In [2]:
```
#---------------- definicions
def entrar():
    print ("Hola")
def sortir():
    print ("Adeu")
#---------------- inici del programa
entrar()
sortir()
```

```
Hola
Adeu
```

In [5]:
```
#---------------- definicions
def entrar(nom, extra = "no surtis"):
    print ("Hola", nom, extra)   #parametre d'entrada
def sortir():
    print ("Adeu")
#---------------- inici del programa

nom = input("nom: ")

entrar(nom, "i no tornis")
sortir()
```

```
nom: carles
Hola carles ino tornis
Adeu
```

In [11]:
```
def imprimir_tabla (num) :
    print ("Tabla del ", num )
    """imprime toda la tabla
    num entero"""
    for i in range (11) :
        print( f" {num} * {i} = {num * i}")
#------------------------------ Inicio
print ("Imprimir tabla del número: ")
print ("s=salir, *= todas")
opcion = ""

while opcion != "s" :
    opcion = input("Introduzca opción o numero :")
    if opcion == "*" :
        for i in range (1, 11) :
            imprimir_tabla (i)
    elif opcion != "s":
        numero = int (opcion)
        imprimir_tabla (numero)
```

```
Imprimir tabla del número:
s=salir, *= todas
Introduzca opción o numero :2
Tabla del  2
 2 * 0 = 0
 2 * 1 = 2
 2 * 2 = 4
 2 * 3 = 6
 2 * 4 = 8
 2 * 5 = 10
 2 * 6 = 12
 2 * 7 = 14
 2 * 8 = 16
 2 * 9 = 18
 2 * 10 = 20
Introduzca opción o numero :s
```

In [15]:
```
def test ():
    n=10
    n **= 2
#.............................

n=7

test()
print (n)
```

```
7
```

In [19]:
```
def test (n):
    n=10
    n **= 2
    return (n)    #tambe return n
#.............................

n = 7

n = test(n)
print (n)
```

```
100
```

In [21]:
```
def fruteria ():
    global fruta
    print (fruta)

fruta = "mango"
fruteria()
```

```
mango
```

In [23]:
```
# retorna dos enteros y un float
def test():
    return 2,4,5.5

n1, n2, fi = test()
print (n1 + n2 + fi)
```

```
11.5
```

In [24]:
```
print ("a", "b", "c", sep="-")
```

```
a-b-c
```

In [25]:
```
help (print)
```

```
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.
```

In [27]:
```
print ("a", "b", "c", sep="*", end="ppp")
print ("a", "b", "c", sep="*", end="\t")
```

```
a*b*cpppa*b*c
```

In [28]:
```
help (print)
```

```
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file:  a file-like object (stream); defaults to the current sys.stdout.
    sep:   string inserted between values, default a space.
    end:   string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.
```

In [31]:
```
def resta (a,b):
    return a - b
print (resta(b=30, a=10))


help (resta)
```

```
-20
Help on function resta in module __main__:

resta(a, b)
```

In [33]:
```
def input_int (mensaje, valor_defecte = 0) :
    valor = input (mensaje)
    if valor.isdigit() :
        valor = int(valor)
    else :
        valor = valor_defecte
    return (valor)
#------------------------------ inici
num = input_int("Entri un número : ", -1)
print ("num = ", num)
```

```
Entri un número : f
num =  -1
```

In [35]:
```
def total_fra_amb_iva (total_sense_iva, perc_iva = 21) :
    return (total_sense_iva + (total_sense_iva * perc_iva /100))
#------------------- inici del programa
t = total_fra_amb_iva (100)
print (t)
```

```
121.0
```

In [37]:
```
#------------------------------ definiciones
def ksuma (numero, k) :
    return (numero + k)
#------------------------------ inici del programa
print (ksuma (7, 3.14))
# Exemple 1 : Tenim una funció que suma un número i una constant
# k, i retorna el resultat de la suma.
# La modificarem perquè, si no informen la constant, assumeixi
# que és zero.
#------------------------------ definiciones
def ksuma (numero, k = 0) :
    return (numero + k)
#------------------------------ inici del programa
print (ksuma (7))
```

```
10.14
7
```

In [40]:
```
def media(grup):
    """Función que calcula la media de una muestra de números.
    Parámetros
    sample: Es una lista de números
    Devuelve la media de los números en sample.
    """
    return sum(grup)/len(grup)

print(media([1, 2, 3, 4, 7]))
print(media([2.3, 5.7, 6.8, 9.7, 12.1, 15.6]))
```

```
3.4
8.700000000000001
```

In [48]:
```
def quadrats(muestra):
    muestra2 = []
    i = 0
    while i < len(muestra):
        muestra2.append(muestra[i]**2)
        i= i+1
    return(muestra2)

print (quadrats ([2, 3, 5]))
```

```
[4, 9, 25]
```

In [43]:
```
def square(sample):
    """Función que calcula los cuadrados de una lista de números.
    Parámetros
    sample: Es una lista de números
    Devuelve una lista con los cuadrados de los números de la lista sample.
    """
    list = []
    for i in sample:
        list.append(i**2)
    return list

print(square([1, 2, 3, 4, 5]))
print(square([2.3, 5.7, 6.8, 9.7, 12.1, 15.6]))
```

```
[1, 4, 9, 16, 25]
[5.289999999999999, 32.49, 46.239999999999995, 94.08999999999999, 146.41, 243.35999
999999999]
```

In [49]:
```
def squares(*sample):
    """Función que calcula los cuadrados de una lista de números.
    Parámetros
    *sample: Es una secuencia de números separados por comas.
    Devuelve una lista con los cuadrados de los números de sample.
    """
    list = []
    for i in sample:
        list.append(i**2)
    return list

print(squares(1, 2, 3, 4, 5))
print(squares(2.3, 5.7, 6.8, 9.7, 12.1, 15.6))
```

```
[1, 4, 9, 16, 25]
[5.289999999999999, 32.49, 46.239999999999995, 94.08999999999999, 146.41, 243.35999
999999999]
```

In [51]:
```
def squaref (muestra):
    muestra2=[]
    for i in muestra:
        muestra2.append(i**2)
    return(muestra2)

print(squaref([1, 2, 3, 4, 5]))
```

```
[1, 4, 9, 16, 25]
```

In [52]:
```
def variar (a, b, c ):
    a = 3
    b = "kk"
    c [0] = 7
#------------------- inici
n = 2
cad ="Hola"
llista = [1,2,3]
print (f"abans n:{n} cad:{cad} llista:{llista}")
variar(n, cad, llista)
print (f"despr n:{n} cad:{cad} llista:{llista}")
```

```
abans n:2 cad:Hola llista:[1, 2, 3]
despr n:2 cad:Hola llista:[7, 2, 3]
```

In [57]:
```
#------------ definició de funcions
def pide_opcion():
    print ("1. Entra coche, 2.Sale coche, 3.Listar coches, 0.Salir")
    opcion = input("opción: ")
    while opcion not in ("0123"):
        print("opción invàlida")
        opcion = input("opción: ")
    return (opcion)
def pide_matricula():
    matricula = input("matrícula: ")
    return(matricula)
def entra_coche():
    matricula = pide_matricula()
    if matricula not in cars :
        cars.append(matricula)
    else:
        print ("el coche ya está dentro")
def sale_coche():
    matricula = pide_matricula()
    if matricula in cars :
        cars.remove(matricula)
    else:
        print ("el coche NO está dentro")
def listar_coches():
    print (cars)
#------------ inicio del programa
cars = []
opcion = pide_opcion()
while opcion != "0" :

    if opcion == "1" :
        entra_coche()
    elif opcion == "2":
        sale_coche()
    elif opcion == "3":
        listar_coches()
    elif opcion != "0":
        print ("opción invàlida")
    opcion = pide_opcion()
# La lista de coches esta
# definida de forma global.
# Todas las funciones la
# pueden usar sin
# definirla.
```

```
1. Entra coche, 2.Sale coche, 3.Listar coches, 0.Salir
opción: 2
matrícula: 23155ASD
el coche NO está dentro
1. Entra coche, 2.Sale coche, 3.Listar coches, 0.Salir
opción: 1
matrícula: 1234ASD
1. Entra coche, 2.Sale coche, 3.Listar coches, 0.Salir
opción: 3
['1234ASD']
1. Entra coche, 2.Sale coche, 3.Listar coches, 0.Salir
opción: 0
```

In [62]:
```
def area (b, a):
    area = b * a
    return area
a =float (input("diguem la alçada en cm: "))
b =float (input("diguem la baseen cm: "))
print (area(b,a))
```

```
diguem la alçada en cm: 2.4
diguem la baseen cm: 1.7
4.08
```

In [73]:
```
def relacion (a,b):
    if a>b :
        return 1
    elif a<b :
        return -1
    else:
        return 0

a = int(input("introdueix un número sencer: "))
b = int(input("introdueix un altre número sencer: "))
print (relacion(a,b))
```

```
introdueix un número sencer: 4
introdueix un altre número sencer: 5
-1
```

In [76]:
```
def relacion (a,b):
    r=0
    if a>b :
        r = 1
    elif a<b :
        r = -1
    else:
        r = 0
    return r

a = int(input("introdueix un número sencer: "))
b = int(input("introdueix un altre número sencer: "))
print (relacion(a,b))
```

```
introdueix un número sencer: 4
introdueix un altre número sencer: 5
-1
```

In [77]:
```
def intermedio (a,b):
    intermedio = (a + b)/2
    return intermedio

a = int(input("introdueix un número sencer: "))
b = int(input("introdueix un altre número sencer: "))
print (intermedio(a,b))
```

```
introdueix un número sencer: 4
introdueix un altre número sencer: 5
4.5
```

In [84]:
```
# Ejercicio 5 (Avanzado). Cálculo de tiempos
# Escribir dos funciones que permitan calcular:
# a) La cantidad de segundos en un tiempo dado en horas, minutos
# y segundos. calcula_segundos (“hh:mm:ss”) devuelve segundos
# b) La cantidad de horas, minutos y segundos de un tiempo dado
# en segundos calcula_hhmmss (segundos) devuelve “hh:mm:ss”
# Después realiza un programa con un menú de opciones
# 1. Calcular cantidad de segundos
# 2. Calcular horas, minutos y segundos
# y pide los datos adecuados a la opción elegida y muestra el
# resultado.

def calcula_segundos (hh,mm,ss):
    h = hh*60
    m = mm*60
    s = ss
    calcula_segundos = h + m +s
    return (calcula_segundos)

#def calcula_hhmmss (ss):



print (calcula_segundos(1, 1, 5))
```

```
125
```

In [ ]: