# Hoja Ejercicios 1

Lista de ejercicios propuestos para el curso de Python del Club de Software EPN

## Temas a evaluar:

• Sintaxis básica

• Tipos primitivos y estructuras de datos básicas

• Condicionales

• Iteradores

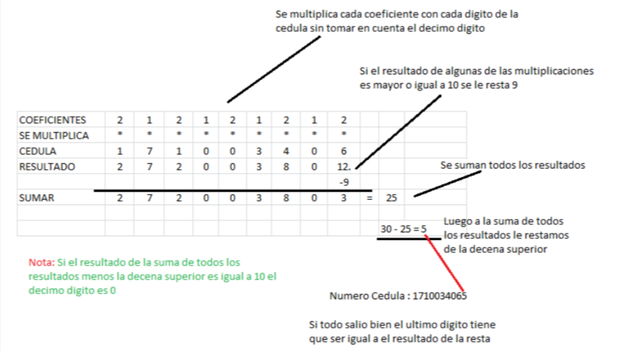
• Programación Funcional

• Uso de nuevas librerías

**Nombre:** Andrés Sebastián Suárez Suárez

### Problema 1:

Solicite que un usuario ingrese su número de cédula, valide lo ingresado tanto en extensión de dígitos, como si contiene caracteres que no sean números, considere todos los posibles casos y emita la advertencia o mensaje de error correspondiente, de ser una cadena de texto válida proceda a aplicar el algoritmo de validación de la cédula de identidad ecuatoriana. Finalmente emita un mensaje indicando si la cédula ingresada es válida o no. Considere todas las posibilidades. Se adjunta una imagen del algoritmo de validación en caso de no conocerlo




In [ ]:
```
#Solución
cedula = input('Ingrese su ceula')
valida = False

# comprobamos longitud de la cedula
if len(cedula) == 10:
    # comprobamos que solo tenga numeros la cedula
    try:
        cedula = int(cedula)
        valida = True

    except:
        print('La cedula contiene algun caracter no deceable')

    if valida == True:
        suma = 0
        cedula = list(str(cedula))
        coeficiente = [2,1,2,1,2,1,2,1,2]
        cedula = list(cedula)

        # Multiplicamos cada numero de la cedula con su coeficiente respectivo
        for i in range(len(coeficiente)):
            resultado = int(coeficiente[i]) * int(cedula[i])

            # Si la multiplicacion es mayor igual a 10 le restamos 9
            if resultado >= 10:
                resultado = resultado - 9

            suma = suma + resultado

        # Scamos la decena superior de nuestra suma
        mod = suma % 10
        redondeo = 10 - mod
        nRedeondeado = suma + redondeo

        # Restamos la decena superior menos nuestra suma
        comprobar = nRedeondeado - suma
        com = int(cedula[len(cedula) - 1])

        # Comprobamos si el ultimo digito de la cedula es igual a nuestro comprobar
        if comprobar == com:
            print('Cedula valida')
        else:
            print('Cedula no valida')

else:
    print('Longitud de la cedula incorrecta')
```

```
Cedula valida
```

### Problema 2

Elabore un diccionario que contenga como llave los números del 1 a 10, y como valor será su llave elevado al cuadrado.

Ejemplo:

{1:1, 2:4, 3:9, ...}

In [ ]:
```
### aqui su solución
n = []
for i in range(1,11):
    n.append(i)

n2 = list(map(lambda x: x**2, n))

print(list(zip(n,n2)))
```

```
[(1, 1), (2, 4), (3, 9), (4, 16), (5, 25), (6, 36), (7, 49), (8, 64), (9, 81), (10, 100)]
```

### Problema 3

Usando programación funcional, elabore una expresión que sea capaz de generar n términos de la serie de Fibonacci

In [ ]:
```
# aqui su solución del problema propuesto

from functools import reduce

numeroN = lambda x, y: x+[x[-1]+x[-2]]
lista = lambda n : reduce(numeroN, [0]*(n - 3), [0,1,1])

print(lista(10))
```

```
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

### Problema 4

Elabore una lista de 100 números aleatorios. Haciendo uso de la compresión de listas, cree una nueva lista que captura únicamente aquellos números que son números primos.

In [ ]:
```
# aqui su solución del problema propuesto

import random

lista = []

for i in range(1,100):
    n = random.randint(1, 100)
    lista.append(n)

print('Lista completa: ')
print(lista)
x = []

print('Lista con numeros primos')
for i in range(len(lista)):
    n = [lista[i] for j in range(1, lista[i]) if lista[i]% j == 0]
    x = x + (list(map(lambda x: n if len(n) == 1 else '', n)))

print(list(filter(lambda y: False if y == '' else True, x)))
```

```
Lista completa:
[63, 59, 61, 75, 67, 30, 45, 80, 24, 23, 96, 9, 17, 72, 44, 91, 56, 76, 44, 9, 95, 5, 93, 9, 26, 21, 33, 73, 93, 85, 4, 56, 70, 38, 67, 68, 57, 73, 1, 41, 94, 79, 6, 95, 76, 29, 43, 77, 83, 34, 24, 22, 60, 20, 73, 31, 5
8, 4, 51, 6, 80, 70, 66, 82, 61, 89, 88, 3, 34, 34, 79, 87, 62, 59, 44, 6, 91, 24, 83, 73, 29, 95, 35, 46, 41, 40, 28, 17, 42, 44, 61, 31, 81, 33, 13, 78, 4, 34, 6]
Lista con numeros primos
[[59], [61], [67], [23], [17], [5], [73], [67], [73], [41], [79], [29], [43], [83], [73], [31], [61], [89], [3], [79], [59], [83], [73], [29], [41], [17], [61], [31], [13]]
```

### Problema 5

Usted ha sido asigando como programador en un laboratorio de investigación, su trabajo consiste en procesar computacionalmente señales usando python. Le solicitan que calcule la transformada rápida de fourier (fft) de las señales que le envían.

Usted sabe que la mejor librería para este tipo de problemas es Scipy.

https://scipy.org

**Actividades:**

1. Instale la librería scipy siguiendo la documentación.
2. Usted recive la señal en un formato array de numpy. (Puede investigar más sobre el tema). La señal ya está lista para usarse en la variable llamada signal. Únicamente pase esa variabe como parámetro.
3. Usando la documentación de Scipy (https://scipy.github.io/devdocs/index.html) busque una solución que le permita calcular la transformada rápida de fourier. Puede buscar términos como fft.
4. Cuando obtenga los resultados de búsqueda, use la solución más simple (el método fft, los primeros resultados).
5. Lea la documentación del método fft, en la documentación encontrará ejemplos para que usted aprenda por su cuenta a usar este método.
6. Finalmente calcule la transformada rápida de fourier de la señal propuesta en el código. Guarde todos los cambios de este notebook y envíelo.

In [ ]:
```
# importar el módulo fft de scipy que corresponda
from scipy.fft import fft
import numpy as np
signal = np.exp(2j * np.pi * np.arange(4) / 4)
# calcular la trnsformada rápida de fourier con el paquete necesario
y = fft(signal)

print(y)
```

```
[-1.2246468e-16+1.2246468e-16j  4.0000000e+00-3.6739404e-16j
  1.2246468e-16+1.2246468e-16j  0.0000000e+00+1.2246468e-16j]
```