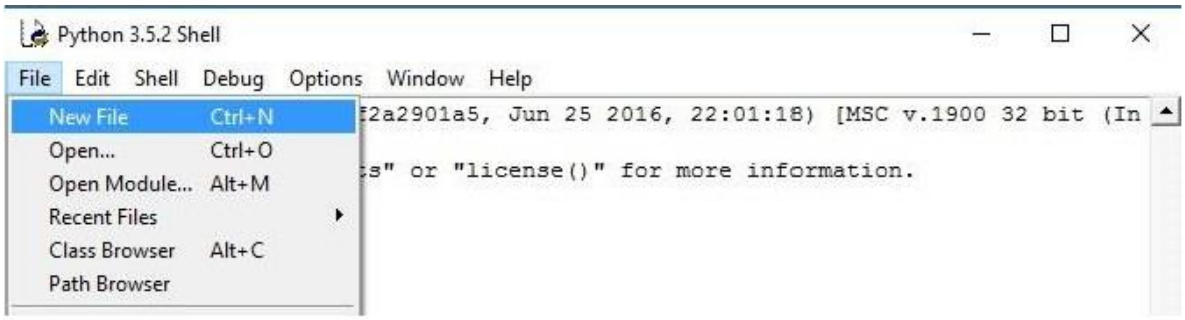
Trabajo Práctico N°2

Estructuras de decisión

Ejercicio 2 – Positivo o Negativo

1. El objetivo del programa es contestar si un número es Positivo, negativo o igual a cero.
2. Abrir el programa IDLE (Python 3.5)
3. Observar que aparece el prompt esperando ingreso de datos.
4. Ingresar al menú **File → New File** o **Ctrl+N**

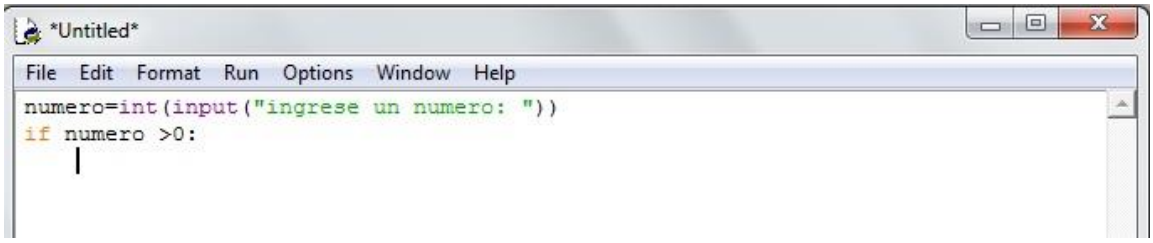


5. Se abrirá una nueva pantalla en blanco

6. Cree un archivo, llamado PositivoNegativo.
7. Dentro del archivo cree la sentencia para leer desde el teclado.
8. Solicite al usuario que ingrese un número y guárdelo en una variable de tipo entera llamada número mediante la conversión de tipos.
9. Dentro del nuevo archivo, agregue la siguiente línea que permite al programa comprobar si se cumple la condición solicitada:

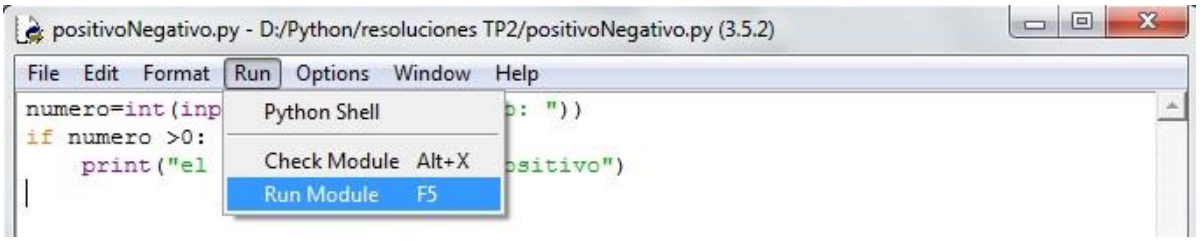
```
if numero>0:
```

10. Luego presione la tecla **Enter**. Observe que automáticamente se produce una tabulación para poder ingresar la sentencia a realizar en caso de cumplirse la condición.

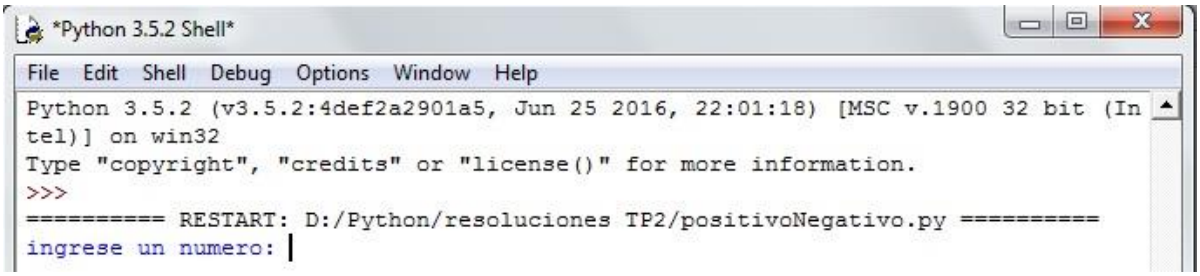


11. Luego de ingresar la sentencia presione nuevamente la tecla **Enter**. Elimine la nueva tabulación producida y continúe con el programa.

12. Determine si el número es Positivo, Negativo o Cero mediante los comparadores <, > o ==.
13. Muestre el resultado en la salida estándar mediante un mensaje.
14. Guarde el código del programa con **File → Save** o **Ctrl+S** con el nombre PositivoNegativo.py (recuerde que el programa debe tener extensión .py o Tipo "Python Files", de lo contrario no funcionará).
    No se olvide de seleccionar la carpeta donde desee que se guarde el archivo (**Recomendación:** Modificar la que aparece por defecto).
15. Ejecute el programa con **F5** o con **Run → Run Module**



16. Se abrirá nuevamente la pantalla principal del IDLE (donde figura el prompt >>>).
17. A continuación comenzará a ejecutarse su programa de color azul.



Ejercicio 2 – Par o Impar

1. El objetivo del programa es contestar si un número es par o impar.
2. Cree un archivo, llamado ParImpar.
3. Dentro del archivo cree la sentencia para leer desde el teclado.
4. Solicite al usuario que ingrese un número y guárdelo en una variable de tipo entera llamada numero mediante la conversión de tipos.
5. Determine si el número es par o impar utilizando el operador módulo %
6. Muestre el resultado en la salida estándar mediante un mensaje.
7. Observe que este ejercicio puede realizarse mediante una estructura if…else.
8. Luego de ingresar la primer condición `if`, la tecla **Enter** y su posterior sentencia observe que automáticamente se mantiene tabulación para poder ingresar la sentencia `else` a realizar en caso de cumplirse la condición.
9. Ingrese la nueva comparación correspondiente mediante la siguiente línea:

```
else:
    print("El número es impar")
```

10. Presione **Enter** nuevamente y elimine la tabulación automática.
11. Guarde el archivo.
12. Ejecute.

Ejercicio 3 – Divisible

1. El objetivo del programa es determinar si un número es divisible por otro.
2. Cree un archivo, llamado Divisible.
3. Dentro del archivo cree la sentencia para leer desde el teclado.
4. Solicite al usuario que ingrese dos números y guárdelos en variables de tipo enteras mediante la conversión de tipos.
5. Calcule si el primer número es divisible por el segundo, utilizando el operador módulo %.
6. Muestre el resultado mediante un mensaje.
7. Guarde el archivo.
8. Ejecute.

Ejercicio 4 – Palabra más larga

1. El objetivo del programa es determinar cuál de dos palabras es la más larga.
2. Cree un archivo llamado Palabras.
3. Solicite al usuario que ingrese dos palabras por separado.
4. Determine la longitud de las mismas utilizando la funcion len().
5. Compare dichas longitudes e indique cual es mayor mediante un mensaje.
6. Muestre el resultado.
7. Guarde el archivo.
8. Ejecute.

Ejercicio 5 – Años

1. El objetivo del programa es determinar cuántos años han pasado o cuánto falta para llegar a ese año, contemplando la posibilidad que puede ser el mismo año.
2. Cree un archivo llamado años.
3. Solicite al usuario que ingrese el qué año estamos.
4. Solicite al usuario que ingrese el qué año al cual quiere llegar.
5. Compare dichos años e indique cuántos años han pasado mediante una resta.
6. Muestre el resultado.
7. Guarde el archivo.
8. Ejecute.

Ejercicio 6 – Día de la semana

1. El objetivo del programa es mostrar el día de la semana a partir de un número
2. Cree un archivo llamado DiaSemana
3. Solicite al usuario que ingrese un número entre 1 y 7.
4. Muestre el nombre del día de la semana, teniendo en cuenta que 1=Domingo, 2=Lunes, etc.
5. Si el usuario ha ingresado un número no válido, muestre un mensaje de error.
6. Guarde el archivo.
7. Ejecute el programa.

Ejercicio 7 – Semáforo

1. El objetivo del programa es mostrar la instrucción correspondiente de acuerdo al color ingresado
2. Cree un archivo llamado Semaforo
3. Solicite al usuario que ingrese un color: Verde, Amarillo o Rojo.
4. Muestre la instrucción correspondiente, teniendo en cuenta que Verde=Avanzar, Amarillo=Precaución, Rojo=Detenerse
5. Si el usuario ha ingresado un color no válido, muestre un mensaje de error.

6. Guarde el archivo.
7. Ejecute el programa.

Ejercicio 8 – Hora del día

1. El objetivo del programa es mostrar una franja horaria a partir de la hora del día
2. Cree un nuevo archivo, llamado HoraDia.
3. Solicite al usuario un número entre 0 y 23, representando la hora del día.
4. Muestre al usuario los siguientes valores:
    - Si el valor está entre 7 y 12: Mañana
    - Si el valor está entre 13 y 19: Tarde
    - Si el valor está entre 20 y las 1 hs: Noche
    - Si el valor está entre 2 y 6: Madrugada
5. Guarde el archivo.
6. Ejecute el programa.

Ejercicio 9 – Número aleatorio

1. El objetivo del programa es generar un número aleatorio que el usuario deberá adivinar.
2. Cree un nuevo archivo, llamado NumeroAleatorio
3. Genere un número aleatorio del 1 al 10 guardándolo en la variable n con el siguiente código:

```
n=random.randint(1, 10).
```

La función `randint` es una función inluida en la librería random de Python.
Para ello debemos colocar en la primer línea del programa la siguiente sentencia:

```
import random
```

Ahi le indicaremos al programa que vamos a utilizar alguna función de dicha librería.

4. Solicite al usuario que ingrese un número entre el 1 y el 10.
5. Si el número ingresado por el usuario coincide con el generado:

a) Mostrar al usuario un mensaje que indique que adivinó el número generado.

6. Si el número ingresado por el usuario no coincide con el generado:

b) Mostrar al usuario un mensaje que indique que no adivinó el número generado.

7. Guarde el archivo.
8. Ejecute el programa.

Ejercicio 10 – Calculo de salario

1. El objetivo del programa es calcular el salario semanal de un obrero
2. Cree un nuevo archivo CalculoSalario
3. Solicite al usuario que ingrese las horas trabajadas.
4. El cálculo del salario se realiza de la siguiente manera:
    - Si trabaja 40 horas o menos se le paga $16 por hora
    - Si trabaja más de 40 horas se le paga $16 por cada una de las primeras 40 horas y $20 por cada hora extra.
5. Muestre el salario correspondiente.
6. Guarde el archivo.
7. E el programa.

Ejercicio 11 – Calculo de descuentos

8. El objetivo del programa es calcular el los descuentos correspondientes en una tienda

9. Cree un nuevo archivo llamado CalculoDescuentos
10. Solicite al usuario que ingrese el precio del producto comprado y la cantidad comprada de cada uno.
11. El cálculo de descuentos se realiza de la siguiente manera:
    - Por compras menores a $50 se hace un descuento de 8%
    - Para compras a partir de $50 el descuento es de 10%.
12. Calcular el descuento a realizar.
13. Muestre el importe total a pagar.
14. Muestre el descuento realizado
15. Guarde el archivo.
16. Ejecute el programa.

Ejercicio 12 – Menú de conversión

1. El objetivo del programa es convertir una cantidad de bytes a una unidad elegida por el usuario.
2. Cree un nuevo archivo llamado Conversion
3. Solicite al usuario que ingrese un número entero que será la cantidad de bytes
4. Muestre al usuario un menú con las siguientes opciones

Convertir a unidad:

1) Kilobytes
2) Megabytes
3) Gigabytes
4) Terabytes
5) Salir

Recuerde que para mostrar una línea debajo de la otra debe anexar `\n` al final de cada línea.

5. Lea la elección del usuario con una variable de tipo int. Asegúrese de que la opción ingresada es correcta, no permitiendo continuar hasta que se haya ingresado un valor válido.
6. Si el usuario quiere salir, cortar la ejecución del programa con la sentencia
   ```
   exit()
   ```
7. Convierta la cantidad de bytes a la unidad elegida, teniendo en cuenta que cada unidad es un múltiplo de 1024 con respecto a la anterior.
    - 1 kilobyte = 1024 bytes
    - 1 Megabyte = 1024 kilobytes
    - 1 Gigabyte= 1024 Megabytes
    - 1 Terabyte = 1024 Gigabytes.
8. Muestre el resultado.
9. Guarde el archivo.
10. Ejecute el programa.

Estructuras de iteración

Ejercicio 13 – Tablas de Multiplicar

11. El objetivo del programa es mostrar la tabla de multiplicación de un número.
12. Cree un nuevo archivo, llamada Tablas
13. Solicite al usuario un número para calcular su tabla de multiplicar.
14. Solicite al usuario un número para la cantidad de multiplicaciones a mostrar.

15. Muestre la tabla del primer número, multiplicado por la cantidad de veces que indica el segundo número. Por ejemplo, si el usuario ingresó 4 y 8, el programa deberá mostrar la tabla del 4, desde el 1 al 8 (4x1=4, 4x2=8, 4x3=12, etc).
16. Guarde el archivo.
17. Ejecute el programa

Ejercicio 14 – Ancho

1. El objetivo del programa es dibujar una línea con el carácter asterisco.
2. Cree un nuevo archivo, llamado linea.
3. Solicite al usuario que ingrese el ancho de una línea y guárdelo en una variable.
4. Utilice un bucle para mostrar el la línea utilizando el carácter * con la siguiente línea:

```
for i in range(ancho):
```

Donde el argumento pasado entre paréntesis es la cantidad de veces que deseamos que se ejecute el bucle.

5. Por ejemplo, si el usuario ingresó 7, el programa mostrará

```
*******
```

6. Guarde el archivo.
7. ejecute el programa.

Ejercicio 15 – Rectángulo

1. El objetivo del programa es dibujar un rectángulo con el carácter asterisco.
2. Cree un nuevo archivo, llamado Rectangulo.
3. Solicite al usuario que ingrese dos números para la base y la altura del rectángulo.
4. Utilice dos bucles anidados para mostrar el rectángulo utilizando el carácter *
5. Por ejemplo, si el usuario ingresó 7 y 3, el programa mostrará

```
*******
*******
*******
```

6. Para mostrar un texto y saltar a la siguiente línea, utilice la siguiente variación de código:

```
print "*",
      print
```

7. Guarde el archivo.
8. Compile y ejecute el programa.

Ejercicio 16 – Triángulo

1. El objetivo del programa es dibujar un Triángulo con el carácter asterisco.
2. Cree un nuevo archivo, llamado Triangulo.
3. Solicite al usuario que ingrese un número para la altura del triángulo.
4. Utilice dos bucles anidados para mostrar la figura utilizando el carácter *
5. Por ejemplo, si el usuario ingresó 4, el programa mostrará

```
*
**
***
****
```

6. Guarde el archivo.
7. Compile y ejecute el programa.

Ejercicio 17 – Número aleatorio 2

1. El objetivo del programa es mejorar el programa del número aleatorio que el usuario debía adivinar, brindándole hasta 5 intentos.
2. Cree un nuevo archivo llamado NumeroAleatorio2
3. Genere un número aleatorio del 1 al 10 guardándolo en la variable n con el siguiente código:

```
n=random.randint(1, 10)
```

4. Solicite al usuario que ingrese un número entre el 1 y el 10.
5. Si el número ingresado por el usuario coincide con el generado:

a) Mostrar al usuario un mensaje que indique que adivinó el número generado.

6. Si el número ingresado por el usuario no coincide con el generado:

b) Mostrar al usuario un mensaje que indique que no adivinó el número generado.

7. Guarde el archivo.
8. Ejecute el programa.

Ejercicio 18 – Factorial

1. El objetivo del programa es calcular el factorial de un número ingresado por el usuario.
2. Cree un nuevo archivo llamado Factorial
3. Solicite al usuario que ingrese un número entero
4. Calcule el factorial del número ingresado utilizando una estructura **while** y la multiplicación aritmética mediante la siguiente línea:

```
factorial = factorial * (termino -1)
```

5. Observe que se llama a la recursión de la operación mediante la invocación del mismo nombre de la variable.
6. Decremente la variable que lleve la cuenta de los pasos realizados.
7. Muestre el resultado.
8. Guarde el archivo.
9. Ejecute el programa.

Ejercicio 19 – Resto de división

1. El objetivo del programa es calcular el resto de una división entre dos números ingresados por el usuario.
2. Cree un nuevo archivo llamado Resto
3. Solicite al usuario que ingrese dos números enteros
4. Utilice una estructura **while** para realizar restas sucesivas entre el primer y el segundo número
5. Realice el decremento del contador añadiendo la siguiente línea:

```
numero1 -=numero2
```

   Observe que esto es lo mismo que realizar numero1 = numero1 – numero2
6. Obtenga el resto cuando ya no sea posible seguir restando sin obtener un resultado negativo.
7. Muestre el resultado.
8. Guarde el archivo.
9. Ejecute el programa.

Ejercicio 20 – Producto

1. El objetivo del programa es calcular el resto de una división entre dos números ingresados por el usuario.
2. Cree un nuevo archivo llamado Producto

3. Solicite al usuario que ingrese dos números enteros
4. Inicialice un contador.
5. Utilice una estructura **while** para realizar sumas sucesivas entre el primer y el segundo número
6. Obtenga el producto cuando ya no sea posible seguir restando sin obtener un resultado negativo.
7. Muestre el resultado.
8. Guarde el archivo.
9. Ejecute el programa.

Ejercicio 21 – Sucesión de Fibonacci

1. El objetivo del programa es mostrar la sucesión de Fibonacci hasta la cantidad de términos ingresados por el usuario.
2. Cree un nuevo archivo, llamado Fibonacci
3. Solicite al usuario que ingrese un número entero.
4. Muestre la cantidad de términos de la sucesión de Fibonacci según el número ingresado.
   Recuerde que cada término de la sucesión se calcula a partir de la suma de los dos términos anteriores, a excepción de los dos primeros términos que son siempre 1 y 1.
5. Para realizar este cálculo necesitará ingresar 3 condiciones mediante el siguiente código:

```
if (condición1):
      acción1
elif(condicion2):
      acción2
else:
      acción3
```

6. Guarde el archivo.
7. Compile y ejecute el programa.

Ejercicio 22 – Suma de Impares

1. El objetivo del programa es determinar la sumatoria de números impares en el rango elegido por el usuario.
2. Cree un archivo, llamado SumatoriaImpares.
3. Dentro del archivo cree la sentencia para leer desde el teclado.
4. Solicite al usuario que ingrese dos números y guárdelos en variables de tipo enteras mediante la conversión de tipos.
5. Declare una variable auxiliar que sirva para llevar el resultado.
6. Recorra los números deseados mediante la siguiente línea:

```
for i in range(numero1,numero2):
```

7. Calcule la sumatoria correspondiente.
8. Muestre el resultado mediante un mensaje.
9. Guarde el archivo.
10. Ejecute.

Ejercicio 23 – Promedio

1. El objetivo del programa es determinar el promedio de calificaciones de un alumno en la facultad

2. Cree un archivo, llamado Promedio.
3. Dentro del archivo cree la sentencia para leer desde el teclado.
4. Solicite al usuario que el número de materias rendidas y guárdelo en una variable de tipo entera mediante la conversión de tipos.
5. Recorra las materias ingresadas
6. Solicite la nota de cada materia.
7. Declare una variable auxiliar que sirva para llevar la sumatoria.
8. Calcule el promedio dividiendo la sumatoria de notas por la cantidad de materias.
9. Muestre el resultado mediante un mensaje.
10. Guarde el archivo.
11. Ejecute.

Ejercicio 24 – Promedio 2

1. El objetivo del programa es mejorar el programa anterior, llevando además la cuenta de materias desaprobadas, regulares y aprobadas de un alumno en la facultad
2. Cree un archivo, llamado PromedioYRegulares.
3. Dentro del archivo cree la sentencia para leer desde el teclado.
4. Solicite al usuario que el número de materias rendidas y guárdelo en una variable de tipo entera mediante la conversión de tipos.
5. Recorra las materias ingresadas.
6. Solicite la nota de cada materia.
7. Declare cuatro variables auxiliares que sirvan para llevar la sumatoria de notas, la cantidad de materias desaprobadas, regulares y la cantidad de promocionadas.
8. Calcule el promedio dividiendo la sumatoria de notas por la cantidad de materias.
9. Calcule la cantidad de materias con nota <= 3 (desaprobadas)
10. Calcule la cantidad de materias con nota >= 4 (regulares)
11. Calcule la cantidad de materias con nota >=7 (promocionadas)
12. Muestre el resultado mediante un mensaje.
13. Guarde el archivo.
14. Ejecute.

Ejercicio 25 – Contador de caracteres

1. El objetivo del programa es contar cuántas veces aparece un carácter en una cadena de texto ingresada por el usuario.
2. Cree un nuevo archivo llamado ContadorCaracteres
3. Inicialice una variable contador en 0
4. Solicite al usuario que ingrese un texto y guárdelo en una variable llamada cadena
5. Solicite al usuario que ingrese un carácter y guárdelo en una variable llamada caracter.
6. Transite la longitud de cadena con una estructura for que recorra la palabra ingresada mediante el siguiente código:
```
for letra in cadena:
```
7. Compare el carácter de cada posición con el ingresado utilizando el siguiente código:
```
if letra == caracter:
```
8. Cuente con el contador la cantidad de veces que la variable caracter es igual a la variable ingresada.
    - Si es igual, sume 1 al contador

9. Muestre el resultado.
10. Guarde el archivo.
11. Ejecute el programa.
12. Pruebe con cadenas de diferente longitud.

Ejercicio 26 – Contador de palabras

1. El objetivo del programa es contar la cantidad de palabras en un texto ingresado por el usuario.
2. Cree un nuevo archivo llamado ContadorPalabras
3. Solicite al usuario que ingrese un texto y guárdelo en una variable llamada texto
4. Recorra la longitud de la cadena y cuente la cantidad de veces que se termina una palabra, tomando como fin el carácter espacio ' '
5. Muestre el resultado.
6. Guarde el archivo.
7. Compile y ejecute el programa.