Trabajar con main.py y un paquete que contenga los módulos con funciones por separado. En los casos que las funciones reciban listas, verificar que el parámetro recibido sea de ese tipo, caso contrario retorna None. Documentar las funciones y detallar tipos de datos de los parámetros y del retorno.

Ejercicio 1:

Dadas las siguientes listas:
Nombres =
["Ana","Luis","Juan","Sol","Roberto","Sonia","Ulises","Sofia","Maria","Pedro","Anto
nio", "Eugenia", "Soledad", "Mario", "Mariela"]
Edades = [23,45,34,23,46,23,45,67,37,68,25,55,45,27,43]
Desarrollar una función que realice el ordenamiento de las listas por nombre de manera ascendente.

Ejercicio 2:

Dadas las siguientes listas:
Nombres = ["Matematica","Investigacion Operativa","Ingles","Literatura","Ciencias Sociales","Computacion","Ingles","Algebra","Contabilidad","Artistica", "Algoritmos", "Base de Datos", "Ergonomia", "Naturaleza"]
Puntos = [100,98,56,25,87,38,64,42,28,91,66,35,49,57,98]
Desarrollar una función que realice el ordenamiento de las listas por nombre de manera ascendente, si el nombre es el mismo, debe ordenar por puntos de manera descendente.

Ejercicio 3:

Dadas las siguientes listas:
Estudiantes =
["Ana","Luis","Juan","Sol","Roberto","Sonia","María","Sofia","Maria","Pedro","Anto
nio", "Eugenia", "Soledad", "Mario", "María"]
Apellidos =
[“Sosa”,”Gutierrez”,”Alsina”,”Martinez”,”Sosa”,”Ramirez”,”Perez”,”Lopez”,”Arregui”
,”Mitre”,”Andrade”,”Loza”,”Antares”,”Roca”,”Perez”]
Nota = [8,4,9,10,8,6,4,8,7,5,6,7,10,4,8]
Desarrollar una función que realice el ordenamiento de las listas por apellido de manera ascendente, si el apellido es el mismo, debe ordenar por nombre de manera ascendente, si el nombre también es el mismo, debe ordenar por nota de manera descendente.

Ejercicio 4:

Generar una función que mediante búsqueda binaria permite devolver una nota ingresando un apellido para las listas del ejercicio 3.