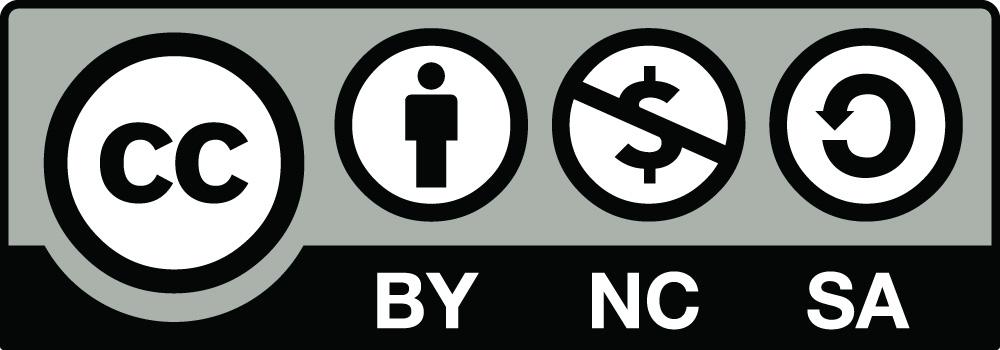
|  |  |
| --- | --- |
| **Estructura de Datos y Algoritmos**  **Grado Ingeniería Informática**  **Universidad Carlos III de Madrid**  **CURSO 2021-2022** | **Logotipo de la Universidad Carlos III de Madrid** |

****

**Grupo 801. Examen Parcial. 14 Marzo 2022.**

**Nombre:**

Instrucciones:

* Duración del examen: 60’.
* Descarga el zip parcial1.zip y descomprimelo.
* En el fichero parcial84.py deberás escribir tu solución. Además, añade un comentario python al principio del fichero con tu nombre y apellidos.
* NO MODIFIQUES el fichero test84.py (te ayuda a testear tu solución).
* Durante el examen, en tu equipo únicamente podrás tener abierto el entorno de desarrollo de Python (por ejemplo, spyder o pycharm). No puedes consultar ningún material (código, apuntes, libros, etc).
* Cuando queden 5 minutos el profesor te avisará para que subas tu solución. Podrás abrir el navegador y entrar en tu grupo reducido de aula global. Sube únicamente el fichero parcial84.py a la actividad “Primer Parcial” en Aula Global.
* Es tu responsabilidad comprobar que has subido la versión correcta del fichero.
* Los móviles deben estar apagados y nunca pueden estar encima de la mesa.
* No puedes salir del aula antes de que pasen 30 minutos.
* No está permitido ir al baño.

**Problema:** En la clase MySList, completa la función *merge (other)*, que recibe una lista “*other”*.

La función *merge* recibe un objeto de la clase MySList (other), y devuelve una nueva lista que incluye los elementos de la lista invocante y también a los de la lista other. La nueva lista de elementos debe estar ordenada de forma numérica y no admite duplicados. En caso de duplicados, la nueva lista sólo guardará una vez el elemento duplicado.

Las listas pueden tener diferente tamaño o estar vacías. La función devuelve la nueva lista generada. Si alguna de las listas no está ordenada la función devolverá *None*.

A continuación, tienes algunos ejemplos:

|  |  |  |
| --- | --- | --- |
| Lista invocante | Operación | Salida de la función |
| [1, 2, 3, 4] | l.*merge* ([5, 6, 7, 8, 9, 10]) | [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] |
| [5, 6, 7, 8, 9, 10] | l. *merge* ([[1, 2, 3, 4]) | [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] |
| [3, 4, 9, 10] | l. *merge* ([6, 1, 7, 9, 3, 10]) | None |
| [6, 1, 7, 9, 3, 10] | l. *merge* ([3, 4, 9, 10]) | None |
| [1, 1, 1, 2, 2, 2, 3, 5, 10, 10, 10] | l*. merge* ([1, 4, 6, 6, 8, 8, 9, 9, 9, 9]) | [1, 2, 3, 4, 5, 6, 8, 9, 10] |
| [3, 4, 9, 10] | l. *merge* (lista vacía) | [3, 4, 9, 10] |
| (lista vacía) | l. *merge* ([3, 4, 9, 10]) | [3, 4, 9, 10] |
| (lista vacía) | l. *merge* (lista vacía) | (lista vacía) |
| [0, 0, 0, 0, 1, 1, 1, 1, 8, 8, 8] | l. merge ([0, 0, 0, 0, 1, 1, 1, 1, 8, 8, 8]) | [0, 1, 8] |

No puedes añadir nuevos atributos o funciones a las clase SNode y MySList.

Para que la solución se considere correcta, la función propuesta debe ser **correcta** (resuelva el problema), **robusta** (no tenga errores y funcione para cualquier entrada) y **eficiente** en términos de complejidad temporal y espacial (evitar el uso de estructuras auxiliares). Además, el código debe ser fácil de entender y mantener.

No está permitido el uso estructuras de Python como los diccionarios o las listas.