Nombre: Salvador Gimeno

Actividad Guiada 1

Github: https://github.com/salvagimeno-ai/03MAIR-Algoritmos-de-optimizacion/tree/master/

Google Colab: https://colab.research.google.com/drive/1njMWKs4bbKJvoL_KlMLdGa5apfYMz

Decorador para medir tiempos

```
from functools import wraps
from time import time

def calcular_tiempo(f):
    @wraps(f)
    def cronometro(*args, **kwargs):
        t_inicial = time()
        salida = f(*args, **kwargs)
        t_final = time()
        print('Tiempo transcurrido (en segundos): {}'.format(t_final - t_inicial))
        return salida
    return cronometro
```

Ordenacion con Algoritmo de Quick Sort con técnica Divide y Vencerás

```
import random
def quick_sort(A):
    if len(A) == 1:
        return A

    elif len(A) == 2:
        return [min(A),max(A)]

    elif len(A) > 2:
        #en este caso el pivote se calculará como la media de los 3 primeros valores de la
        pivote = (A[0] + A[1] + A[2])/3

        IZQ = []
        DER = []

        for i in A:
            if i < pivote:
                IZQ.append(i)
            else:
                DER.append(i)
        #print('paso recursivo')
        return quick_sort(IZQ)  + quick_sort(DER)
```

```
# LISTAS PARA PRUEBAS:
A = [9187, 244, 4054, 9222, 8373, 4993, 5265, 5470, 4519, 7182, 2035, 3506, 4337, 7580, 25
B = [9187, 244, 1, 24, 154, 2321, 123, 12]
C = [9187]


@calcular_tiempo
def ordenar(A):
    print(quick_sort(A))

ordenar(A)
```

```
[244, 2035, 2554, 2824, 3506, 4054, 4337, 4447, 4519, 4993, 5265, 5470, 7182, 7379, 7
Tiempo transcurrido (en segundos): 0.0010006427764892578
```

```
D=list(map(lambda x: random.randrange(1, 10000), range(1,300)))
ordenar(D)
```

```
[56, 79, 110, 114, 126, 133, 147, 241, 246, 258, 293, 295, 311, 345, 452, 484, 499, 5
Tiempo transcurrido (en segundos): 0.0010120868682861328
```

Algoritmo Voraz con Python - Problema de Cambio de monedas

```
SISTEMA=[25,10,5,1]

@calcular_tiempo
def cambio_monedas(C,SISTEMA):
    SOLUCION = [0 for i in range(len(SISTEMA))]

    VALOR_ACUMULADO = 0

    for i in range(len(SISTEMA)):
        monedas = int((C - VALOR_ACUMULADO)/SISTEMA[i])

        SOLUCION[i] = monedas
        VALOR_ACUMULADO += monedas*SISTEMA[i]

        if C == VALOR_ACUMULADO:
            return SOLUCION



cambio_monedas(77,SISTEMA)
```

```
Tiempo transcurrido (en segundos): 0.0
[3, 0, 0, 2]
```

Problema de las 4 Reinas

```
N=4
```

```
Solucion0=[0 for i in range(N)]

Etapa=0

def es_prometedora(Solucion,Etapa):
    #print(Solucion)
    for i in range (Etapa+1):
        if Solucion.count(Solucion[i])>1:
            return False
        for j in range(i+1,Etapa+1):
            if abs(i-j)== abs(Solucion[i]-Solucion[j]): return False
    return True

def Dibuja(S):
    n = len(S)
    for x in range(n):
        print("")
        for i in range(n):
            if S[i] == x+1:
                print(" X " , end="")
            else:
                print(" - ", end="")

def Reinas (N,Solucion=Solucion0,Etapa=0):
    for i in range(1,N+1):
        Solucion[Etapa]=i
        EsPrometedora=es_prometedora(Solucion,Etapa)

        if EsPrometedora and Etapa==N-1:
            print ("\n\nla solución es:")
            print (Solucion)
            Dibuja(Solucion)
        elif EsPrometedora:
            Reinas(N,Solucion,Etapa+1)
        else:
            None
        Solucion[Etapa]=0

@calcular_tiempo
def TR(N):
    return Reinas(N)

TR(N)
```

```
la solución es:
[2, 4, 1, 3]

 -  -  X  -
 X  -  -  -
 -  -  -  X
 -  X  -  -

la solución es:
[3, 1, 4, 2]

 -  X  -  -
 -  -  -  X
 X  -  -  -
 -  -  X  - Tiempo transcurrido (en segundos): 0.0019986629486083984
```