Universidad de San Carlos de Guatemala
Escuela de Ingeniería en Ciencias y Sistemas, Facultad de Ingeniería
Introducción a la programación y computación 2, 1er. Semestre 2021.

PROYECTO 1 IPC2

202110568 – Damián Ignacio Peña Afre

Resumen

Se implementó un sistema capaz de simular el comportamiento de una rejilla que representan células del tejido de un paciente.

El sistema está diseñado para analizar todos los patrones generados en un número determinado de periodos para después diagnosticar el tipo de enfermedad, ya sea que se repita el patrón o estado inicial, o bien algun otro patrón distinto del inicial. Se puede cargar en la aplicación un archivo xml que contenga la información de todos los pacientes a los que se requiere analizar. Asimismo, el sistema es capaz de exportar en otro archivo o xml los resultados de todos los pacientes, o si se requiere, de forma individual. Es posible generar también un reporte en PDF donde se presentan todas las rejillas que se generaron en la simulación de uno o de todos los pacientes.

El programa puede beneficiar a las múltiples instituciones que realizan estudios epidemiológicos, brindándoles una forma sencilla e interactiva de realizar diagnósticos.

Palabras clave

Software de simulación, epidemiología, estructuras de datos

Abstract

It was developed a system capable of simulating the behavior of a grid that emulate the cells of a patient.

The system is designed to analyze all the patterns generated in a given number of periods in order to diagnose the type of disease, whether the initial pattern is repeated, or some other pattern than the initial one. It can be loaded an xml file that contains the information of all the patients that need to be analyzed. Likewise, the system is capable of exporting the results of all patients to an xml file, or if required, individually. It is also possible to generate a PDF report where all the states that were generated in the simulation one by one or of all the patients.

The program can benefit the multiple institutions that carry out epidemiological studies, providing them with a simple and interactive way of making diagnoses.

Keywords

Simulation software, epidemiology, data structures

Introducción

El presente proyecto tiene la finalidad de implementar un programa de simulación, para ello se pretende hacer uso de estructuras que permitan manejar la memoria dinámica, como lo son las listas enlazadas. Se analizarán también las ventajas y desventajas de las estructuras realizadas.

El uso de programación orientada a objetos es inherente a la construcción de la aplicación por lo que también se examinará por qué el uso de ciertas clases y formas de estructurar el proyecto.

El entorno gráfico resulta muy importante para el usuario final y es clave considerar el uso de librerías para crear un entorno amigable para el usuario.

Estructuras de datos

Llega un punto en el que las estructuras básicas o datos primitivos como *boolean, int o float* ya no nos son suficientes para realizar lo que requiere una aplicación y mucho menos cuando se requiere agrupar de alguna manera tipos de datos compuestos. Es ahí donde entran las estructuras de datos.

"Una estructura de datos es una colección de datos que pueden ser caracterizados por su organización y las operaciones que se definen en ella." (Ferrer, 2018)

Y es por esto que nos podemos aprovechar de una de las ventajas del lenguaje *Python* ya que este nos permite crear tipos de datos adicionales y por tanto generar estas estructuras necesarias para la resolución de algun problema en específico.

Dentro de las estructuras de datos de encuentran aquellas que son dinámicas y por tanto no tienen limitaciones en el tamaño de memoria que ocupan. Mediante *punteros* es posible construir estos tipos de estructuras.

Otra subclasificación de las estructuras de datos dinámicas es aquellas que se denominan como lineales y particularmente se analizará la listas enlazadas.

"Una lista es una colección de elementos organizados en secuencia que puede crecer y decrecer libremente y a cuyos elementos individuales se puede acceder, insertar y eliminar en cualquier posición." (Ferrer, 2018).

Entre algunas de las operaciones que se pueden realizar con las listas se encuentra: insertar, eliminar o localizar un elemento; determinar el tamaño de la lista; recorrer la lista; entre muchos otros.

Un *nodo* es un elemento de la lista que almacena la referencia algun otro u otros nodos y tiene un espacio reservado para almacenar algun dato.

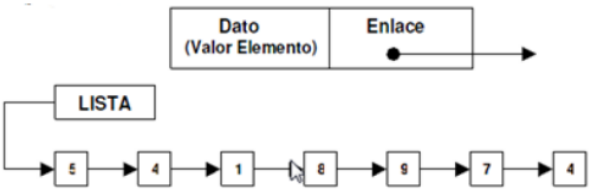

Figura 1. Lista enlazada

Fuente: Ferrer, 2018, pág.: 63.

Específicamente existen bastantes tipos de listas enlazadas, aquellas que tienen 2 apuntadores para identificar su *nodo* siguiente y anterior se denominan doblemente enlazadas. Para realizar una lista doblemente enlazada podrías seguir el siguiente diagrama de clases:

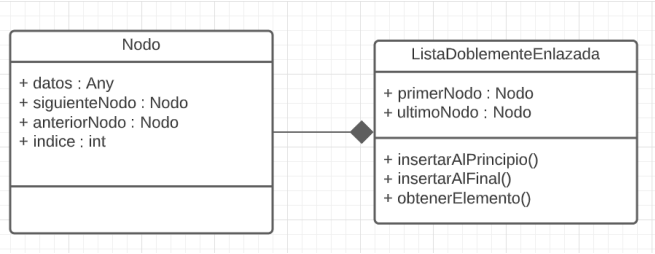

Figura 2. Diagrama de clases lista doblemente enlazada

Fuente: Elaboración propia, 2022.

Para brindar una solución al problema de la rejilla de células se propone armar una matriz con listas dobles, esto con el objetivo de hacer un poco mas eficiente la búsqueda, al no tener que recorrer todos los nodos en busca de cierto valor.

Se propone entonces, crear una lista doblemente enlazadas para los índices de columna y dentro del cuerpo o información del nodo de estos índices otra lista doblemente enlazada que representarían las columnas de dicha fila.

En el cuerpo de los nodos que forman las columnas es posible, gracias en parte al tipado dinámico de *Python*, introducir cualquier otro tipo de datos. Por simplicidad se podría guardar un valor *booleano* que representa si la célula esta infectada.

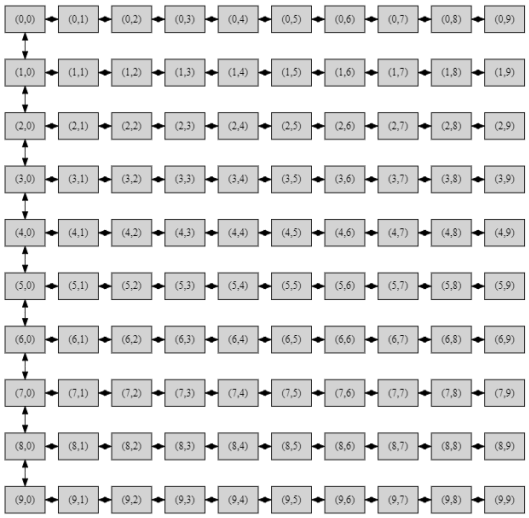

Figura 3. Matriz con listas doblemente enlazadas

Fuente: Elaboración propia, 2022.

Aprovechando la lista con nodos índices creada previamente se puede realizar un historial de estados de una simulación.

Lógica de la aplicación

El punto de entrada para la información dentro de la aplicación es la lectura del archivo *xml*, para ello puede ser utilizada una de las librerías preinstaladas en *Python* que es ampliamente utilizada para manejar el *DOM*. El *DOM* es un representación del documento por medio de objetos de manera que es mucho más fácil manejar sus elementos. Y es por ello por lo que dicha librería tiene el nombre de *minidom.*

Posteriormente a realizar la lectura del archivo se extraen los datos requeridos para un nuevo registro de paciente.

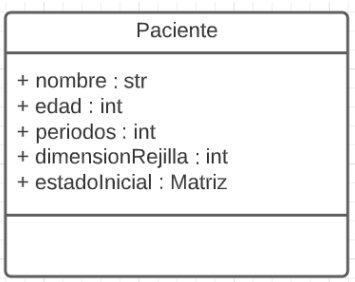

Figura 3. Clase paciente

Fuente: Elaboración propia, 2022.

La clase paciente puede ser pasada por parámetro a un objeto simulación para que éste guarde todos los elementos necesarios para generar información de utilidad

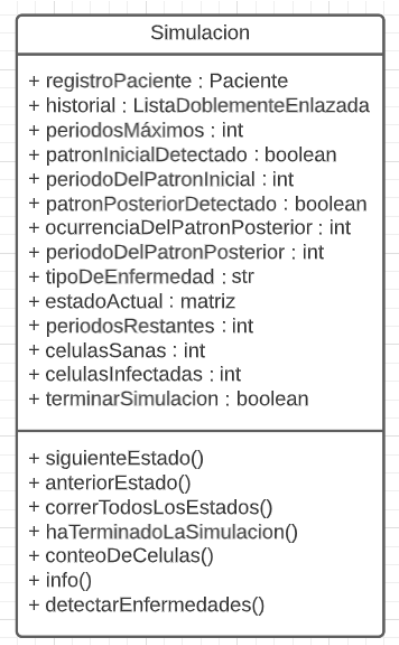

Figura 4. Clase Simulación

Fuente: Elaboración propia, 2022.

Posteriormente se puede generar reportes con la información de una simulación. Como se mencionó al principio, en un archivo *xml* o *PDF*. Se utiliza un software de visualización de gráficos llamado *Graphviz* en con junto con la librería *ElementTree* para generar el nuevo archivo *xml*.

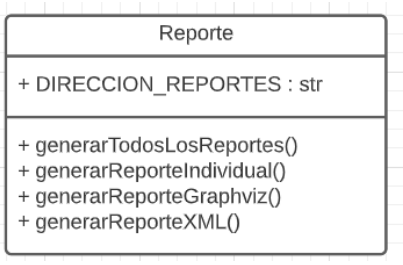

Figura 5. Clase Reporte

Fuente: Elaboración propia, 2022.

Entorno gráfico

Se utiliza la librería *Tkinter* propiamente instalada en *Python* para crear una interfaz gráfica fácil de utilizar y que muestra mensajes de retroalimentación al usuario cuando es necesario.

Conclusiones

- Se pueden utilizar estructuras de datos dinámicas lineales para representar otro tipo de estructuras más complejas.

- El uso de listas doblemente enlazadas como índice de fila provee una búsqueda más eficiente cuando se requiere implementar una matriz.

Referencias bibliográficas

Yarleque Ferrer, R. E. (2018). Estructura de datos Introducción, conceptos, tipos de datos, clasificación general, arrays, listas enlazadas, pilas, colas, inicialización y asignación de valores.