Estructura de Datos y Algoritmos. **Grado en Ingeniería Informática** **Doble Grado en Ingeniería Informática y Administración de Empresas** **Universidad Carlos III de Madrid** **Caso Práctico. CURSO 2020-2021**	uc3m | Universidad Carlos III de Madrid

Campaña Vacunación Covid-19

El caso práctico se compone de tres fases. Este documento describe el enunciado de la primera fase. Los enunciados de la fase 2 y 3 se publicarán en las próximas semanas.

Fase 1 - Campaña de Vacunación de Covid-19 (10 puntos)

El objetivo de esta fase es permitir obtener información sobre el estado de la campaña de vacunación del Covid-19 en los centros de salud.

Cada centro de salud (clase HealthCenter) se define por un nombre (por ejemplo, "Los Frailes") y la lista de sus pacientes.

Nuestro sistema de salud guarda una información detallada de cada paciente, pero para este caso práctico, vamos a suponer que de cada paciente sólo es necesario conocer la siguiente información (ver clase Patient):

- name: Apellidos y Nombre (por ejemplo, "Segura, Isabel").
- year: Su año de nacimiento (por ejemplo, 1974)
- covid: Si ha pasado la covid (True para indicar si ha pasado la covid y False en otro caso).
- vaccine: Si ha sido vacunado o no (0 para indicar que no ha sido vacunado, 1 para indicar que ha recibido la primera dosis y 2 para indicar que ya ha recibido la segunda dosis).

El fichero fase1.py ya contiene la implementación de la clase Patient y parte de la implementación de la clase HealthCenter, que deberás modificar para incluir la siguiente funcionalidad:

- La función, **addPatient**, que reciba un paciente y lo añada a la lista de pacientes del centro de salud. La lista debe estar ordenada alfabéticamente y el método deberá insertar el nuevo paciente en su posición correspondiente. El paciente sólo se añade si no está almacenado en la lista de pacientes. **La complejidad del método debería ser lineal.**

- La función, **searchPatients**, que reciba los siguientes parámetros:
    - year: la función buscará todos los pacientes nacidos en dicho año y en años anteriores. Ten en cuenta que si el parámetro recibe el año actual (2021), la función buscará todos los pacientes.
    - covid: Su valor por defecto será None, que indicará que se buscarán todos los pacientes, hayan o no sufrido la covid. Si el valor del argumento es True, la función deberá buscar los pacientes que han sufrido la covid. Si su valor es False, se buscarán los pacientes que no han sufrido la covid.
    - vaccine: Su valor por defecto será None, que indica que se buscarán todos los pacientes, hayan recibido alguna dosis o no. Si el valor del argumento es 0, la función deberá buscar los pacientes que no hayan recibido ninguna dosis. Si el valor es 1, la función deberá buscar los pacientes que sólo hayan recibido una dosis. Si el valor es 2, la función deberá buscar los pacientes que hayan recibido dos dosis.

  La función devolverá un nuevo centro cuya lista de pacientes cumpla los criterios de búsqueda definidos por los argumentos de entrada de la función. **La complejidad del método debe ser lineal.**

- La función, **statistics**, que devuelva los siguientes valores:
    - Porcentaje de pacientes del centro de salud que ya han pasado la covid. El formato del porcentaje deberá ser 0.## (sólo dos decimales)
    - Porcentaje de pacientes mayores de 70 años (nacidos en 1950 o antes) que ya han pasado la covid. El formato del porcentaje deberá ser 0.## (sólo dos decimales)
    - Porcentaje de pacientes que aún quedan por vacunar (no han recibido ninguna dosis de la vacuna). El formato del porcentaje deberá ser 0.## (sólo dos decimales)
    - Porcentaje de mayores de 70 años (nacidos en 1950 o antes) que aún quedan por vacunar (no han recibido ninguna dosis de la vacuna). El formato del porcentaje deberá ser 0.## (sólo dos decimales)
    - Porcentaje de pacientes que ya han recibido la primera dosis. El formato del porcentaje deberá ser 0.## (sólo dos decimales)
    - Porcentaje de pacientes que ya han recibido la segunda dosis. El formato del porcentaje deberá ser 0.## (sólo dos decimales)

La complejidad de la función statistics debe ser lineal:

- Una función, **merge**, que reciba un objeto de la clase HealthCenter, other, y que devuelva un nuevo centro de salud cuya lista de pacientes incluya a los pacientes del centro invocante, y también a los pacientes del centro other. Recuerda que la lista de pacientes debe estar ordenada de forma alfabética y que no admite duplicados. En caso de duplicados, el nuevo centro sólo guardará el paciente del centro invocante (self). No está permitido que utilices un algoritmo de ordenación para ordenar dicha lista. **La complejidad del método debe ser lineal.**
- Una función, **minus**, que reciba un objeto de la clase HealthCenter, other, y que devuelva un nuevo centro de salud que contenga a los pacientes del centro invocante, pero estos pacientes no pueden pertenecer al centro other. Recuerda que la lista de pacientes debe estar ordenada alfabéticamente, y que no admitir duplicados. No está permitido que utilices un algoritmo de ordenación para ordenar dicha lista. **Tu solución debe ser lo más eficiente posible.**
- Un método, **inter**, que reciba un objeto de la clase HealthCenter, other, y que devuelva un nuevo centro de salud cuya lista de pacientes sólo incluya a aquellos pacientes que pertenecen a ambos centros de salud. Recuerda que su lista de pacientes debe estar ordenada alfabéticamente, y que no admite duplicados. Es decir, cada paciente, aunque aparezca en los dos centros, sólo se añadirá una vez al nuevo centro. No está permitido que utilices un algoritmo de ordenación para ordenar dicha lista. **Tu solución debe ser lo más eficiente posible.**

Normas:

1. El caso práctico debe ser realizado por un grupo formado por dos miembros (ambos deben pertenecer al mismo grupo reducido). En ningún caso se permitirá grupos con más de dos miembros. Tampoco se permiten grupos individuales ya que una de las competencias a evaluar será el trabajo en equipo. Si no tienes compañero, por favor, envía un correo a tu profesor de prácticas.
2. Junto con el enunciado de esta fase, se entrega al estudiante lo siguientes ficheros:
    a. La carpeta data que contiene una serie de ficheros tsv con datos para cargar las estructuras. **Estos ficheros no pueden ser modificados en ningún caso. Si detectas algún error, por favor comunícaselo a tu profesor.**
    b. El fichero dlist.py que contiene la implementación de la clase lista doblemente enlazada y nodo doblemente enlazado. **Este fichero no puede ser modificado en ningún caso. Si detectas algún error, por favor comunícaselo a tu profesor.**

c. El fichero fase1.py que tendrás que completar para elaborar la solución a esta fase. El fichero ya contiene algo de código (por ejemplo, la carga de datos para el centro de salud), que te ayudará a empezar.
d. El fichero unittest-fase1.py contiene los tests necesarios para evaluar cada uno de los métodos propuestos en el caso práctico. **Este fichero no puede ser modificado en ningún caso. Si detectas algún error, por favor comunícaselo a tu profesor.** Te recomendamos que inicialmente no ejecutes todo el test, sino que te centres en un método (por ejemplo, addPatient). Te facilitará comentar el resto de métodos tests. Según vayas avanzando en la implementación, podrás ejecutar todos los métodos test en la misma ejecución. Este fichero te dará una nota provisional de la fase 1. La nota final dependerá de esta nota provisional, de tu nota en la defensa y de la eficiencia de los métodos.

3. **En la implementación del caso práctico, no está permitido utilizar estructuras de Python como Listas, Queues o Diccionarios, etc.**
4. Sin embargo, sí puedes utilizar las implementaciones de SList o DList, que hemos estudiado en clase. Si deseas utilizar una pila o cola, su implementación deberá estar basada en lista enlazada (es decir, internamente la cola o la pila no podrá estar implementada con un lista de Python). **No está permitido modificar las implementaciones de SList o DList. Usa directamente la que te proporcionamos, de esta forma, te podrás evitar errores por usar una versión de lista incorrecta.**

5. Modo de entrega: En aula global, se publicará una tarea titulada '**Entrega Fase 1**'. La fecha de Entrega para esta primera fase será **14 de Abril, 23.59**.
6. Formato de entrega: un zip cuyo nombre sea los dos NIAS de los alumnos que forman el grupo, separados por un guión. Por ejemplo, *10001234-10005678.zip.* Sólo uno de los dos miembros, será el encargado de subir la solución del grupo. El zip deberá contener el fichero con fase1.py con tu solución.
7. **Defensa**: durante la clase presencial del **Jueves 15 de Abril**. La defensa del caso práctico es un examen oral. La asistencia es obligatoria. Si algún alumno no asiste, su calificación en el caso práctico será NP. Durante esta defensa, el profesor planteará a cada miembro del equipo una serie de preguntas que deberá responder de forma individual. Cada miembro del equipo debe ser capaz de discutir cualquiera de las decisiones tomadas en el diseño e implementación de cualquiera de las funcionalidades descritas en el caso práctico. La nota final del caso práctico estará condicionada por la nota obtenida en la defensa. Si un alumno no es capaz de discutir y defender ciertas decisiones, no será calificado con respecto a estas, aunque hayan sido correctamente implementadas.

8. Se recomienda seguir las recomendaciones descritas en Zen of Python (https://www.python.org/dev/peps/pep-0020/) y la guía de estilo (https://www.python.org/dev/peps/pep-0008/) publicada en la página oficial de Python.