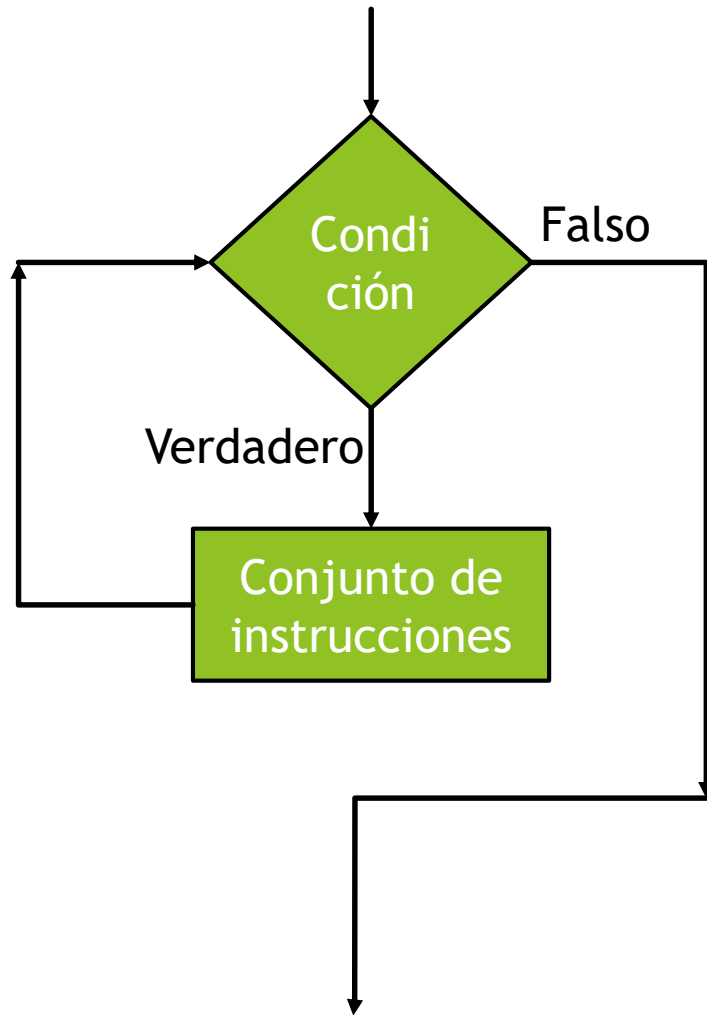


Iteración: *while*



- Mientras el resultado de evaluar la condición sea *verdadero* se ejecuta el conjunto de instrucciones.
- Si al evaluar la condición el resultado es falso se termina la iteración.

Iteración: *while*

Sintaxis

```
while condicion:  
    instruccion  
    ...
```

Nota: existen espacios delante de la/las instrucción/es. Esto se llama indentación. Por lo tanto, el/las instrucción/es que estén dentro del *while*, deben estar indentadas (Python lo realiza automáticamente)

Iteración: *while*

```
while condicion:  
    instruccion  
    ...
```

Por ejemplo, el siguiente programa escribe los números del 1 al 10:

```
i=1  
while i<11:  
    print(i)  
    i = i+1
```

Instrucciones que
se ejecutarán si
condición se cumple

condición:

“Mientras i sea menor que 11”

Ejecutando paso a paso con Thonny

Thonny - C:\Users\usuario\Documents\Asignaturas\Taller ...

Archivo Editar Ver Ejecutar Device Herramientas Ayuda

ejemplowhile1.py <<< REPLAYING >>>

```
1 i=1
2 while i<11
3     print("i=",i)
4     i=i+1
```

Variables

| Name | Value |
|------|-------|
| i | 2 |

Shell

```
>>> %Debug ejemplowhile1.py
i= 1
```

Thonny - C:\Users\usuario\Documents\Asignaturas...

Archivo Editar Ver Ejecutar Device Herramientas Ayuda

ejemplowhile1.py <<< REPLAYING >>>

```
1 i=1
2 while i<11
3     print("i=",i)
4     i=i+1
```

Variables

| Name | Value |
|------|-------|
| i | 5 |

Shell

```
i= 1
i= 2
i= 3
i= 4
```

Thonny - C:\Users\usuario\Documents\Asignaturas...

Archivo Editar Ver Ejecutar Device Herramientas Ayuda

ejemplowhile1.py

```
1 i=1
2 while i<11
3     print("i=",i)
4     i=i+1
```

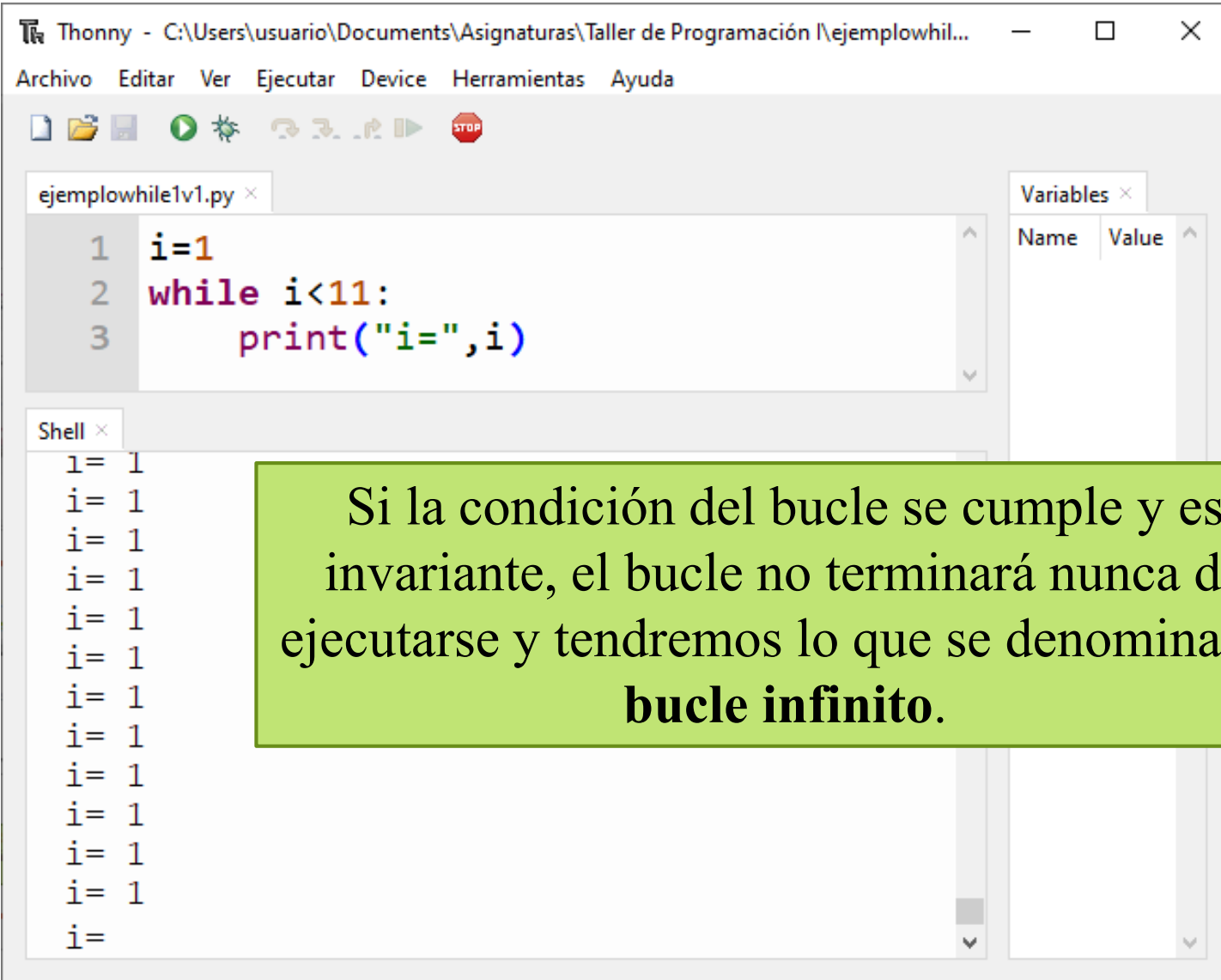
Variables

| Name | Value |
|------|-------|
| i | 11 |

Shell

```
i= 1
i= 2
i= 3
i= 4
i= 5
i= 6
i= 7
i= 8
i= 9
i= 10
```

Iteración: *While* *bucles infinitos*



Thonny - C:\Users\usuario\Documents\Asignaturas\Taller de Programación I\ejemplowhil...

Archivo Editar Ver Ejecutar Device Herramientas Ayuda

ejemplowhile1v1.py ×

```
1 i=1
2 while i<11:
3     print("i=",i)
```

Variables ×

| Name | Value |
|------|-------|
|------|-------|

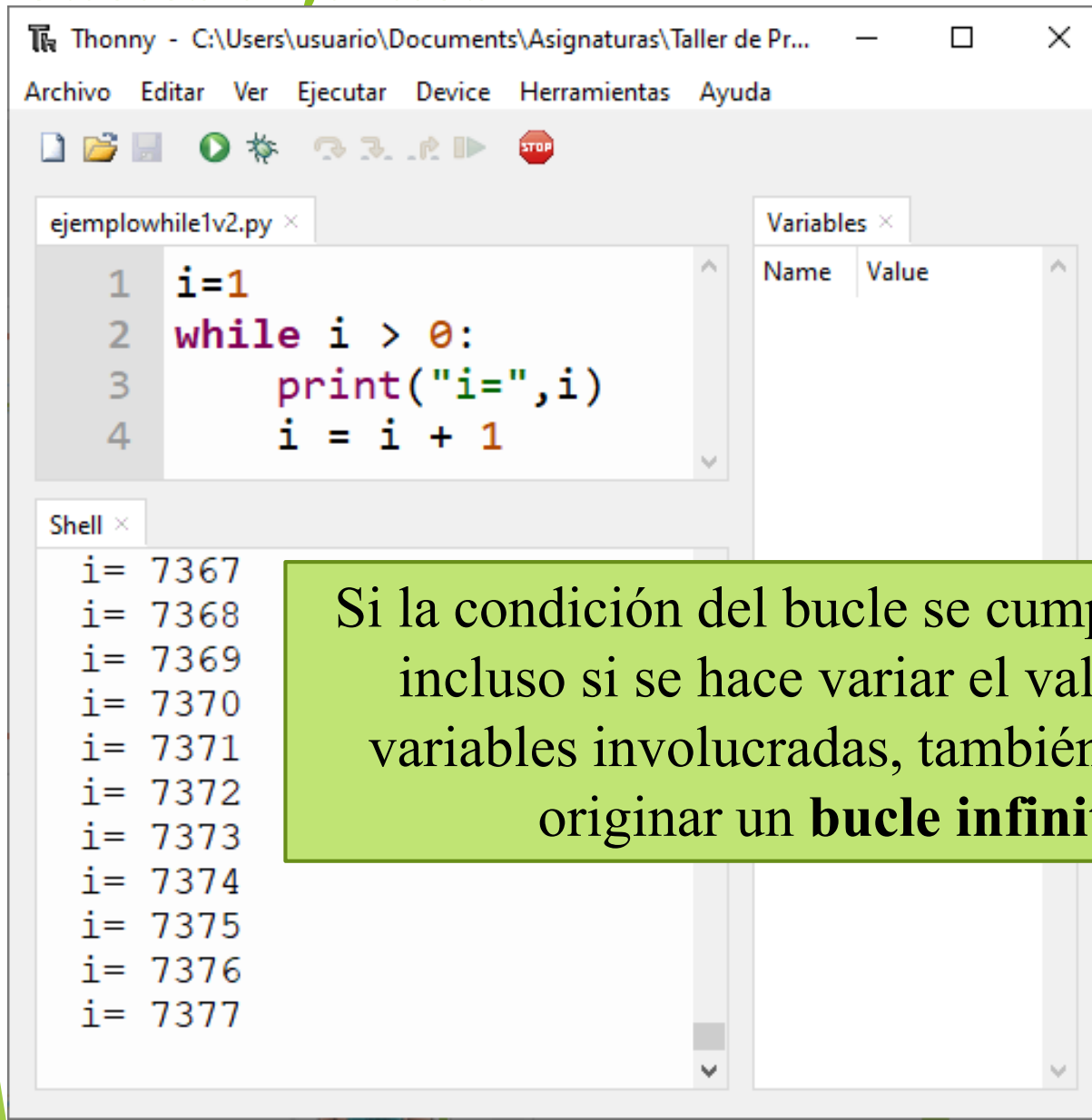
Shell ×

```
1= 1
i= 1
i= 1
i= 1
i= 1
i= 1
i= 1
i= 1
i= 1
i= 1
i= 1
i=
```

Si la condición del bucle se cumple y es invariante, el bucle no terminará nunca de ejecutarse y tendremos lo que se denomina un **bucle infinito**.

Iteración: *While*

bucles infinitos



Thonny - C:\Users\usuario\Documents\Asignaturas\Taller de Pr...

Archivo Editar Ver Ejecutar Device Herramientas Ayuda

ejemplowhile1v2.py ×

```
1 i=1
2 while i > 0:
3     print("i=",i)
4     i = i + 1
```

Variables ×

| Name | Value |
|------|-------|
|------|-------|

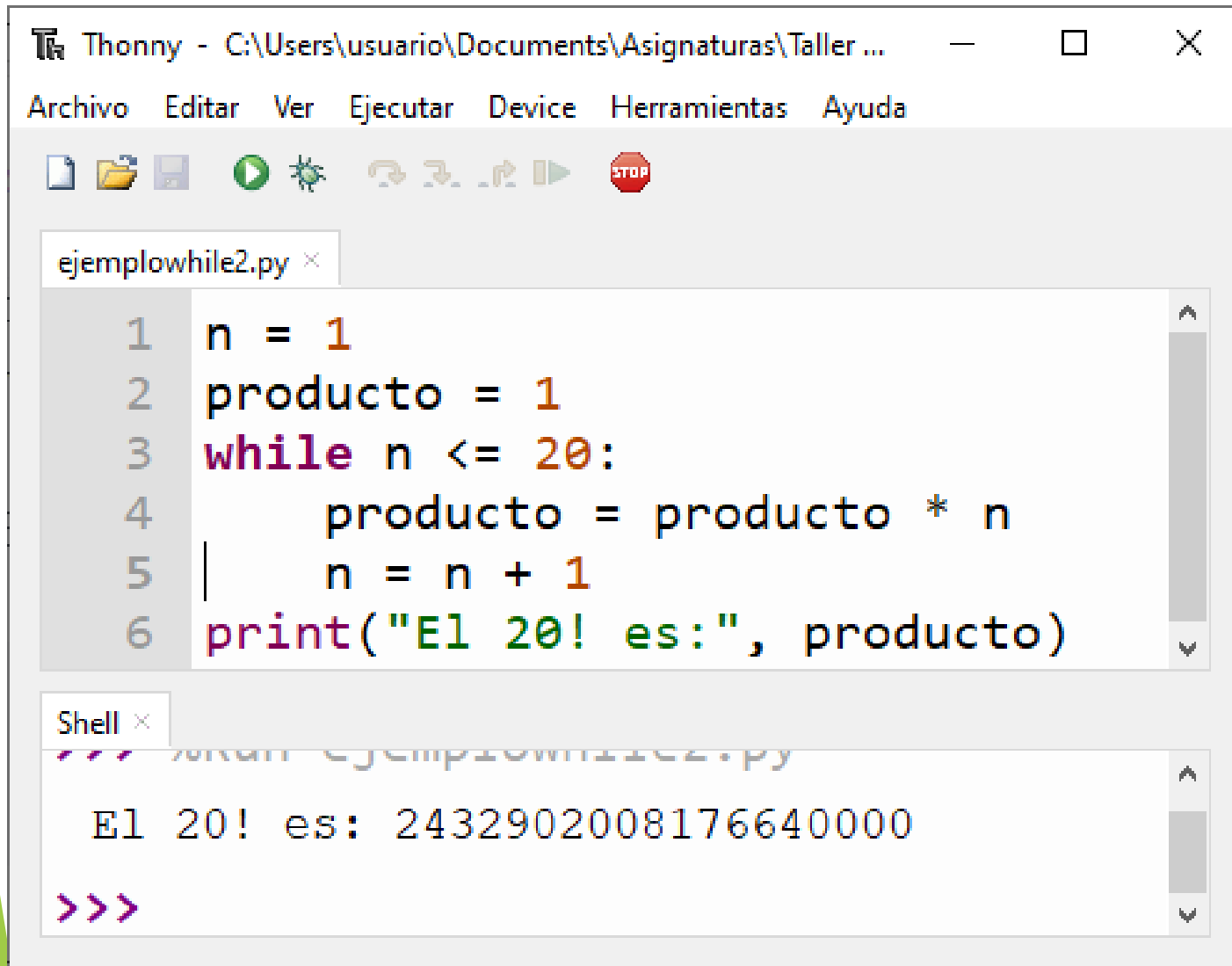
Shell ×

```
i= 7367
i= 7368
i= 7369
i= 7370
i= 7371
i= 7372
i= 7373
i= 7374
i= 7375
i= 7376
i= 7377
```

Si la condición del bucle se cumple siempre incluso si se hace variar el valor de las variables involucradas, también se puede originar un **bucle infinito**.

Ejemplo 1:

Calcular el factorial de 20 : $20! \longleftrightarrow \prod_1^{20} n$



The screenshot shows a Thonny Python IDE window. The title bar reads "Thonny - C:\Users\usuario\Documents\Asignaturas\Taller ...". The menu bar includes "Archivo", "Editar", "Ver", "Ejecutar", "Device", "Herramientas", and "Ayuda". The toolbar contains icons for file operations, running, and debugging. The editor window, titled "ejemplowhile2.py", contains the following Python code:

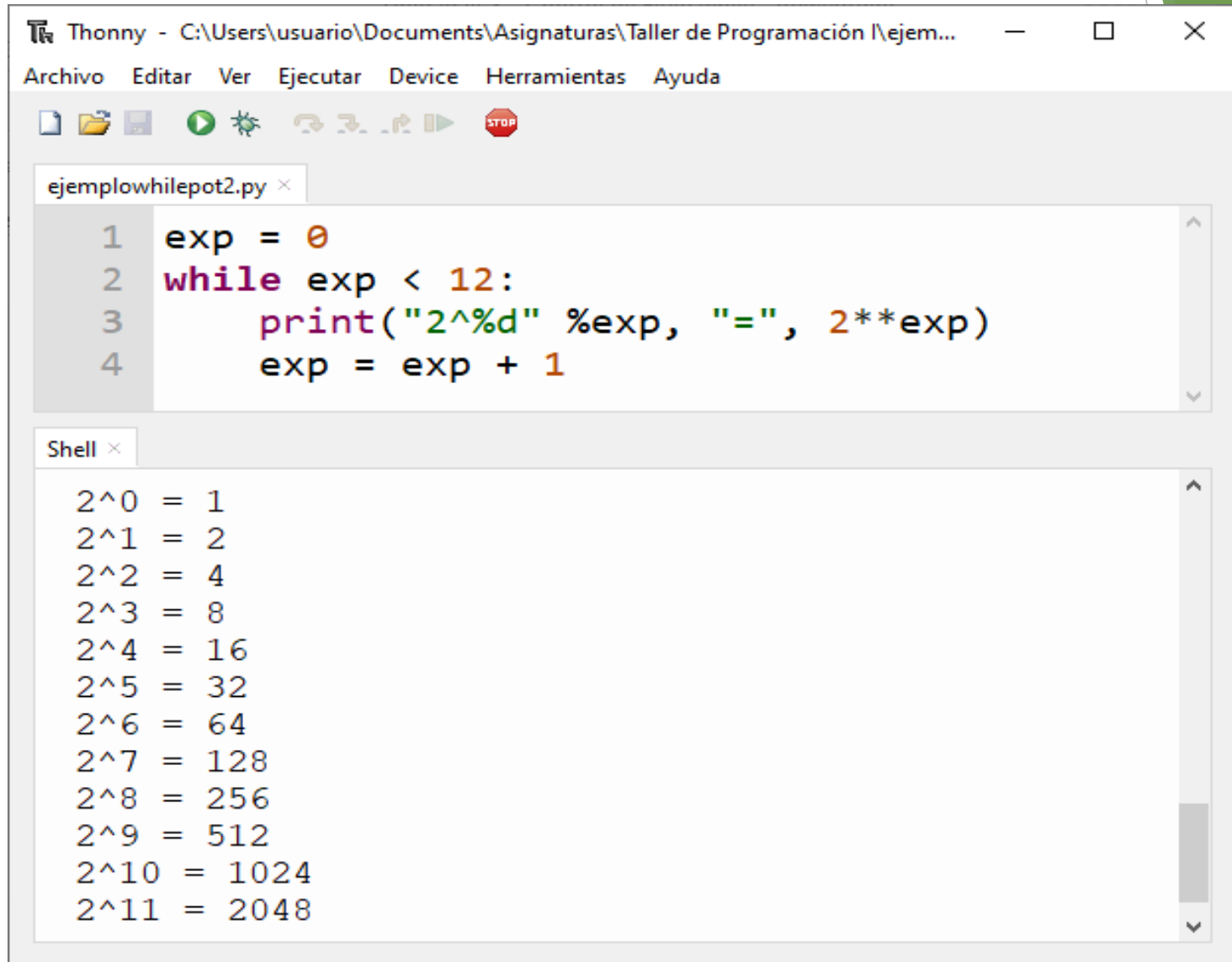
```
1 n = 1
2 producto = 1
3 while n <= 20:
4     producto = producto * n
5     n = n + 1
6 print("El 20! es:", producto)
```

Below the editor is a "Shell" window showing the output of the program:

```
Python 3.7.4 Shell
>>> python ejemplowhile2.py
El 20! es: 2432902008176640000
>>>
```

Ejemplo 2:

Desplegar las doce primeras potencias de dos: $2^0, 2^1, 2^2 \dots 2^{11}$



The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python script named 'ejemplowhilepot2.py' with the following code:

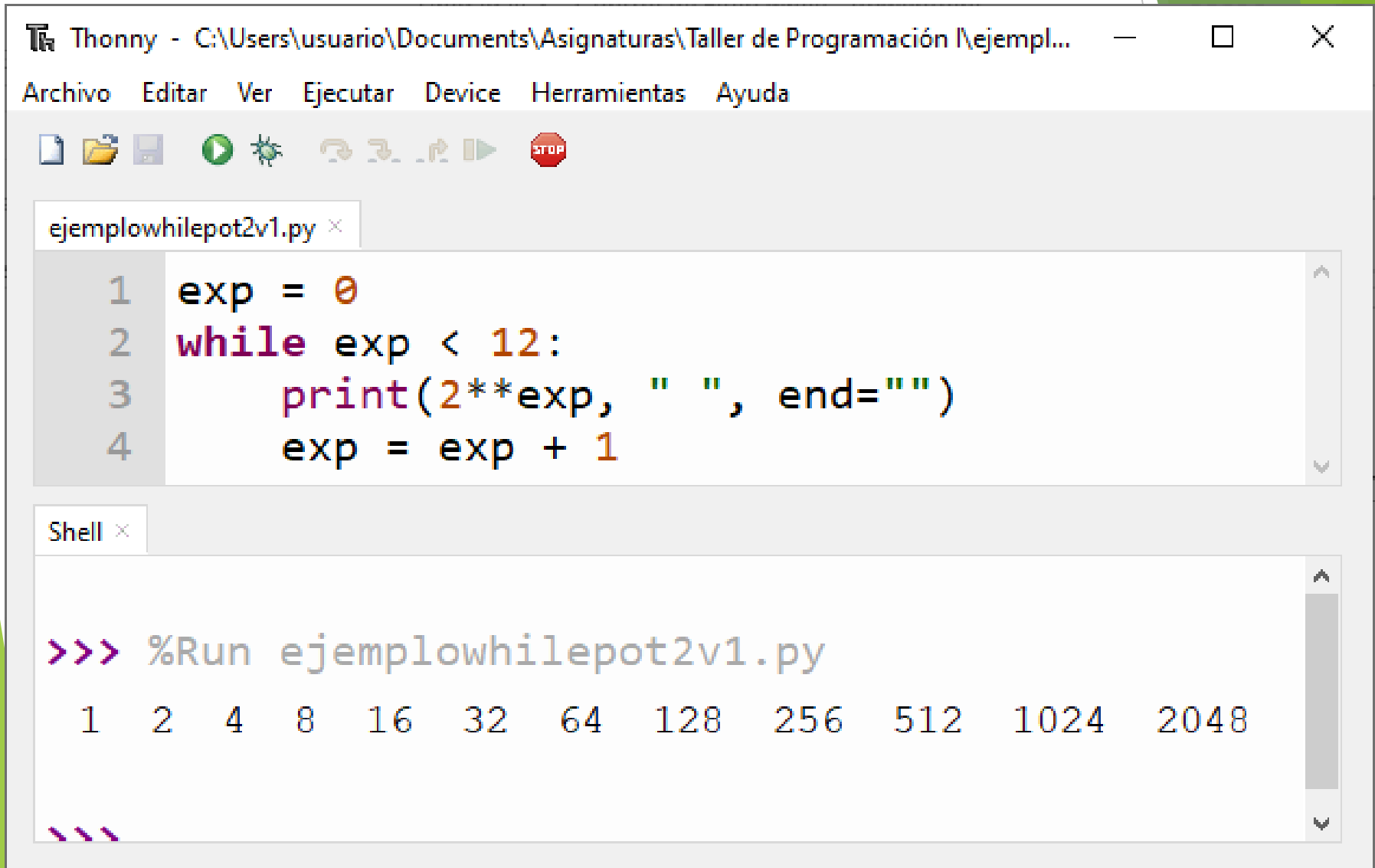
```
1 exp = 0
2 while exp < 12:
3     print("2^%d" %exp, "=", 2**exp)
4     exp = exp + 1
```

Below the editor is a 'Shell' window showing the output of the program, which prints the first 12 powers of 2:

```
2^0 = 1
2^1 = 2
2^2 = 4
2^3 = 8
2^4 = 16
2^5 = 32
2^6 = 64
2^7 = 128
2^8 = 256
2^9 = 512
2^10 = 1024
2^11 = 2048
```


Ejemplo 2, otra implementación:

Desplegar las doce primeras potencias de dos: $2^0, 2^1, 2^2 \dots 2^{11}$



The screenshot shows the Thonny Python IDE interface. The title bar reads "Thonny - C:\Users\usuario\Documents\Asignaturas\Taller de Programación I\ejempl...". The menu bar includes "Archivo", "Editar", "Ver", "Ejecutar", "Device", "Herramientas", and "Ayuda". The toolbar contains icons for file operations, running, and debugging. The editor window, titled "ejemplowhilepot2v1.py", contains the following Python code:

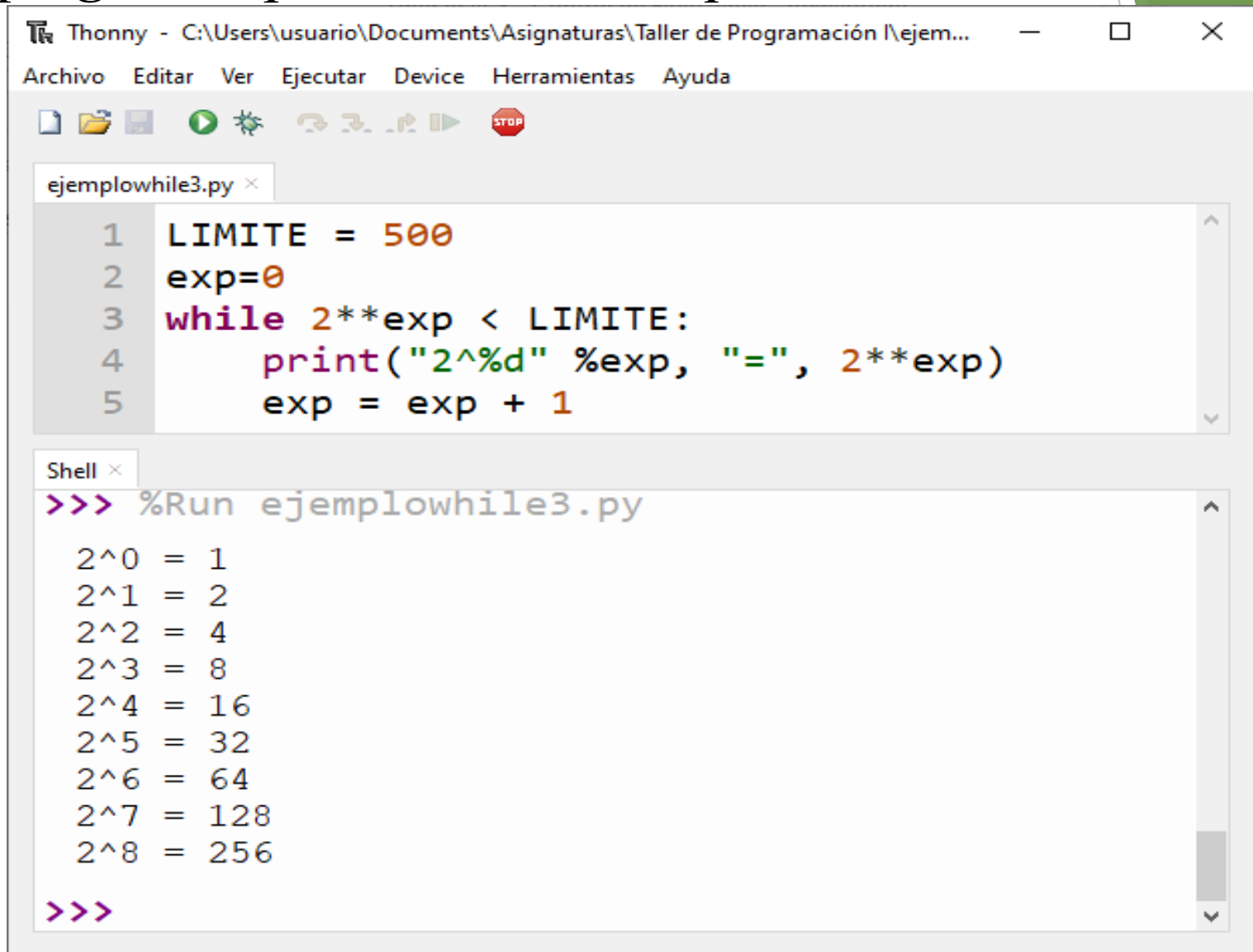
```
1 exp = 0
2 while exp < 12:
3     print(2**exp, " ", end=" ")
4     exp = exp + 1
```

Below the editor is a "Shell" window showing the execution of the script:

```
>>> %Run ejemplowhilepot2v1.py
1  2  4  8 16 32 64 128 256 512 1024 2048
>>>
```

Ejemplo 3:

Desplegar las potencias de dos que son menores a 500:



The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python script named `ejemplowhile3.py` with the following code:

```
1 LIMITE = 500
2 exp=0
3 while 2**exp < LIMITE:
4     print("2^%d" %exp, "=", 2**exp)
5     exp = exp + 1
```

Below the editor is a Shell window showing the execution of the script using the command `>>> %Run ejemplowhile3.py`. The output displays the powers of 2 that are less than 500:

```
>>> %Run ejemplowhile3.py
2^0 = 1
2^1 = 2
2^2 = 4
2^3 = 8
2^4 = 16
2^5 = 32
2^6 = 64
2^7 = 128
2^8 = 256
>>>
```

Ejemplo 4:

Desplegar las “n” primeras potencias de 2

Thonny - C:\Users\usuario\Documents\Asignaturas\Taller de Programación\ejemplowhile4.py @ 5:18

Archivo Editar Ver Ejecutar Device Herramientas Ayuda



Ejecutar el script actual (F5)

ejemplowhile4.py ×

```
1 n = int(input("Ingrese número de potencias de 2 a desplegar: "))
2 exp=0
3 while exp < n:
4     print("2^%d" %exp, "=", 2**exp)
5     exp = exp + 1
```

Shell ×

```
>>> %Run ejemplowhile4.py
```

```
Ingrese número de potencias de 2 a desplegar: 6
```

```
2^0 = 1
```

```
2^1 = 2
```

```
2^2 = 4
```

```
2^3 = 8
```

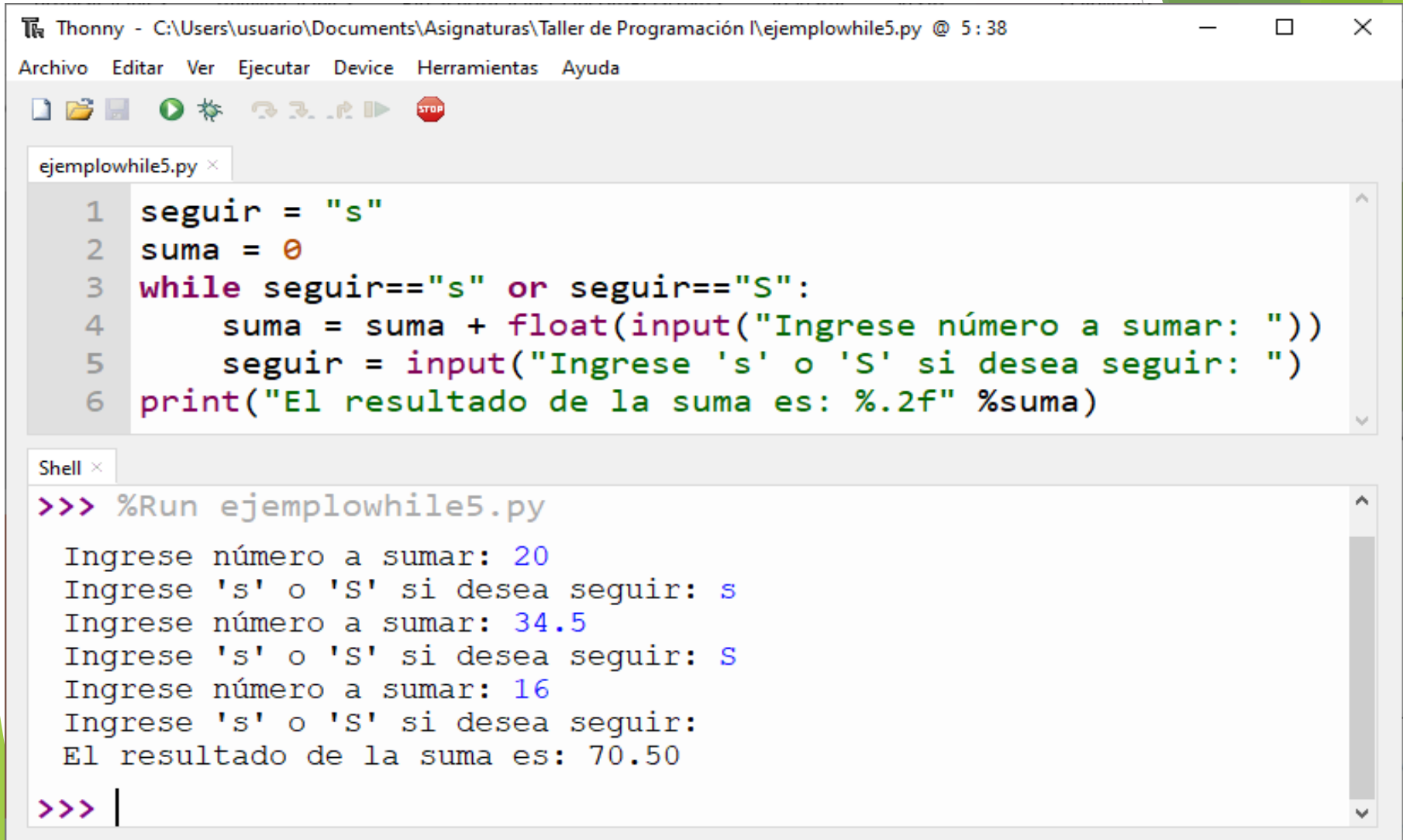
```
2^4 = 16
```

```
2^5 = 32
```

```
>>> |
```

Ejemplo 5:

Desplegar la suma de todos los números que un usuario desee ingresar, se debe preguntar si se desea seguir ingresando números (s o S).



The screenshot shows the Thonny IDE interface. The top window displays the Python script `ejemplowhile5.py` with the following code:

```
1 seguir = "s"
2 suma = 0
3 while seguir=="s" or seguir=="S":
4     suma = suma + float(input("Ingrese número a sumar: "))
5     seguir = input("Ingrese 's' o 'S' si desea seguir: ")
6 print("El resultado de la suma es: %.2f" %suma)
```

The bottom window, titled "Shell", shows the execution of the script:

```
>>> %Run ejemplowhile5.py
Ingrese número a sumar: 20
Ingrese 's' o 'S' si desea seguir: s
Ingrese número a sumar: 34.5
Ingrese 's' o 'S' si desea seguir: S
Ingrese número a sumar: 16
Ingrese 's' o 'S' si desea seguir:
El resultado de la suma es: 70.50
>>> |
```

Ejemplo 6: Construir un programa que muestre la edad promedio de los invitados a una fiesta. No se sabe cuántos son los invitados.

Thonny - C:\Users\usuario\Documents\Asignaturas\Taller de Programación I\ejemplowhile6.py @ 9:68

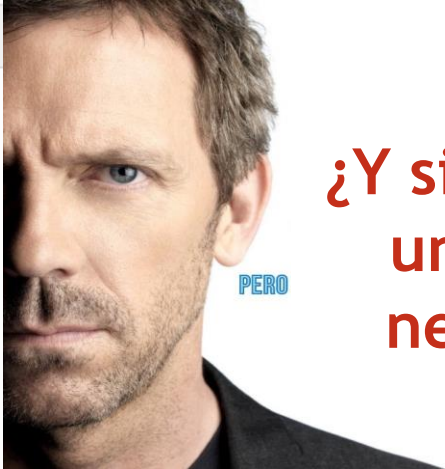
Archivo Editar Ver Ejecutar Device Herramientas Ayuda

ejemplowhile6.py ×

```
1 cont_invitados = 0
2 suma_edad = 0
3 seguir = input("Responda 'si', si vinieron invitados: ")
4 while seguir=="si" or seguir=="Si" or seguir=="sI" or seguir=="SI":
5     suma_edad = suma_edad + int(input("Ingrese la edad del invitado: "))
6     cont_invitados = cont_invitados + 1
7     seguir = input("Ingrese 'si', si desea seguir ingresando edades: ")
8 if cont_invitados > 0:
9     print("La edad promedio de los invitados es: ", round(suma_edad/cont_invitados))
10 else:
11     print("No se puede calcular la edad promedio porque no asistieron invitados")
```

Shell ×

```
>>> %Run ejemplowhile6.py
Responda 'si', si vinieron invitados: si
Ingrese la edad del invitado: 20
Ingrese 'si', si desea seguir ingresando edades: si
Ingrese la edad del invitado: 30
Ingrese 'si', si desea seguir ingresando edades:
La edad promedio de los invitados es: 25
>>> |
```



PERO

¿Y si ingresan una edad negativa?

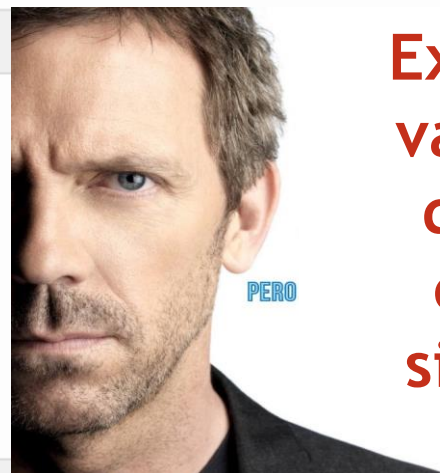


ejemplowhile6v1.py ×

```
1 cont_invitados = 0
2 suma_edad = 0
3 seguir = input("Responda 'si', si vinieron invitados: ")
4 while seguir=="si" or seguir=="Si" or seguir=="sI" or seguir=="SI":
5     edad = int(input("Ingrese la edad del invitado: "))
6     while edad < 0:
7         print("No existen las edades negativas!!")
8         edad = int(input("Ingrese la edad del invitado: "))
9     suma_edad = suma_edad + edad
10    cont_invitados = cont_invitados + 1
11    seguir = input("Ingrese 'si', si desea seguir ingresando edades: ")
12 if cont_invitados > 0:
13     print("La edad promedio de los invitados es: ", round(suma_edad/cont_invitados))
14 else:
15     print("No se puede calcular la edad promedio porque no asistieron invitados")
```

Shell ×

```
Responda 'si', si vinieron invitados: si
Ingrese la edad del invitado: -3
No existen las edades negativas!!
Ingrese la edad del invitado: 20
Ingrese 'si', si desea seguir ingresando edades: si
Ingrese la edad del invitado: 30
Ingrese 'si', si desea seguir ingresando edades:
La edad promedio de los invitados es: 25
```



PERO

**Excelente,
validar los
datos de
entrada
siempre!!**

Ejemplo 7:

Construya un programa que solicite un número y muestre la suma de todos sus dígitos. Ejemplo: **10883** \rightarrow **1 + 0 + 8 + 8 + 3 = 20**

Habrà que descomponer el número obteniendo cada uno de sus dígitos, ¿cuàndo terminar?

10883:10=1088 \rightarrow :10=108 \rightarrow :10=10 \rightarrow :10=1 \rightarrow :10=0 \rightarrow
3 8 8 0 1

Entonces, tomamos el número y aplicamos módulo 10, así obtenemos el dígito menos significativo, luego aplicamos el mismo procedimiento al resultado de la división entera por 10... hasta cuando el resultado de la división entera por 10 de 0!!

Ejemplo 7: Implementación

Thonny - C:\Users\usuario\Documents\Asignaturas\Taller de Programación\ejemplowhile7.py @ 6:43

Archivo Editar Ver Ejecutar Device Herramientas Ayuda

ejemplowhile7.py ×

```
1 numero = int(input("Ingrese número para sumar sus dígitos: "))
2 suma = 0
3 while numero != 0:
4     suma = suma + numero%10
5     numero = numero // 10
6 print("La suma de sus dígitos es: ", suma)
```

Shell ×

```
/// %kum ejemplowhile7.py

Ingrese número para sumar sus dígitos: 10883
La suma de sus dígitos es: 20

>>>
```



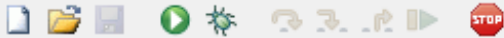
¿Y si ingresan
un número
negativo?

PERO

Ejemplo 7v1: Implementación

Thonny - C:\Users\usuario\Documents\Asignaturas\Taller de Programación I\ejemplowhile7v1.py @ 4: 67

Archivo Editar Ver Ejecutar Device Herramientas Ayuda



ejemplowhile7v1.py ×

```
1 numero = int(input("Ingrese número para sumar sus dígitos: "))
2 while numero < 0:
3     print("No se permiten números negativos!!")
4     numero = int(input("Ingrese número para sumar sus dígitos: "))
5 suma = 0
6 while numero != 0:
7     suma = suma + numero%10
8     numero = numero // 10
9 print("La suma de sus dígitos es: ", suma)
```

Shell ×

```
>>> %Run ejemplowhile7v1.py
```

```
Ingrese número para sumar sus dígitos: -202
No se permiten números negativos!!
Ingrese número para sumar sus dígitos: 10883
La suma de sus dígitos es: 20
```

```
>>> |
```



**No olvidarse
de validar los
datos de
entrada**

Ejercicios propuestos

- ▶ Solicitar al usuario que ingrese N números. Al finalizar el programa, se debe mostrar la suma de todos los números.
- ▶ Solicitar al usuario que ingrese 2 números, mostrar su suma mientras ambos sean distintos de cero.
- ▶ Mostrar los múltiplos de 3 a partir del número 15 hasta el 40
- ▶ Mostrar los múltiplos de 3 y 5 simultáneamente a partir de 15 y hasta el 135
- ▶ Generar una cantidad de múltiplos de 5, la cantidad debe ser ingresada.
- ▶ Solicitar los extremos de un intervalo, luego solicitar un número y mostrar todos los múltiplos de este número que existen en el intervalo. Utilice sólo número positivos.