

""""Se requiere realizar la solución al siguiente problema: Lucas y Micaela, poseen una agencia de venta de autos, y solicitan a ud. la realización de un programa informático. Puntualmente el sistema que le solicitan es de registro de pedidos. Este registro se realiza al finalizar el día. Se cargan los datos de las solicitudes HASTA QUE el modelo sea "FIN". Los modelos son Familiar y Utilitario, una vez cargado el modelo se carga la cantidad solicitada. (dos vectores) Se muestran en formato tabla, los datos ingresados:

- 1.- Se calcula la cantidad mínima solicitada y cual es el modelo asociado (SIN ORDENAR EL VECTOR).
 - 2.- Calcular y mostrar los porcentajes pedidos para ambos modelos (según las cantidades solicitadas) sobre el total
 - 3.- Ordenar los vectores según las cantidades solicitadas de forma programática (TENGA EN CUENTA QUE AL TRATARSE DE DOS VECTORES, EL ORDENAMIENTO TIENE QUE ESTAR RELACIONADO)
 - 4.- Indicar cual fue el promedio de cantidades solicitadas por cada modelo
 - 5.- Donde se encuentre una solicitud de un vehículo Familiar con la cantidad múltiplo de 3, se reemplazarán los valores por "reemplazo" y -1
 - 6.- Eliminar los datos de ambos vectores donde haya una cantidad solicitada de 7 unidades.
- NOTA: SE DEBE MOSTRAR EL RESULTANTE DE TODO LO SOLICITADO."""

#FUNCIONES

```
def cargar_pedidos(arr_mod, arr_cant):
```

```
    modelo = input("Ingrese Modelo (F/U o FIN para terminar): ").upper()
    while modelo != "FIN":
        while modelo != "F" and modelo != "U" or modelo == "":
            modelo = input("Error - Ingrese Modelo (F/U): ").upper()
        cantidad = int(input(f"Cantidad solicitada para {modelo}: "))
        while cantidad < 0:
            cantidad = int(input(f"Error - Cantidad solicitada para {modelo}: "))
        arr_mod.append(modelo)
        arr_cant.append(cantidad)
        modelo = input("Ingrese Modelo (F/U o FIN para terminar): ").upper()
```

```
def mostrar_tabla(arr_mod, arr_cant):
    titulo = "Modelo"
    titulo1 = "Cantidad"
    print("-----")
    print(f"\n\t{titulo:<15}{titulo1:>10}")
    print("-----")
    for i in range(len(arr_mod)):
        print(f"\t{arr_mod[i]:<15}{arr_cant[i]:>10}")
```

#1.- Se calcula la cantidad mínima solicitada y cual es el modelo asociado.

```
def buscar_minimo(arre):
    pos_min = 0
    for i in range(1, len(arre)):
        if arre[i] < arre[pos_min]:
            pos_min = i
    return pos_min
```

```
def suma_cant(arre):
```

```

acum = 0
for i in range (len(arre)):
    acum += arre[i]
return acum

def suma_fam(arr_mod, arr_cant):
    sum_fam = 0
    for i in range (len(arr_mod)):
        if arr_mod[i] == "F":
            sum_fam += arr_cant[i]
    return sum_fam

def cont_fam(arr_mod):
    cont_fam = 0
    for i in range (len(arr_mod)):
        if arr_mod[i] == "F":
            cont_fam += 1
    return cont_fam

def suma_util(arr_mod, arr_cant):
    sum_util = 0
    for i in range (len(arr_mod)):
        if arr_mod[i] == "U":
            sum_util += arr_cant[i]
    return sum_util

def cont_util(arr_mod):
    cont_util = 0
    for i in range (len(arr_mod)):
        if arr_mod[i] == "F":
            cont_util += 1
    return cont_util

#2.- Calcular y mostrar los porcentajes para ambos modelos sobre el total
def calcular_porcentajes(arr_mod,arr_cant):

    total_fam = suma_fam(arr_mod,arr_cant)
    total_util = suma_util(arr_mod,arr_cant)
    totales = suma_cant(arr_cant)

    if totales > 0:
        por_fam = (total_fam / totales) * 100
        por_util = (total_util / totales) * 100

        if total_fam > 0:
            print(f"\nPorcentaje del modelo Familiar: {por_fam:.2f}%")
        else:
            print("\nPorcentaje del modelo Familiar: No hay datos.")

        if total_util > 0:
            print(f"\nPorcentaje del modelo Utilitario: {por_util:.2f}%")
        else:
            print("\nPorcentaje del modelo Utilitario: No hay datos.")
    else:
        print("No hay suficientes datos.")

```

```
def swap(arr_mod, i, j):
    aux = arr_mod[i]
    arr_mod[i] = arr_mod[j]
    arr_mod[j] = aux
```

#3.- Ordenar los vectores según las cantidades (TIENE QUE ESTAR RELACIONADO)

```
def ordenar(arr_mod, arr_cant):
    for i in range(len(arr_cant) - 1):
        for j in range(i + 1, len(arr_cant)):
            if arr_cant[i] > arr_cant[j]:
                swap(arr_mod, i, j)
                swap(arr_cant, i, j)
```

#4.- Promedio de cantidades solicitadas por cada modelo

```
def promedio_por_modelo(arr_mod, arr_cant): # Necesitamos sumar y contar por separado
```

```
    sum_fam = suma_fam(arr_mod, arr_cant)
    count_fam = cont_fam(arr_mod)
    sum_util = suma_util(arr_mod, arr_cant)
    count_util = cont_util(arr_mod)
```

```
    if count_fam > 0:
        print(f"\nPromedio Familiar: {(sum_fam / count_fam):.2f}")
    else:
        print("Promedio Familiar: No hay datos.")
```

```
    if count_util > 0:
        print(f"Promedio Utilitario: {(sum_util / count_util):.2f}")
    else:
        print("Promedio Utilitario: No hay datos.")
```

#5.- Vehículo Familiar con múltiplo de 3, se reemplazarán por "Reemplazo" y -1

```
def reemplazar_familiar_mult3(arr_mod, arr_cant):
    for i in range(len(arr_mod)):
        if arr_mod[i] == "F" and arr_cant[i] % 3 == 0:
            arr_mod[i] = "Reemplazo"
            arr_cant[i] = -1
```

#6.- Eliminar los datos donde haya una cantidad solicitada de 7 unidades.

```
def eliminar_cantidad_7(arr_mod, arr_cant):
    i = 0
    while i < len(arr_cant):
        if arr_cant[i] == 7:
            arr_mod.pop(i)
            arr_cant.pop(i)
        else:
            i += 1
```

```
# ----- PROGRAMA PRINCIPAL -----
```

```
print("\n-----\n")
```

```
modelos = []
```

```
cantidades = []
```

```
cargar_pedidos(modelos, cantidades)
```

```
print("\n-----")
```

```

if len(cantidades)>0:
    # Mostrar tabla inicial
    print("\nDatos Ingresados:")
    mostrar_tabla(modelos, cantidades)
    print("\n-----")
    # 1. Mínimo
    index_min = buscar_minimo(cantidades)
    print(f"\nMínimo solicitado: {cantidades[index_min]} ({modelos[index_min]})")
    print("\n-----")
    # 2. Porcentajes
    calcular_porcentajes(modelos, cantidades)
    print("\n-----")
    # 3. Ordenamiento
    ordenar(modelos, cantidades)
    print("\nDatos ordenados por cantidad (ascendente):")
    mostrar_tabla(modelos, cantidades)
    print("\n-----")
    # 4. Promedios
    promedio_por_modelo(modelos, cantidades)
    print("\n-----")
    # 5. Reemplazo
    reemplazar_familiar_mult3(modelos, cantidades)
    print("\nDespués de reemplazar modelo F multiplos de 3:")
    mostrar_tabla(modelos, cantidades)
    print("\n-----")
    # 6. Eliminación
    eliminar_cantidad_7(modelos, cantidades)
    print("\nDespués de eliminar modelos con cantidad 7:")
    if len(cantidades) > 0:
        mostrar_tabla(modelos, cantidades)
    else:
        print("\nLa lista quedó vacía tras las eliminaciones.\n")
else:
    print("No se ingresaron pedidos.")
print("-----\n")
print("Fin del programa...")
print("-----\n")

```