



PROBLEMA

1. Dado un vector de n elementos, realiza una función para mostrar un segmento del mismo, delimitado por 2 parámetros a,b que se le pedirá al usuario.

Vector inicial							Resultado
$V = \begin{bmatrix} 2 & 1 & 5 & 7 & 1 & 3 & 5 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{bmatrix}$							Si $a=2, b=5$ 5,7,1,3

SOURCE IN PYTHON

```
# _____ function para generar vector _____
def leer():
    tam = int(input("Tamaño Vector="))
    return tam

# _____ function para generar vector _____
def llenarVector(tamano):
    V = [0]*n
    for i in range(n):
        x=int(input(f"[{i}]="))
        V[i]=x
    return V

def mostrar(V, n):
    print("[",end="")
    for i in range(n):
        print(V[i],end="")
    print("]")

def leerSegmento():
    print("_____ Leer Segmento_____")
    a = int(input("índice inicial = "))
    b = int(input("índice final = "))
    return a,b

def mostrarSegmento(V):
    segmento=[ ] # nuevo vector llenamos el segmento
    a,b = leerSegmento()
    if a < 0 or b>=len(V) or a>b:
        return "Rango invalido"
    for i in range(len(V)): # puedo usar for i in rage(a,b+1):
        if i >=a and i<=b:
            segmento.append(V[i])
            #print(segmento)
    return segmento

#_____ function main _____
n = leer()
print("Ingrese valores para llenar vector:")
vector = llenarVector(n)
print("_____ Mostrar Original_____")
mostrar(vector,n)
res=mostrarSegmento(vector)
print("_____ MostrarSegmento_____")
print(res)
```

DESK CHECK

```
Tamano Vector=7
Ingrese valores para llenar vector:
[0]=2
[1]=1
[2]=5
[3]=7
[4]=1
[5]=3
[6]=5
Mostrar Original_____
[2, 1, 5, 7, 1, 3, 5,]
Leer Segmento_____
índice inicial = 2
índice final = 5
MostrarSegmento_____
[5, 7, 1, 3]
```



PROBLEMA

2. Rotar 3 veces a la derecha, un vector de tamaño N.

Ejemplo.

Vector inicial	Resultado
5 3 8 9 1 2	9 1 2 5 3 8

SOURCE IN PYTHON

```
# _____ function para leer vector _____  
def leer():  
    tam = int(input("Tamaño Vector="))  
    return tam  
  
# _____ function para generar vector _____  
def llenarVector(tamano):  
    V = [0]*n  
    for i in range(n):  
        x=int(input(f"[{i}]="))  
        V[i]=x  
    return V  
# _____ function mostrar vector _____  
def mostrar(V, n):  
    print("[",end="")  
    for i in range(n):  
        print(V[i],end="")  
    print("]")  
  
def leerDesplazar():  
    print("_____ cuantos numeros desplazara_____")  
    d = int(input("# de Desplazamiento ="))  
    return d  
  
def mostrarDesplazamiento(V,n):  
  
    d= leerDesplazar()  
    # o se pude escribir aux=V[n-1]  
  
    for j in range(d):  
        ultimo=V[-1]  
        for i in range(n-1,0,-1):  
            V[i]=V[i-1]  
        V[0]=ultimo  
    return V
```

DESK CHECK

```
# _____ function main _____  
n = leer()  
print("Ingrese valores para llenar vector: ")  
vector = llenarVector(n)  
print("_____ Mostrar Original _____")  
mostrar(vector,n)  
res=mostrarDesplazamiento(vector,n)  
print("_____ MostrarDesplazamiento _____")  
print(res)
```

```
Tamanho Vector=6  
Ingrese valores para llenar vector:  
[0]=5  
[1]=3  
[2]=8  
[3]=9  
[4]=1  
[5]=2  
_____ Mostrar Original _____  
[5,3,8,9,1,2,]  
_____ cuantos numeros kiere despla:  
# de Desplazamiento = 3  
_____ MostrarDesplazamiento _____  
[9, 1, 2, 5, 3, 8]
```





PROBLEMA

3. Dado el vector V de tamaño N (par), dividir imaginariamente el vector en dos partes iguales. Rotar los elementos de la primera parte hacia la izquierda y los elementos de la segunda parte hacia la derecha. Ejemplo.

Vector inicial	Resultado																														
<table border="1"><tr><td>12</td><td>5</td><td>13</td><td>6</td><td>8</td><td>1</td><td>3</td><td>22</td><td>80</td><td>65</td></tr><tr><td colspan="5" style="text-align: center;">← →</td><td colspan="5"></td></tr></table>	12	5	13	6	8	1	3	22	80	65	← →										<table border="1"><tr><td>5</td><td>13</td><td>6</td><td>8</td><td>12</td><td>65</td><td>1</td><td>3</td><td>22</td><td>80</td></tr></table>	5	13	6	8	12	65	1	3	22	80
12	5	13	6	8	1	3	22	80	65																						
← →																															
5	13	6	8	12	65	1	3	22	80																						

SOURCE IN PYTHON

```
# _____ function para leer tamaño del vector _____  
def leer():
```

```
    tam = int(input("Tamaño del vector = "))  
    return tam
```

```
# _____ function para generar vector _____  
def llenarVector(tamano):
```

```
    V = [0] * tamano  
    for i in range(tamano):  
        x = int(input(f"Elemento {i+1} = "))  
        V[i] = x  
    return V
```

```
# _____ function para mostrar vector _____  
def mostrar(V, n):
```

```
    print("[", end="")  
    for i in range(n):  
        print(V[i], end="," if i < n - 1 else "")  
    print("]")
```

```
def rotarlzquierdaMitad(V, n):
```

```
    mitad = n // 2  
    primero = V[0]  
  
    for i in range(mitad - 1):  
        V[i] = V[i + 1]  
    V[mitad - 1] = primero
```

```
def rotarDerechaMitad(V, n):
```

```
    mitad = n // 2  
    ultimo = V[n - 1]  
  
    for i in range(n - 1, mitad, -1):  
        V[i] = V[i - 1]  
    V[mitad] = ultimo
```

```
# _____ function para mostrar el desplazamiento _____  
def mostrarDesplazamiento(V, n):
```

```
    rotarlzquierdaMitad(V, n)  
  
    rotarDerechaMitad(V, n)  
  
    return V
```

```
# _____ function main _____
```

```
n = leer()  
print("Ingrese valores para llenar el vector:")  
vector = llenarVector(n)
```

```
print("_____ Mostrar Original _____")  
mostrar(vector, n)
```

```
# Aplicar el desplazamiento en ambas mitades  
res = mostrarDesplazamiento(vector, n)  
print("_____ Mostrar Desplazamiento _____")  
mostrar(res, n)
```

DESK CHECK

Tamaño del vector = 10

Ingrese valores para llenar el vector:

Elemento 1 = 12

Elemento 2 = 5

Elemento 3 = 13

Elemento 4 = 6

Elemento 5 = 8

Elemento 6 = 1

Elemento 7 = 3

Elemento 8 = 22

Elemento 9 = 80

Elemento 10 = 65

Mostrar Original _____

[12, 5, 13, 6, 8, 1, 3, 22, 80, 65]

Mostrar Desplazamiento _____

[5, 13, 6, 8, 12, 65, 1, 3, 22, 80]



PROBLEMA

4. Dado un vector de n elementos, realiza una función para rotar a la derecha un segmento del mismo, delimitado por 2 parámetros a,b que se le pedirá al usuario.

SOURCE IN PYTHON

```
# _____ function para leer vector _____  
def leer():  
    tam = int(input("Tamanho Vector="))  
    return tam  
def leerDelimitador():  
    print("_____ Delimitando del segmento a rotar _____")  
    a = int(input("Delimitador inicial a="))  
    b = int(input("Delimitador final b="))  
    return a,b  
  
# _____ function para generar vector _____  
def llenarVector(tamano):  
    V = [None]*n  
    for i in range(n):  
        x=int(input(f"[{i}]="))  
        V[i]=x  
    return V  
# _____ function mostrar vector _____  
def mostrar(V, n):  
    print("[",end="")  
    for i in range(n):  
        print(V[i],end="")  
    print("]")  
# _____ delimitador derecha _____  
def segmentoDilimitadorIzquierda(V,n):  
    a,b=leerDelimitador()  
    if a < 0 or b >= len(V) or a>=b: #len(V) cuenta tamno de fila pongale n si gusta  
        print("Fuera de Limite")  
        return V  
    inicio=V[a]  
    for i in range(a,b):  
        V[i]=V[i+1]  
    V[b]=inicio  
    return V  
# _____ Deloimitador derecha _____  
def segmentoDilimitadorDerecha(V,n):  
    a,b=leerDelimitador()  
    if a < 0 or b >= len(V) or a>=b: #len(V) cuenta tamno  
        print("Fuera de Limite")  
        return V  
    ultimo=V[b]  
    for i in range(b,a-1):  
        V[i]=V[i-1]  
    V[a]=ultimo  
    return V  
# _____ funcion main _____  
n = leer()  
print("Ingrese valores para llenar el vector: ")  
vector = llenarVector(n)  
print("____ Vector original____")  
mostrar(vector,n)  
print("____ Rotar Segemento____")  
print(segmentoDilimitadorDerecha(vector,n))
```

Vector inicial	Resultado																												
<table border="1"><tr><td>2</td><td>1</td><td>5</td><td>7</td><td>1</td><td>3</td><td>5</td></tr><tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr></table>	2	1	5	7	1	3	5	0	1	2	3	4	5	6	Si a=2, b=5 <table border="1"><tr><td>2</td><td>1</td><td>3</td><td>5</td><td>7</td><td>1</td><td>5</td></tr><tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr></table>	2	1	3	5	7	1	5	0	1	2	3	4	5	6
2	1	5	7	1	3	5																							
0	1	2	3	4	5	6																							
2	1	3	5	7	1	5																							
0	1	2	3	4	5	6																							

DESK CHECK

```
Tamanho Vector=7  
Ingrese valores para llenar el vector:  
[0]=2  
[1]=1  
[2]=5  
[3]=7  
[4]=1  
[5]=3  
[6]=5  
____ Vector original____  
[2,1,5,7,1,3,5,]  
____ Rotar Segemento____  
____ Delimitando del segmento a rotar____  
Delimitador inicial a=2  
Delimitador final b=5  
[2, 1, 3, 7, 1, 3, 5]
```



PROBLEMA

5. Dado el vector V de tamaño N, con elementos enteros positivos. Ordenar sus elementos ascendenteamente, aplicando el Método Burbuja.

2	1	5	7	1	3	5
0	1	2	3	4	5	6

SOURCE IN PYTHON

```
# _____ function para leer vector _____
def leer():
    tam = int(input("Tamanho Vector="))
    return tam

# _____ function para generar vector _____
def llenarVector(tamanho):
    V = [0]*n
    for i in range(n):
        x=int(input(f"[{i}]="))
        V[i]=x
    return V
# _____ function mostrar vector _____
def mostrar(V, n):
    print("[",end="")
    for i in range(n):
        print(V[i],end="")
    print("]")

def burbuja(V,n):
    for j in range(n):
        for i in range(n-1):
            if V[i]>V[i+1]:
                aux = V[i]
                V[i] = V[i+1]
                V[i+1]=aux
    return V

# _____ function main _____
n = leer()

vector = llenarVector(n)
print("_____Mostrar Vector Original_____")
mostrar(vector,n)

print("METODO DE BURBUJA: ")
burbuja(vector,n)
mostrar(vector,n)

DESK CHECK
```

```
Tamanho Vector=7
[0]=2
[1]=1
[2]=5
[3]=7
[4]=1
[5]=3
[6]=5
_____Mostrar Vector Original_____
[2,1,5,7,1,3,5,]
METODO DE BURBUJA:
[1,1,2,3,5,5,7,]
```

PROBLEMA

6. Dado el vector V de tamaño N, con elementos enteros positivos. Ordenar sus elementos ascendenteamente, aplicando el Método de Selección.

2	1	5	7	1	3	5
0	1	2	3	4	5	6

SOURCE IN PYTHON

```
def dimension():
    n=int(input("ingrese la dimension del vector:"))
    return n

def llenar(n):
    v=[0]*n
    for i in range(n):
        v[i]=int(input(f"V[{i}]="))
    return v

def mostrar(v,n):
    for i in range(n):
        print(v[i], end=" ")

def seleccion(v,n):
    for k in range(n):
        menor=v[k]
        M=k
        for j in range(k+2,n):
            if v[j]<menor:
                menor=v[j]
                M=j
        v[M]=v[k]
        v[k]=menor
    return v

n=dimension()
vector=llenar(n)
print("_____Mostrar Vector Original_____")
mostrar(vector,n)
print()
print("_____ Metodo Seleccion_____")
print(seleccion(vector,n))
```

DESK CHECK

```
ingrese la dimension del vector:7
V[0]= 2
V[1]= 1
V[2]= 5
V[3]= 7
V[4]= 3
V[5]= 1
V[6]= 5
_____ Mostrar Vector Original_____
2 1 5 7 3 1 5
_____ Metodo Seleccion_____
[1, 1, 2, 5, 3, 7, 5]
```



PROBLEMA

7. Dado el vector V de tamaño N, con elementos enteros positivos. Ordenar sus elementos ascendenteamente, aplicando el Método de Inserción.

2	1	5	7	1	3	5	
0	1	2	3	4	5	6	

SOURCE IN PYTHON

```
# _____ function para leer vector _____
def leer():
    tam = int(input("Tamaño Vector="))
    return tam

# _____ function para generar vector _____
def llenarVector(tamano):
    V = [0]*n
    for i in range(n):
        x=int(input(f"[{i}]="))
        V[i]=x
    return V
# _____ function mostrar vector _____
def mostrar(V, n):
    print("[",end="")
    for i in range(n):
        print(V[i],end="")
    print("]")

def insercion(V,n):
    for i in range(1,n):
        e = V[i]
        k = i - 1
        while(k>=0 and e < V[k]):
            V[k+1]=V[k]
            k = k-1
        V[k+1] = e

    return V
# _____ funcion main_____
n = leer()
print("Ingrese valores para llenar el vector: ")
vector = llenarVector(n)
print("_____Vector original_____")
mostrar(vector,n)
print("_____metodo de insercion_____")
print(insercion(vector,n))
```

DESK CHECK

Tamano Vector=7

Ingrese valores para llenar el vector:

```
[0]=2
[1]=1
[2]=5
[3]=7
[4]=1
[5]=3
[6]=5
_____Vector original_____
[2,1,5,7,1,3,5,]
_____metod de insercion_____
[1, 1, 2, 3, 5, 5, 7]
```

PROBLEMA

8. Dado el vector V de tamaño N, con elementos enteros positivos. Ordenar sus elementos ascendenteamente, aplicando el Método Merge.

2	1	5	7	1	3	5	
0	1	2	3	4	5	6	

SOURCE IN PYTHON

```
# _____ function para leer tamaño del vector _____
def leer():
    tam = int(input("Tamaño del vector = "))
    return tam

# _____ function para generar vector _____
def llenarVector(tamano):
    V = [0]*tamano
    for i in range(tamano):
        x = int(input(f"Elemento {i+1} = "))
        V[i] = x
    return V

# _____ function para mostrar vector _____
def mostrar(V, n):
    print("[", end="")
    for i in range(n):
        print(V[i], end="," if i < n - 1 else "")
    print("]")

def merge_sort(a):
    #n = len(a) NO LO USAREMOS ESTAMOS ENVIANDO COMO PARAMETRO
    # Caso base: si el arreglo tiene un solo elemento, ya está ordenado
    if n == 1:
        return a

    # Dividir el arreglo en dos mitades sin usar primitivas
    mitad = n // 2
    arrayOne = [0]*mitad
    arrayTwo = [0]*(n-mitad) # Crear el segundo sub-arreglo
    # Copiar elementos manualmente a arrayOne y arrayTwo
    for i in range(mitad):
        arrayOne[i] = a[i]

    for i in range(mitad, n):
        arrayTwo[i-mitad] = a[i]
    # Llamadas recursivas para ordenar cada mitad
    arrayOne = merge_sort(arrayOne)
    arrayTwo = merge_sort(arrayTwo)
    # Combinar (merge) las dos mitades ordenadas
    return merge(arrayOne, arrayTwo)
```

CONTINUA EN LA SIGUIENTE PAGINA



```
def merge(a, b):
    n = len(a) + len(b)
    c = [0] * n
# Crear un arreglo 'c' de tamaño suficiente para almacenar a y b
    combinados
    i = 0 # Índice para el arreglo `a`
    j = 0 # Índice para el arreglo `b`
    k = 0 # Índice para el arreglo `c`

# Mientras ambos arreglos tengan elementos
while i < len(a) and j < len(b):
    if a[i] > b[j]:
        c[k] = b[j]
        j += 1
    else:
        c[k] = a[i]
        i += 1
    k += 1

# Copiar los elementos restantes de `a`, si los hay
while i < len(a):
    c[k] = a[i]
    i += 1
    k += 1

# Copiar los elementos restantes de `b`, si los hay
while j < len(b):
    c[k] = b[j]
    j += 1
    k += 1

return c

#_____ function main _____
n = leer()
print("Ingrese valores para llenar el vector:")
vector = llenarVector(n)

print("_____ Mostrar Original_____")
mostrar(vector, n)
#_____ show function merge _____
print("_____ Mostrar Merge _____")
print(merge_sort(vector))
DESK CHECK

Tamaño del vector = 7
Ingrese valores para llenar el vector:
Elemento 1 = 2
Elemento 2 = 1
Elemento 3 = 5
Elemento 4 = 7
Elemento 5 = 1
Elemento 6 = 3
Elemento 7 = 5
_____ Mostrar Original_____
[2, 1, 5, 7, 1, 3, 5]
_____ Mostrar Merge _____
[1, 1, 2, 3, 5, 5, 7]
```

PROBLEMA

9. Dado el vector V de tamaño N, con elementos enteros positivos. Ordenar sus elementos ascendenteamente, aplicando el Método Quick Sortmostrar el nuevo número que se conformó,

2	1	5	7	1	3	5
0	1	2	3	4	5	6

SOURCE IN PYTHON

```
#_____ function para leer tamaño del vector _____
def leer():
    tam = int(input("Tamaño del vector = "))
    return tam

#_____ function para generar vector _____
def llenarVector(tamano):
    V = [0] * tamano
    for i in range(tamano):
        x = int(input(f"Elemento {i+1} = "))
        V[i] = x
    return V

#_____ function para mostrar vector _____
def mostrar(V, n):
    print("[", end="")
    for i in range(n):
        print(V[i], end=", " if i < n - 1 else "")
    print("]")

def separacion(lista):
    if len(lista)<1:
        return []
    #else:
    #    return lista
    izquierda = []
    derecha = []
    pivot = lista[0]
    for i in range(1,len(lista)):
        if lista[i]<pivot:
            izquierda.append(lista[i])
        else:
            derecha.append (lista[i])
    #print(izquierda)
    #print(derecha)
    #print(pivot)
    return izquierda, pivot,derecha

def quickSortOr(lista):
    if len(lista) < 2:
        return lista
    izquierda, pivot, derecha = separacion(lista)
    return quickSortOr(izquierda) + [pivot] + quickSortOr(derecha)

#_____ function main _____
n = leer()
print("Ingrese valores para llenar el vector:")
vector = llenarVector(n)

print("_____ Mostrar Original_____")
mostrar(vector, n)
#_____ show function merge _____
print("_____ Mostrar quickSort _____")
print(quickSortOr(vector))
```



DESK CHECK

```
Tamaño del vector = 7
Ingrese valores para llenar el vector:
Elemento 1 = 2
Elemento 2 = 1
Elemento 3 = 5
Elemento 4 = 7
Elemento 5 = 1
Elemento 6 = 3
Elemento 7 = 5
Mostrar Original _____
[2, 1, 5, 7, 1, 3, 5]
Mostrar quickSort _____
[1, 1, 2, 3, 5, 5, 7]
```

PROBLEMA

10. Dado el vector V de tamaño N, ordenar sus elementos pares ascendenteamente.

Vector inicial

11	29	16	29	24	16	19	29	15	14
----	----	----	----	----	----	----	----	----	----

Resultado

11	29	14	29	16	16	19	29	15	24
----	----	----	----	----	----	----	----	----	----

SOURCE IN PYTHON

```
# _____ function para leer vector _____
def leer():
    tam = int(input("Tamanho Vector="))
    return tam
# _____ function para generar vector _____
def llenarVector(tamanho):
    V=[0]*n
    for i in range(n):
        x=int(input(f"[{i}]="))
        V[i]=x
    return V
# _____ function mostrar vector _____
def mostrar(V,n):
    print("[",end="")
    for i in range(n):
        print(V[i],end="")
    print("]")
# _____ function ordenarPares ASc. _____
def ordenarPares(V,n):
    for i in range(n):
        for j in range(i+1,n):
            #print(V[j])
            if V[i]%2==0 and V[j]%2==0:
                if V[i]>V[j]:
                    aux=V[i]
                    V[i]=V[j]
                    V[j]=aux
    return V
def ordenar(V,n):
    for i in range(n):
        for j in range(n-1):
            if V[j]%2==0:
                for k in range(j+1,n):
                    if V[k]%2==0:
                        if V[j]>V[k]:
                            #print(V[k])
                            if V[j]>V[k]:
                                aux=V[j]
                                V[j]=V[k]
                                V[k]=aux
    return V
# _____ function main _____
n = leer()
print("_____")
vector = llenarVector(n)
print("Mostrar Vector Original _____")
mostrar(vector,n)
res = ordenar(vector,n)
print("Ordenar Pares Asc. ")
print(res)
```

DESK CHECK

Tamanho Vector=10

[0]=11
[1]=29
[2]=16
[3]=29
[4]=24
[5]=16
[6]=19
[7]=29
[8]=15
[9]=14

Mostrar Vector Original _____
[11,29,16,29,24,16,19,29,15,14]
Ordenar Pares Asc.
[11, 29, 16, 29, 14, 16, 19, 29, 15, 24]



PROBLEMA

11. Cargar un vector T con n elementos y ordenar en forma ascendente todos los elementos múltiplos de 3 contenidos en este vector. Mostrar el vector T antes y después de la

Vector inicial	Resultado
[81 7 9 5 21 12 4]	[9 7 12 5 21 81 4]

SOURCE IN PYTHON

```
# _____ function para leer vector _____
def leer():
    tam = int(input("Tamanho Vector="))
    return tam
# _____ function para generar vector _____
def llenarVector(tamano):
    V = [0]*n
    for i in range(n):
        x=int(input(f"[{i}]="))
        V[i]=x
    return V
# _____ function mostrar vector _____
def mostrar(V, n):
    print("[",end="")
    for i in range(n):
        print(V[i],end="")
    print("]")
# _____ function Multiploss de 3 ordenar ASc. _____
def multiploTresAsc(V,n):
    for i in range(n):
        for j in range(i+1,n):
            if V[i]%3==0 and V[j]%3==0:
                #print(V[j])
                if V[i] >V[j]:
                    aux = V[i]
                    V[i] = V[j]
                    V[j] = aux
    return V
# _____ function main _____
n = leer()
print("_____")
vector = llenarVector(n)
print("____Mostrar Vector Original____")
mostrar(vector,n)
res = multiploTresAsc(vector,n)
print("Ordenar Pares Asc. ")
print("_____")
print(res)
```

DESK CHECK

Tamanho Vector=7

```
[0]=91
[1]=7
[2]=9
[3]=5
[4]=21
[5]=12
[6]=4
____Mostrar Vector Original____
[91, 7, 9, 5, 21, 12, 4]
Ordenar Pares Asc.

[91, 7, 9, 5, 12, 21, 4]
```

PROBLEMA

12. En un vector de dimensión par, ordenar la primera mitad de los elementos en forma ascendente y la segunda mitad en forma descendente.

Vector inicial	Resultado
[5 8 2 1 9 3]	[2 5 8 9 3 1]

SOURCE IN PYTHON

```
# _____ function para leer vector _____
def leer():
    tam = int(input("Tamanho Vector="))
    return tam
# _____ function para generar vector _____
def llenarVector(tamano):
    V = [0]*n
    for i in range(n):
        x=int(input(f"[{i}]="))
        V[i]=x
    return V
# _____ function mostrar vector _____
def mostrar(V, n):
    print("[",end="")
    for i in range(n):
        print(V[i],end="")
    print("]")
def burbuja(V,n,n1):
    for j in range(n//2):
        for i in range(n//2-1):
            if V[i]>V[i+1]:
                aux = V[i]
                V[i] = V[i+1]
                V[i+1]=aux
    for y in range(n1,n):
        for k in range(n1,n-1):
            if V[k]<V[k+1]:
                aux = V[k]
                V[k] = V[k+1]
                V[k+1]=aux
    return V
# _____ function main _____
n = leer()
print("____El valor debe ser PAR:____ ")
print("_____")
vector = llenarVector(n)
print("____Mostrar Vector Original____")
mostrar(vector,n)
n1=n//2
res = burbuja(vector,n,n1)
print("Orden Asc./Desc. ")
print(res)
```

DESK CHECK

Tamanho Vector=6

```
[0]=5
[1]=8
[2]=2
[3]=1
[4]=9
[5]=3
____Mostrar Vector Original____
[5, 8, 2, 1, 9, 3, ]
Orden Asc./Desc.
[2, 5, 8, 9, 3, 1]
```



PROBLEMA

13. Dado un vector de n elementos, realiza una función para ordenar ascendenteamente un segmento del mismo, delimitado por 2 parámetros a,b que se le pedirá al usuario.

Vector inicial	Resultado
Si $a=2, b=5$ [2 1 5 7 1 3 5] 0 1 2 3 4 5 6	[2 1 1 3 5 7 5] 0 1 2 3 4 5 6

SOURCE IN PYTHON

```
#_____ function para leer vector _____
def leer():
    tam = int(input("Tamanho Vector="))
    return tam
def leerParametro():
    print("Leer delimitador del segmento")
    a = int(input("Posicion Inicio ="))
    b = int(input(" Posision Fonal="))
    return a,b
#_____ function para generar vector _____
def llenarVector(tamanho):
    V = [0]*n
    for i in range(n):
        x=int(input(f"[{i}]="))
        V[i]=x
    return V
#_____ function mostrar vector _____
def mostrar(V, n):
    print("[",end="")
    for i in range(n):
        print(V[i],end="")
    print("]")
#_____ function kparametros ASc. _____
def ordenarParametroAB(V,n):
    inicio, final = leerParametro()
    if inicio < 0 or final >=n or inicio >= final:
        print("Índices no válidos.")
        return V
    for i in range(inicio,final+1):
        for j in range(i+1,final+1):
            if V[i] >V[j]:
                aux = V[i]
                V[i] = V[j]
                V[j] = aux
    return V
#_____ function main _____
n = leer()
print("_____")
vector = llenarVector(n)
print("____ Mostrar Vector Original____ ")
mostrar(vector,n)
res = ordenarParametroAB(vector,n)
print("Ordenar Pares Asc. \n",res)
Tamanho Vector=7
```

DESK CHECK

```
[0]=2
[1]=1
[2]=5
[3]=7
[4]=1
[5]=3
[6]=5
Mostrar Vector Original_
[2,1,5,7,1,3,5,]
Leer delimitador del segmento
Posicion Inicio =2
Posicion Fonal=5
Ordenar Pares Asc.

[2, 1, 1, 3, 5, 7, 5]
```



PROBLEMA

14. Dado un vector de n elementos, realiza una función para encontrar un elemento k. La función debe devolver la posición i, del primer elemento encontrado. Aplique el algoritmo para la búsqueda lineal.

Vector inicial	Resultado
Si $k=8$, [3 5 4 2 8 2 1] 0 1 2 3 4 5 6	"Elemento encontrado en la posición i=4"

SOURCE IN PYTHON

```
#_____ function para leer vector _____
def leer():
    tam = int(input("Tamanho Vector="))
    return tam
def leerBuscar():
    print("Leer elemento a buscar")
    a = int(input("Valor ="))
    return a
#_____ function para generar vector _____
def llenarVector(tamanho):
    V = [0]*n
    for i in range(n):
        x=int(input(f"[{i}]="))
        V[i]=x
    return V
#_____ function mostrar vector _____
def mostrar(V, n):
    print("[",end="")
    for i in range(n):
        print(V[i],end="")
    print("]")
#_____ function kparametros ASc. _____
def buscaElemento(V,n):
    data = leerBuscar()
    for j in range(n):
        if V[j]==data:
            buscar=V[j]
            return buscar,j
    print("No hay elementos a mostrar")
    return None, -1
#_____ function main _____
n = leer()
print("_____")
vector = llenarVector(n)
print("____ Mostrar Vector Original____ ")
mostrar(vector,n)
res, posicion = buscaElemento(vector,n)
print("____ ")
print(f" El elemento es = {res} en la posicion = {posicion} ")
```

DESK CHECK

Tamanho Vector=7

```
[0]=3
[1]=5
[2]=4
[3]=2
[4]=8
[5]=2
[6]=1
Mostrar Vector Original_
[3,5,4,2,8,2,1,]
Leer elemento a buscar
Valor =8
```

El elemento es = 8 en la posicion = 4



PROBLEMA

15. Dado un vector de n elementos, realiza una función para encontrar un elemento k. La función debe devolver la posición i, del elemento encontrado. Aplique el algoritmo para la búsqueda binaria.

Vector inicial						
3	5	4	2	8	2	1
0	1	2	3	4	5	6

Resultado
Si k=8, "Elemento encontrado en la posición i=6

SOURCE IN PYTHON

```
#_____ function para leer vector _____
def leer():
    tam = int(input("Tamanho Vector="))
    return tam
def leerBuscar():
    print("Leer elemento a buscar")
    a = int(input("Valor ="))
    return a
#_____ function para generar vector _____
def llenarVector(tamanho):
    V = [0]*n
    for i in range(n):
        x=int(input(f"[{i}]="))
        V[i]=x
    return V
#_____ function mostrar vector _____
def mostrar(V, n):
    print("[",end="")
    for i in range(n):
        print(V[i],end="")
    print("]")
#_____ function kparametros ASc. _____
def buscaBinario(V):
    data = leerBuscar()# Elemento a buscar
    inicio=0
    fin=len(V)-1
    while inicio <= fin:
        mitad= (inicio + fin)//2 #obtenemos la mitad
        if V[mitad]==data:
            buscar=V[mitad]
            #print(buscar)
            return mitad,buscar
        elif V[mitad]<data:
            inicio=mitad+1
        else:
            fin=mitad-1
    return -1
```

```
#_____ function ordenar _____
```

```
def ordenarVector(V,n):
    for i in range(n):
        for j in range(n-1):
            if V[j]>V[j+1]:
                aux = V[j]
                V[j]=V[j+1]
                V[j+1]=aux
    return V
```

```
#_____ function main _____
n = leer()
print("_____")
vector = llenarVector(n)
#vector=[3,5,4,2,8,2,1]
#print(vector.sort())
#n=len(vector)
print("_____Mostrar Vector Original_____")
mostrar(vector,n)
ordenarVector(vector,n)
resultado,valor = buscaBinario(vector)

if resultado != -1:
    print("_____")
    print(f" El elemento es = {valor} en la posicion = {resultado}")
else:
    print("_____")
    print(f" No hay elemntos a mostrar")
```

DESK CHECK

```
Tamanho Vector=7
_____
[0]=3
[1]=5
[2]=4
[3]=2
[4]=8
[5]=2
[6]=1
Mostrar Vector Original_____
[3, 5, 4, 2, 8, 2, 1]
Ler elemento a buscar
Valor =8

El elemento es = 8 en la posicion = 6
```

NOTA: el vector debes ordenar para que funcione el metodo busqueda binaria, ara ello usamos metodo ordenar o puedes usar la funcion primitiva como no nos dejan usar "sort (vector) esa funcion ordenaria el vector"



MATRICES



INGENIERÍA
DE SISTEMAS
UNIVERSIDAD PÚBLICA DE EL ALTO

PROBLEMA

16. Hallar el mayor elemento de la diagonal principal y el menor de la diagonal secundaria de una matriz cuadrada.

Ejemplo.

Matriz inicial				Resultado																
<table border="1"><tr><td>1</td><td>7</td><td>9</td><td>4</td></tr><tr><td>9</td><td>2</td><td>3</td><td>6</td></tr><tr><td>3</td><td>6</td><td>8</td><td>7</td></tr><tr><td>7</td><td>5</td><td>4</td><td>6</td></tr></table>				1	7	9	4	9	2	3	6	3	6	8	7	7	5	4	6	El mayor de la diagonal principal es 8 El menor de la diagonal secundaria es 3
1	7	9	4																	
9	2	3	6																	
3	6	8	7																	
7	5	4	6																	

SOURCE IN PYTHON

```
# _____ function para leer vector _____
def leer():
    f = int(input("Tamaño filas="))
    c = int(input("Tamaño filas="))
    return f,c

# _____ function para generar vector _____
def llenarMatriz(f,c):
    M = [[0]*c for i in range(f)]
    for i in range(f):
        for j in range(c):
            x=int(input(f"M[{i}][{j}]="))
            M[i][j]=x
    return M

# _____ function mostrar vector _____
def mostrar(M,f,c):
    print("[")
    for i in range(f):
        for j in range(c):
            print(M[i][j],end="")
        print()
    print("]")

def mostrarPrincipalSuperior(M, f,c):
    mayor=M[0][0]
    menor=M[0][c-1]
    for i in range(f):
        for j in range(c):
            if(i==j):
                if M[i][j]>mayor:
                    mayor=M[i][j]
            if i + j == c-1:
                if M[i][j]<menor:
                    menor=M[i][j]

    print("Mayor de la diagonal principal es =",mayor)
    print("Menor de la diagonal secundaria es =",menor)

# _____ function main _____
f,c = leer()
print("Ingrese valores para llenar Matriz:")
matriz = llenarMatriz(f,c)
print("_____ Mostrar Original _____")
mostrar(matriz,f,c)
print("_____ Diagonal Principal y secundario _____")
mostrarPrincipalSuperior(matriz, f,c)
```

DESK CHECK

```
Tamano filas=4
Tamano filas=4
Ingrese valores para llenar Matriz:
M[0][0]=1
M[0][1]=7
M[0][2]=9
M[0][3]=4
M[1][0]=9
M[1][1]=2
M[1][2]=3
M[1][3]=6
M[2][0]=3
M[2][1]=6
M[2][2]=8
M[2][3]=7
M[3][0]=7
M[3][1]=5
M[3][2]=4
M[3][3]=6
_____ Mostrar Original _____
[
1,7,9,4,
9,2,3,6,
3,6,8,7,
7,5,4,6,
]
```

_____ Diagonal Principal y secundario _____
Mayor de la diagonal principal es = 8
Menor de la diagonal secundaria es = 3



PROBLEMA

17 Ordenar ascendentemente los elementos de la diagonal principal.

Ejemplo:

Matriz inicial				Resultado			
5	8	8	9	2	8	8	9
3	2	4	4	3	3	4	4
2	1	3	8	2	1	5	8
1	2	1	8	1	2	1	8

SOURCE IN PYTHON

```
def leer():
    f = int(input("Tamaño filas: "))
    c = int(input("Tamaño columnas: "))
    return f, c

def llenarMatriz(f, c):
    M = [[0] * c for i in range(f)]
    for i in range(f):
        for j in range(c):
            x = int(input(f"M[{i}][{j}]="))
            M[i][j] = x
    return M

def mostrar(M, f, c):
    print("")
    for i in range(f):
        for j in range(c):
            print(M[i][j], end=" ")
        print()
    print("")

def ordenarDiagonalPrincipal(M, f, c):
    n = min(f, c)
    for i in range(n):
        min_index = i
        for j in range(i + 1, n):
            if M[j][j] < M[min_index][min_index]:
                min_index = j
        if min_index != i: # Solo intercambiar si es necesario
            # Intercambio secuencial
            temp = M[i][i]
            M[i][i] = M[min_index][min_index]
            M[min_index][min_index] = temp

    f, c = leer()
    print("Ingrese valores para llenar la matriz:")
    matriz = llenarMatriz(f, c)
    print("_____ Mostrar Original _____")
    mostrar(matriz, f, c)

    ordenarDiagonalPrincipal(matriz, f, c)

    print("_____ Mostrar Matriz con Diagonal Principal Ordenada _____")
    mostrar(matriz, f, c)
```

DESK CHECK

```
Tamaño filas: 4
Tamaño columnas: 4
Ingrese valores para llenar la matriz:
M[0][0]=5
M[0][1]=8
M[0][2]=8
M[0][3]=9
M[1][0]=3
M[1][1]=2
M[1][2]=4
M[1][3]=4
M[2][0]=2
M[2][1]=1
M[2][2]=3
M[2][3]=8
M[3][0]=1
M[3][1]=2
M[3][2]=1
M[3][3]=8
_____ Mostrar Original _____
5, 8, 8, 9,
3, 2, 4, 4,
2, 1, 3, 8,
1, 2, 1, 8,
1
_____ Mostrar Matriz con Diagonal Principal Ordenada _____
2, 8, 8, 9,
3, 3, 4, 4,
2, 1, 5, 8,
1, 2, 1, 8,
```



PROBLEMA

18 Cargar una matriz D de NxM elementos y ordenar los elementos de la fila K de la matriz en forma ascendente: Ejemplo:

Matriz inicial				Resultado			
Entrada: N=3 M=4 K=1							
61	8	2	11	61	8	2	11
36	10	40	20	10	20	36	40
7	5	45	19	7	5	45	19

SOURCE IN PYTHON

```
# _____ function para leer vector _____
def leer():
    f = int(input("Tamanho filas="))
    c = int(input("Tamanho Columnas="))
    return f,c

# _____ function para generar vector _____
def llenarMatriz(f,c):
    M = [[0]*c for i in range(f)]
    for i in range(f):
        for j in range(c):
            x=int(input(f"M[{i}][{j}]="))
            M[i][j]=x
    return M

# _____ function mostrar vector _____
def mostrar(M, f,c):
    print("[")
    for i in range(f):
        for j in range(c):
            print(M[i][j],end="")
        print()
    print("]")

def leerFila():
    print("_____ Fila a Ordenar_____")
    d = int(input("# de Fila = "))
    return d

def mostrarFilaAsc(M, f,c):
    x=leerFila()
    if x < 0 or x >= f:
        print("Número de fila no válido.")
        return M
    for i in range(c-1):
        for j in range(0,c-i-1):
            if M[x][j]> M[x][j+1]:
                aux=M[x][j]
                M[x][j]=M[x][j+1]
                M[x][j+1]=aux

    return M

# _____ function main _____
f,c = leer()
print("Ingrese valores para llenar Matriz: ")
matriz = llenarMatriz(f,c)
print("_____ Mostrar Original_____")
mostrar(matriz,f,c)
print("f_____ Diagonal fila K={k}_____")
mostrarFilaAsc(matriz, f,c)
mostrar(matriz,f,c)
```

DESK CHECK

```
Tamanho filas=3
Tamanho Columnas=4
Ingresar valores para llenar Matriz:
M[0][0]=61
M[0][1]=8
M[0][2]=2
M[0][3]=11
M[1][0]=66
M[1][1]=10
M[1][2]=40
M[1][3]=20
M[2][0]=7
M[2][1]=5
M[2][2]=45
M[2][3]=19
____ Mostrar Original_____
[
61,8,2,11,
66,10,40,20,
7,5,45,19,
]
f_____ Diagonal fila K={k}_____
____ Fila a Ordenar_____
# de Fila = 1
[
61,8,2,11,
10,20,40,66,
7,5,45,19,
1]
```



PROBLEMA

19. Generar la siguiente matriz cuadrada de dimensión N.

Resultado				
si K=4				si k=5
1	2	3	4	1
0	0	5	0	0
0	6	0	0	0
7	8	9	10	9
				10
				11
				12
				13

DESK CHECK

SOURCE IN PYTHON

```
#_____ function para leer vector _____
def leer():
    n = int(input("Tamanho Matriz cuadrada="))

    return n

#_____ function para generar vector _____
def llenarMatriz(f,c):
    M = [[0]*c for i in range(f)]
    return M

#_____ function mostrar vector _____
def mostrar(M,f,c):
    print("[")
    for i in range(f):
        for j in range(c):
            print(M[i][j],end=" , ")
        print()
    print("]")

def leerFila():
    print("_____ Fila a Ordenar_____")
    d = int(input("# de Fila ="))
    return d

def matrizZ(M, n):
    k=0
    for i in range(n):

        for j in range(n):
            if i==0:
                k=k+1
                M[0][j]=k
            elif i+j==n-1:
                k=k+1
                M[i][j]=k

            if i==n-1:
                k=k+1
                M[n-1][j]=k

    return M

#_____ function main _____
c = leer()
f = c# puedes usar n se gustas ya k es una mtriz cuadrada
print("Generando Matriz:")
matriz = llenarMatriz(f,c)
print("_____ Mostrar Original_____")
mostrar(matriz,f,c)
print("f_____ Mostrar Matriz Z_____")
matrizZ(matriz, c)
mostrar(matriz,f,c)
```

Tamanho Matriz cuadrada=5

Generando Matriz:

Mostrar Original

```
[  
0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 ,
```

f_____ Mostrar Matriz Z_____

```
[  
1 , 2 , 3 , 4 , 5 ,  
0 , 0 , 0 , 6 , 0 ,  
0 , 0 , 7 , 0 , 0 ,  
0 , 8 , 0 , 0 , 0 ,  
10 , 11 , 12 , 13 , 14 ,
```



a)

PROBLEMA

20. Generar las siguientes matrices especiales de NxN elementos:

SOURCE IN PYTHON

5	0	0	0	0
4	5	0	0	0
3	4	5	0	0
2	3	4	5	0
1	2	3	4	5

```
def leer():
# _____ function para leer vector _____
def leer():
    n = int(input("Tamanho Matriz cuadrada="))

    return n

# _____ function para generar vector _____
def llenarMatriz(f,c):
    M = [[0]*c for i in range(f)]
    return M

# _____ function mostrar vector _____
def mostrar(M,f,c):
    print("[")
    for i in range(f):
        for j in range(c):
            print(M[i][j],end=" , ")
        print()
    print("]")

def leerFila():
    print("_____ Fila a Ordenar_____")
    d = int(input("# de Fila = "))
    return d

def matrizZ(matriz, size):
    for i in range(size):
        for j in range(size):
            if j <= i: # Solo llenar la parte diagonal y hacia la izquierda
                matriz[i][j] = 5 - (i - j) # Calcular el valor a asignar

    return matriz
```

```
# _____ function main _____
c = leer()
f = c# puedes usar n se gustas ya k esuna mtriz cuadrada
print("Generando Matriz: ")
matriz = llenarMatriz(f,c)           DESK CHECK
print("_____ Mostrar Original_____")
mostrar(matriz,f,c)
print("f_____ Mostrar Matriz Z_____")
matrizZ(matriz, c)
mostrar(matriz,f,c)
```

Tamanho Matriz cuadrada=5
Generando Matriz:
Mostrar Original _____

```
[  
0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 ,  
]  
f_____ Mostrar Matriz Z _____
```

[
5 , 0 , 0 , 0 , 0 ,
4 , 5 , 0 , 0 , 0 ,
3 , 4 , 5 , 0 , 0 ,
2 , 3 , 4 , 5 , 0 ,
1 , 2 , 3 , 4 , 5 ,
]





b)

PROBLEMA

20. Generar las siguientes matrices especiales de NxN elementos:

SOURCE IN PYTHON

_____ function para leer vector _____

```
def leer():
    n = int(input("Tamanho Matriz cuadrada="))
```

```
return n
```

_____ function para generar vector _____

```
def llenarMatriz(f,c):
    M = [[0]*c for i in range(f)]
    return M
```

_____ function mostrar vector _____

```
def mostrar(M,f,c):
    print("[")
    for i in range(f):
        for j in range(c):
            print(M[i][j],end=" , ")
        print()
    print("]")
```

```
def generateMatrix(matrix,n):
```

```
for j in range(n):
    matrix[0][j] = j + 1
```

```
num = n + 1 # Comenzar la numeración después de n
```

```
for i in range(1, n - 1):
```

```
    for j in range(n):
        if j < n - 1 - i:
            matrix[i][j] = 1
        elif j == n - 1 - i:
            matrix[i][j] = num
            num += 1
        else:
            matrix[i][j] = 3
```

```
for j in range(n):
```

```
    if j < n - 1:
        matrix[n - 1][j] = num
        num += 1
    else:
```

```
        matrix[n - 1][j] = num # Cambiar a `num` en lugar de 3 para el último elemento
```

```
return matrix
```

b)

1	2	3	4	5	6
3	3	3	3	7	1
3	3	3	8	1	1
3	3	9	1	1	1
3	10	1	1	1	1
11	12	13	14	15	16

DESK CHECK

Tamanho Matriz cuadrada=6

Generando Matriz:

Mostrar Original _____

```
[  
0 , 0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 , 0 ,  
0 , 0 , 0 , 0 , 0 , 0 , ]
```

Mostrar Matriz Z _____

```
[  
1 , 2 , 3 , 4 , 5 , 6 ,  
1 , 1 , 1 , 1 , 7 , 3 ,  
1 , 1 , 1 , 8 , 3 , 3 ,  
1 , 1 , 9 , 3 , 3 , 3 ,  
1 , 10 , 3 , 3 , 3 , 3 ,  
11 , 12 , 13 , 14 , 15 , 16 , ]
```

_____ function main _____

```
c = leer()
```

f = c# puedes usar n se gustas ya k esuna mtriz cuadrada

```
print("Generando Matriz: ")
```

```
matriz = llenarMatriz(f,c)
```

```
print("_____ Mostrar Original_____")
```

```
mostrar(matriz,f,c)
```

```
print("f_____ Mostrar Matriz Z_____")
```

```
generateMatrix(matriz, c)
```

```
mostrar(matriz,f,c)
```

PROBLEMA

21 Generar la matriz MAT de tamaño NxN (N es par). Dividir la matriz en 4 partes iguales.

Resultado						
N=6	0	1	2	3	4	5
0	1	2	3	19	20	21
1	4	5	6	22	23	24
2	7	8	9	25	26	27
3	28	29	30	10	11	12
4	31	32	33	13	14	15
5	34	35	36	16	17	18

SOURCE IN PYTHON

```
#_____ function para leer vector _____
def leer():
    n = int(input("Tamanho Matriz cuadrada="))

    return n
```

```
#_____ function para generar vector _____
def llenarMatriz(f,c):
    M = [[0]*c for i in range(f)]
    return M
```

```
#_____ function mostrar vector _____
def mostrar(M, f,c):
    print("[")
    for i in range(f):
        for j in range(c):
            print(M[i][j],end=" , ")
        print()
    print("]")
```

```
def generarMmatrizDividida(matriz,N):
    if N % 2 != 0:
        print("N debe ser un número par para dividir la matriz en cuadrantes iguales.")
        return #Termina la función si N es impar
```

```
#matriz = [[0] * N for _ in range(N)]
contador = 1
mitad = N // 2
```

```
# Llenado del primer cuadrante (superior izquierdo) - primer cuadrante
for i in range(mitad):
    for j in range(mitad):
        matriz[i][j] = contador
        contador += 1
```

```
# Llenado del tercer cuadrante (inferior derecho) - cuartos cuadrante
for i in range(mitad, N):
    for j in range(mitad, N):
        matriz[i][j] = contador
        contador += 1
```

```
# Llenado del segundo cuadrante (superior derecho) - tercero cuadrante
for i in range(mitad):
    for j in range(mitad, N):
        matriz[i][j] = contador
        contador += 1
```

```
# Llenado del cuarto cuadrante (inferior izquierdo) - segundos número
for i in range(mitad, N):
    for j in range(mitad):
        matriz[i][j] = contador
        contador += 1

# Imprimir la matriz
for fila in matriz:
    print(fila)

#_____ function main _____
c = leer()
f = c# puedes usar n se gustas ya k esuna mtriz cuadrada
print("Generando Matriz:")
matriz = llenarMatriz(f,c)
print("_____ Mostrar Original_____")
mostrar(matriz,f,c)
print("f_____ Mostrar Matriz dividida en 4 partes_____")
generarMmatrizDividida(matriz, c)
```

DESK CHECK

```
Tamanho Matriz cuadrada=6
Generando Matriz:
_____ Mostrar Original_____
[
[0 , 0 , 0 , 0 , 0 , 0 ,
0 , 0 , 0 , 0 , 0 , 0 ,
0 , 0 , 0 , 0 , 0 , 0 ,
0 , 0 , 0 , 0 , 0 , 0 ,
0 , 0 , 0 , 0 , 0 , 0 ,
0 , 0 , 0 , 0 , 0 , 0 ,
]
f_____ Mostrar Matriz dividida en 4
[1, 2, 3, 19, 20, 21]
[4, 5, 6, 22, 23, 24]
[7, 8, 9, 25, 26, 27]
[28, 29, 30, 10, 11, 12]
[31, 32, 33, 13, 14, 15]
[34, 35, 36, 16, 17, 18]
```



PROBLEMA

22 Dada la matriz MAT de tamaño NxN (N es par). Dividir la matriz en 4 partes iguales y ordenar los elementos de cada parte de manera ascendente.

Matriz inicial							Resultado						
0	1	2	3	4	5	0	1	2	3	4	5		
0	6	1	3	19	24	21	1	3	6	19	21	24	
1	8	5	2	26	23	20	2	5	8	20	23	26	
2	7	9	4	22	27	25	4	7	9	22	25	27	
3	28	33	30	15	12	10	28	30	33	10	12	15	
4	35	32	29	11	14	17	29	32	35	11	14	17	
5	31	34	36	16	13	18	31	34	36	13	16	18	

SOURCE IN PYTHON

```
# _____ function para leer vector _____
def leer():
    n = int(input("Tamanho Matriz cuadrada="))

    return n

# _____ function para generar vector _____
def llenarMatriz(f,c):
    M = [[0]*c for i in range(f)]
    return M

# _____ function mostrar vector _____
def mostrar(M, f,c):
    print("[")
    for i in range(f):
        for j in range(c):
            print(M[i][j],end=" , ")
        print()
    print("]")

def burbuja(lista):
    n = len(lista)
    for i in range(n):
        for j in range(0, n-i-1):
            if lista[j] > lista[j+1]:
                temp = lista[j]
                lista[j] = lista[j+1]
                lista[j+1] = temp

def dividirOrdenarMatriz(N, matriz):
    mitad = N // 2

    # Extraer elementos de cada cuadrante
    cuadrante1 = [matriz[i][j] for i in range(mitad) for j in range(mitad)]
    cuadrante2 = [matriz[i][j] for i in range(mitad) for j in range(mitad, N)]
    cuadrante3 = [matriz[i][j] for i in range(mitad, N) for j in range(mitad)]
    cuadrante4 = [matriz[i][j] for i in range(mitad, N) for j in range(mitad, N)]
```



```
# Ordenar cada cuadrante usando el algoritmo de burbuja
burbuja(cuadrante1)
burbuja(cuadrante2)
burbuja(cuadrante3)
burbuja(cuadrante4)

# Volver a colocar los elementos ordenados en la matriz
index = 0
for i in range(mitad):
    for j in range(mitad):
        matriz[i][j] = cuadrante1[index]
        index += 1

index = 0
for i in range(mitad):
    for j in range(mitad, N):
        matriz[i][j] = cuadrante2[index]
        index += 1

index = 0
for i in range(mitad, N):
    for j in range(mitad):
        matriz[i][j] = cuadrante3[index]
        index += 1

index = 0
for i in range(mitad, N):
    for j in range(mitad, N):
        matriz[i][j] = cuadrante4[index]
        index += 1

# Imprimir la matriz resultante
for fila in matriz:
    print(fila)

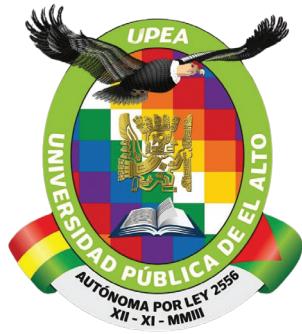
lista = [
    [6, 1, 3, 19, 24, 21],
    [8, 5, 2, 26, 23, 20],
    [7, 9, 4, 22, 27, 25],
    [28, 30, 33, 12, 11, 10],
    [32, 29, 35, 15, 14, 13],
    [31, 34, 36, 16, 18, 17]
]
#_____ function main _____
c = leer()
f = c# puedes usar n se gustas ya k esuna mtriz cuadrada

# aca solo llena de ceros la matriz
matriz = llenarMatriz(f,c)
matriz=lista
print("Generando Matriz: ")
print("_____Mostrar Original_____")
mostrar(matriz,f,c)

print("_____ Mostrar Matriz dividida en 4 partes_____")
dividirOrdenarMatriz(f, matriz)
```

DESK CHECK

```
Tamanho Matriz cuadrada=6
Generando Matriz:
_____ Mostrar Original _____
[
  6 ,  1 ,  3 ,  19 ,  24 ,  21 ,
  8 ,  5 ,  2 ,  26 ,  23 ,  20 ,
  7 ,  9 ,  4 ,  22 ,  27 ,  25 ,
  28 ,  30 ,  33 ,  12 ,  11 ,  10 ,
  32 ,  29 ,  35 ,  15 ,  14 ,  13 ,
  31 ,  34 ,  36 ,  16 ,  18 ,  17 ,
]
_____ Mostrar Matriz dividida en 4 partes _____
[1,  2,  3,  19,  20,  21]
[4,  5,  6,  22,  23,  24]
[7,  8,  9,  25,  26,  27]
[28, 29, 30, 10, 11, 12]
[31, 32, 33, 13, 14, 15]
[34, 35, 36, 16, 17, 18]
```



INGENIERÍA DE SISTEMAS

UNIVERSIDAD PÚBLICA DE EL ALTO

MATERIA : SIS-211 PROGRAMACIÓN I
PRÁCTICA GENERAL NRO 2
NOMBRE : PORFIRIO ELIAS QUISPE QUISPE
PARALELO : 2 " A"
TURNO : MAÑANA
GESTIÓN : II-2024
DOCENTE : LIC. PEREZ MARTINEZ KATYA MARICELA