```
"""Desarrollar un programa en Python para cada ejercicio utilizando funciones:
El Ministerio de Salud le solicitó realizar un programa para llevar adelante un censo de
salud. Para ello se tomaron los siguientes datos: peso, edad y sexo.
Se realizan dos encuestas, y los datos fueron tomados en dos estaciones ferroviarias: Retiro
y Constitución.
Cargar los datos de todos los encuestados de la estación correspondiente. El ingreso
concluye cuando el encuestador ingresa un número menor a 0 en el peso.
Todos los datos deben ser validados.
a) Mostrar los datos cargados.
b) Determinar promediando cuál de las dos estaciones tiene las personas con mayor peso
(mayor promedio).
c) Ordenar los datos correspondientes al que se ingresaron más encuestas y mostrarlo.
d) Generar un nuevo arreglo con los pesos de ambos lotes que superen el promedio general y
mostrarlo."""

# FUNCIONES...
def validar_estacion():
    estacion_ingresada = input("Ingrese la estación (R/C): ").upper()
    while estacion_ingresada != "R" and estacion_ingresada != "C":
        estacion_ingresada = input("Error - Ingrese la estación (R/C): ").upper()

    return estacion_ingresada

def validar_peso():     # Valida que el peso sea positivo.
    peso_ingreso = float(input("Ingrese el Peso/kg (< 0 para salir): "))
    while peso_ingreso == 0:
        peso_ingreso = float(input("Error - Ingrese el Peso/kg (< 0 para salir): "))
    return peso_ingreso

def validar_edad():     # Valida que la edad sea positiva y razonable (max 120).
    edad_ingreso = int(input("Ingrese la edad: "))
    while edad_ingreso <= 0 or edad_ingreso >= 120:
        edad_ingreso = int(input("Error - Ingrese la edad: "))
    return edad_ingreso

def validar_sexo():     # Valida que el sexo sea 'M' o 'F'.
    sexo_ingreso = input("Ingrese el sexo (M/F): ").upper()
    while sexo_ingreso != "M" and sexo_ingreso != "F":
        sexo_ingreso = input("Error - Ingrese sexo (M/F): ").upper()
    return sexo_ingreso

def cargar_datos(r_estacion,r_peso,r_edad,r_sexo,c_estacion,c_peso,c_edad,c_sexo):
    print("\n---------------------------------------")
    print("\t... INICIO DE CARGA ...")
    print("---------------------------------------\n")
    peso  = validar_peso()
    while peso >= 0:
        estacion = validar_estacion()
        edad = validar_edad()
        sexo = validar_sexo()

        if estacion == "R":
            r_estacion.append(estacion)
            r_peso.append(peso)
            r_edad.append(edad)
            r_sexo.append(sexo)
            print("\n>> Guardado en RETIRO.")
```

```
        else:
            c_estacion.append(estacion)
            c_peso.append(peso)
            c_edad.append(edad)
            c_sexo.append(sexo)
            print("\n>> Guardado en CONSTITUCIÓN.")

        print("\nDato cargado exitosamente...\n")
        print("----------------------------------------\n")
        peso  = validar_peso()
    print("\nCarga de datos finalizada...")

#A) Mostrar los datos cargados.
def mostrar_datos(titulo,arr_peso,arr_edad,arr_sexo):

    print(f"\n\t{titulo}")
    print("-" * 28)
    print(f"{"Peso":<10} {'Edad':<10} {'Sexo':<10}")
    print("-" * 28)
    for i in range(len(arr_peso)):
        print(f"{arr_peso[i]:<10.2f} {arr_edad[i]:<10} {arr_sexo[i]:<10}")

# Funciones auxiliares matemáticas
def suma_arreglo(arr):
    acum = 0
    i = 0
    while i < len(arr):
        acum += arr[i]
        i += 1
    return acum

# Calcula el promedio de un arreglo.
def calcular_promedio(arr):
    if len(arr) > 0:
        total = suma_arreglo(arr)
        promedio = total / len(arr)
    else:
        print("No se ingresaron datos.")
    return promedio

# B) Determinar promediando cuál de las dos estaciones tiene las personas con mayor peso
(mayor promedio).
def mayor_promedio(r_pesos, c_pesos):
    #Compara los promedios de peso de ambas estaciones.
    if len(r_pesos)> 0 and len(c_pesos) > 0:
        prom_R = calcular_promedio(r_pesos)
        prom_C = calcular_promedio(c_pesos)

    print("\nComparación de Promedios...\n")
    print(f"Retiro: {prom_R:.2f} kg")
    print(f"Constitución: {prom_C:.2f} kg")

    if prom_R > prom_C:
        print("\nRetiro tiene el mayor promedio de peso.")
    elif prom_C > prom_R:
        print("\nConstitución tiene el mayor promedio de peso.")
    elif prom_C == prom_R:
        print("El promedio de peso es igual en ambas estaciones.")
```

```
    else:
        print("\nNo hay suficientes datos.")

# Auxiliar para intercambiar valores en ordenamiento
def intercambiar(arr, i, j):
    aux = arr[i]
    arr[i] = arr[j]
    arr[j] = aux

# C) Ordenar los datos correspondiente al que se ingresaron más encuestas y mostrarlo.
def ordenar_lote_mayor(r_peso,r_edad,r_sexo,c_peso,c_edad,c_sexo):

    if len(r_peso) >= len(c_peso):
        arr_peso = r_peso
        arr_edad = r_edad
        arr_sexo = r_sexo
        titulo = "Retiro"
    elif len(c_peso) >= len(r_peso):
        arr_peso = c_peso
        arr_edad = c_edad
        arr_sexo = c_sexo
        titulo = "Constitución"

    elif len(arr_peso) >0:
        print(f"\nOrdenando: {titulo} ({len(arr_peso)} encuestas)...")
        # Ordenamiento por Burbuja (o Selección) en paralelo (por Peso ascendente)
        for i in range (len(arr_peso)):
            for j in range (len(arr_peso)):
                if arr_peso[i] > arr_peso[j]:
                    intercambiar(arr_peso, i, j)
                    intercambiar(arr_edad, i, j)
                    intercambiar(arr_sexo, i, j)

        mostrar_datos(f"{titulo} (Por Peso)", arr_peso, arr_edad, arr_sexo)
    elif len(c_peso) == len(r_peso):
        print("\nAmbos lotes tienen la misma cantidad de encuestas. No se realiza el
ordenamiento específico.")
    else:
        print("No hay suficientes datos para ordenar.")
# D) Generar un nuevo arreglo con los pesos de ambos lotes que superen el promedio general y
mostrarlo.
def pesos_superan_promedio(r_pesos, c_pesos):
    total_pesos = r_pesos + c_pesos
    promedio_general = calcular_promedio(total_pesos)
    pesos_superiores = []

    i = 0
    while i < len(total_pesos):
        if total_pesos[i] > promedio_general:
            pesos_superiores.append(total_pesos[i])
        i += 1
    if len(pesos_superiores) >0 :
        print(f"\nPromedio general de peso: {promedio_general:.2f} kg.")
        print("\nPesos que superan el promedio...\n")
        print(pesos_superiores)

#PROGRAMA PRINCIPAL...
```

```
print("\n  ... CENSO DE SALUD FERROVIARIO ...  ")
# Arreglos para Retiro (R)
est_retiro = []
pesos_retiro = []
edades_retiro = []
sexos_retiro = []
# Arreglos para Constitución (C)
est_constitucion = []
pesos_constitucion = []
edades_constitucion = []
sexos_constitucion = []
# Cargar los datos de todas los encuestados.
cargar_datos(est_retiro, pesos_retiro, edades_retiro, sexos_retiro, est_constitucion,
pesos_constitucion, edades_constitucion, sexos_constitucion)

if len(pesos_retiro) > 0 or len(pesos_constitucion) > 0:
    """ a) Mostrar los datos cargados.
b) Determinar promediando cuál de las dos estaciones tiene las personas con mayor peso
(mayor promedio).
c) Ordenar los datos correspondientes al que se ingresaron más encuestas y mostrarlo.
d) Generar un nuevo arreglo con los pesos de ambos lotes que superen el promedio general y
mostrarlo. """
    # A) Mostrar datos cargados.
    if len(pesos_retiro) > 0:
        mostrar_datos("RETIRO", pesos_retiro, edades_retiro, sexos_retiro)
    else:
        print("\nNo se registraron datos en RETIRO.")
    if len(pesos_constitucion) > 0:
        mostrar_datos("CONSTITUCIÓN", pesos_constitucion, edades_constitucion,
sexos_constitucion)
    else:
        print("\nNo se registraron datos en CONSTITUCIÓN.")
    print("\n----------------------------------------")

    # B) Determinar estación con mayor promedio de peso.
    if len(pesos_retiro) > 0 and len(pesos_constitucion) > 0:
        mayor_promedio(pesos_retiro, pesos_constitucion)
        print("\n----------------------------------------")

    # C) Ordenar la estación con más encuestados
    if len(pesos_retiro) > 0 and len(pesos_constitucion) > 0:
        ordenar_lote_mayor(pesos_retiro, edades_retiro, sexos_retiro, pesos_constitucion,
edades_constitucion, sexos_constitucion)
    else:
        print("No hay suficientes datos.")

    # D) Generar un nuevo arreglo con los pesos de ambos lotes que superen el promedio
general y mostrarlo.
    if len(pesos_retiro) > 0 and len(pesos_constitucion) > 0:
        pesos_superan_promedio(pesos_retiro, pesos_constitucion)
else:
    print("\nNo se ingresaron datos en ninguna estación.")
print("\n----------------------------------------")
print("\t... FIN DEL PROGRAMA ...")
print("----------------------------------------\n")
```