a" or letra=="e" or letra=="i" or letra=="o" or letra=="u":
        print("Es vocal")
```

20.

```
n1=int(input("Primer número:"))
n2=int(input("Segundo número:"))
n3=int(input("Tercer número:"))
if n1<n2:
    if n1<n3:
        print("Menor:", n1)
    else:
        print("Menor:", n3)
```

```
else:
    if n2<n3:
        print("Menor:", n2)
    else:
        print("Menor:", n3)
```

21.

```
nombre=input("Nombre de usuario:")
password=input("Contraseña:")
if nombre=="Gwenevere" and password =="excalibur":
    print("Usuario y contraseña correctos. Puede ingresar a Camelot")
else:
    print("Acceso denegado")
```

22.

```
anio=int(input("Año:"))
if anio%4 == 0:
    if anio%100 != 0 or anio%400 == 0:
        print("Bisiesto")
    else:
        print("No bisiesto")
else:
    print("No bisiesto")
```

23.

```
numero=int(input("Ingresá un número:"))
for i in range(1,numero+1):
   print(i)
```

24.

```
total=0
for i in range(101):
    total=total+i
print("Sumatoria:", total)
```

25.

```
numero=int(input("Número:"))
print("Divisores:")
for n in range(1,numero+1):
    if numero%n == 0:
        print(n)
```

26.

```
frase=input("Frase:")
vocales="aeiou"
cantidad=0
for c in frase:
    if c in vocales:
        cantidad=cantidad+1
print("Vocales:", cantidad)
```

27.

```
n1=0
n2=1
print(n1)
print(n2)
for i in range(8):
    n3=n1+n2
    print(n3)
    n1=n2
    n2=n3
```

28.

```
numero=int(input("Número: "))
f=1
if numero!=0:
   for i in range(1,numero+1):
       f=f*i
print("Factorial:", f)
```

29.

```
sumaPositivos=0
cantidadPositivos=0
sumaNegativos=0
for i in range(6):
   nro=int(input("Número: "))
   if nro>0:
       sumaPositivos=sumaPositivos+nro
       cantidadPositivos=cantidadPositivos+1
   else:
       sumaNegativos=sumaNegativos+nro
print("Sumatoria de los negativos: ", sumaNegativos)
if cantidadPositivos!=0:
   print("Promedio de los positivos: ",sumaPositivos/cantidadPositivos)
```

30.

```
frase=input("Frase:")
caracter=input("Carácter:")
nueva=""
for c in frase:
    if c==caracter:
        nueva=nueva+"*"
    else:
        nueva=nueva+c
print(nueva)
```

31.

```
frase=input("Frase:")
nueva=""
i=len(frase)-1
while i>=0:
    nueva=nueva+frase[i]
    i=i-1
print(nueva)
```

32.

```
total=0
monto=float(input("Monto de una venta: $"))
while monto!=0:
    if monto<0:
        print("Monto no válido.")
    else:
        total=total+monto
    monto=float(input("Monto de una venta: $"))
if total>1000:
    total=total-(total*0.1)
print("Monto total a pagar: $", total)
```

33.

```
mayor=-1
n=int(input("Número positivo:"))
while n!=0:
    if n>mayor:
        mayor=n
    n=int(input("Número positivo:"))
print("Mayor número ingresado:", mayor)
```

34.

```
aprobados=0
cantidad=0
sumaAprobados=0
a=input("¿Analizar calificaciones? 'S' para 'sí':")
while a == "S":
    calificacion=int(input("Calificación de un alumno:"))
    if calificacion > 4:
        aprobados=aprobados+1
        sumaAprobados=sumaAprobados+calificacion
    cantidad=cantidad+1
    a=input("¿Continuar? 'S' para 'sí':")
print("Porcentaje de alumnos aprobados:", (aprobados*100)/cantidad, "%")
print("Promedio de los aprobados:", sumaAprobados/aprobados)
```

35.

```
cadenaTotal=""
cantidad_a=0
caracter=input("Escribí un carácter:")
while (len(caracter)==1 and caracter!="0"):
    cadenaTotal=cadenaTotal+caracter
    if caracter=="a":
        cantidad_a=cantidad_a+1
    caracter=input("Escribí un carácter:")
print("El string completo es:", cadenaTotal)
print("Porcentaje de letras 'a':", (cantidad_a*100)/len(cadenaTotal))
```

36.

```
numero=int(input("Escribí un número:"))
total=0
while numero != 0:
    ultimoDigito=numero%10
    total=total+ultimoDigito
    numero=numero//10
print("Suma de los dígitos:", total)
```

37.

```
multiplosDe3=0
numero=int(input("Número (-1 para terminar el programa):"))
while numero!=-1:
    if numero%3 == 0:
        multiplosDe3=multiplosDe3+1
```

```
    digitosPares=0
    digitosImpares=0
    while numero!=0:
        ultimodigito=numero%10
        if ultimodigito%2==0:
            digitosPares=digitosPares+1
        else:
            digitosImpares=digitosImpares+1
        numero=numero//10
    print("Dígitos pares:", digitosPares)
    print("Dígitos impares:", digitosImpares)
    numero=int(input("Número (-1 para terminar el programa):"))
print("Se ingresaron", multiplosDe3, "múltiplos de 3.")
```

38.

```
cadena=input("Cadena de caracteres:")
letras="abcdefghijklmnñopqrstuvwxyzABCDEFGHIJKLMNÑOPQRSTUVWXYZ"
digitos="0123456789"
cantidadLetras=0
cantidadDigitos=0
cantidadSimbolos=0
cantidadMultiplos4=0
for i in cadena:
    if i in letras:
        cantidadLetras=cantidadLetras+1
    else:
        if i in digitos:
            cantidadDigitos=cantidadDigitos+1
            if int(i)%4 == 0:
                cantidadMultiplos4=cantidadMultiplos4+1
        else:
            cantidadSimbolos=cantidadSimbolos+1
print ("Cantidad de letras:", cantidadLetras)
print ("Cantidad de dígitos numéricos:", cantidadDigitos)
print ("Cantidad de símbolos:", cantidadSimbolos)
print ("Cantidad de múltiplos de 4:", cantidadMultiplos4)
```

39.

```
longitudes=""
digito0=""
numero=input("Número:")
while int(numero)%10 != 0 and int(numero) >= 0:
```

```
    if len(numero)%3 == 0:
        longitudes=longitudes+numero+"-"
    if "0" in numero:
        digito0=digito0+numero+"-"
    numero=input("Número:")
print("Números cuya cantidad de dígitos es múltiplo de 3:", longitudes)
print("Números que contienen el 0:", digito0)
```

40.

```
lineas=0
digitos="0123456789"
cantidadDigitos=0
cadena=input("Cadena: ")
while cadena!="*":
    for caracter in cadena:
        if caracter in digitos:
            cantidadDigitos+=1
    if cadena=="/":
        lineas+=1
        print("Aparecen ", cantidadDigitos, " dígitos en la línea")
        cantidadDigitos=0
    cadena=input("Cadena: ")
print("Se leyeron ",lineas," líneas completas")
```

Sección 3: Funciones

41.

```
def esPar(numero):
    return numero%2 == 0

sumaPares=0
sumaImpares=0
for i in range(10):
    num=int(input("Número:"))
    if esPar(num):
        sumaPares=sumaPares+1
    else:
        sumaImpares=sumaImpares+1
print("Suma de los pares:", sumaPares)
print("Suma de los impares:", sumaImpares)
```

42.

```
def sumatoriaDigitos(numero):
    total=0
    while numero != 0:
        ultimoDigito=numero%10
        total=total+ultimoDigito
        numero=numero//10
    return total

n=int(input("Escribí un número:"))
print("Suma de los dígitos:", sumatoriaDigitos(n))
```

43.

```
def esPar(numero):
   return numero%2 == 0

def sumatoriaDigitos(numero):
   total=0
   while numero != 0:
       ultimoDigito=numero%10
       total=total+ultimoDigito
       numero=numero//10
   return total

cantidadImpares=0
n=int(input("Escribí un número:"))
while sumatoriaDigitos(n)<1000 and sumatoriaDigitos(n)%3!=0:
    if not esPar(n):
        cantidadImpares=cantidadImpares+1
    n=int(input("Escribí un número:"))
print("Cantidad de impares:", cantidadImpares)
```

44.

```
def palindromo(cadena):
    if len(cadena)==1:
        return True
    else:
        i = 0
        d = len(cadena)-1
        while (d > i):
            if (cadena[i] != cadena[d]):
```

```
                return False
            i=i+1
            d=d-1
        return True

cantidad=0
cadena=input("Cadena:")
while cadena != "fin":
    if palindromo(cadena):
        cantidad=cantidad+1
    cadena=input("Cadena:")
print("Cantidad de palíndromos:", cantidad)
```

45.

```
def primerDigito(numero):
    while numero//10 != 0:
        numero=numero//10
    return numero

def cantidadDivisores(numero):
    cantidad=0
    for n in range(1,numero+1):
        if numero%n == 0:
            cantidad=cantidad+1
    return cantidad

cantidad=0
n=int(input("Número entero:"))
while primerDigito(n)!=9:
    if cantidadDivisores(n)==2:
        cantidad=cantidad+1
    n=int(input("Número entero:"))
print("Tienen sólo 2 divisores:", cantidad, "números")
```