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.

ANALIZADOR DE TEJIDO DE PACIENTES

202109754 – Aldo Saúl Vásquez Moreira

Resumen

El presente proyecto realizado en conjunto con el Laboratorio de Investigación Epidemiológica de Guatemala tendrá un alto impacto en la medicina moderna. La institución realizó diversas investigaciones acerca de la forma en que las enfermedades infectan las células del cuerpo humano y se expanden produciendo enfermedades graves e incluso la muerte. Gracias a esta ardua labor, lograron identificar los patrones que determinan el comportamiento de las enfermedades; con dicho patrón fue posible desarrollar un algoritmo el cual simula el comportamiento de las células dentro del tejido de cada paciente. Con el algoritmo desarrollado, esto se llevó al código por medio del lenguaje de programación Python, permitiendo al usuario ingresar al sistema la información correspondiente a cada paciente por medio de un archivo en formato XML. Ya con la información cargada al sistema, este permite visualizar el comportamiento paso por paso del tejido que el usuario seleccione. Seguidamente, este generará un archivo de salida con el resultado del paciente.

Palabras clave

1. *Célula*
2. *Tejido*
3. *Patrón*

Abstract

The project developed in collaboration with the Epidemiological Research Laboratory of Guatemala will be really important for the modern medicine. This institution made a lot of researches to understand and predict the way diseases infect the cells of the human body and spread causing serious illness and even death. Finally, they were able to identify the patterns that determine how diseases affect out health. This pattern allowed to develop and algorithm that simulates the behavior of cells within the tissue of each patient. With the algorithm developed, this was coded using the Python programming language, allowing the user to enter the information corresponding to each patient into the system by means of file in XML format. Once the information is loaded into the system, it allows the user to visualize the step-by-step behavior of the tissue selected by the user. Then, it will generate an output file with the patient's result.

Keywords

1. *Cell*
2. *Tissue*
3. *Pattern*

Introducción

Con este ensayo se pretende familiarizarlo a usted como lector con el tema a tratar, la problemática planteada y la solución dada. Primeramente, se debe aclarar que es de conocimiento de todos que existen enfermedades que se desarrollan más rápido que otras e incluso llegan a ser mortal; por otro lado, existen otras enfermedades que se desarrollan de una forma más lenta y a pesar de que sean graves no son mortales. Planteado esto, es muy alta la importancia de algún sistema que permita determinar y predecir la forma en que una enfermedad se desarrollará en nuestro cuerpo para así tomar el tratamiento adecuado lo más pronto posible. Dada la necesidad y gracias al trabajo de investigación del Laboratorio de Investigación Epidemiológica de Guatemala fue posible desarrollar un software que cumple con todos los requerimientos planteados anteriormente.

Desarrollo del tema

Para el desarrollo de este proyecto se utilizaron documentos con formato XML para el ingreso y salida de información del sistema haciendo uso de ElementTree. Sabiendo que XML es un formato de datos inherentemente jerárquico, y la forma más natural de representarlo es con un árbol. ElementTree tiene dos características para este propósito: ElementTree el cual representa todo el documento XML como un árbol y Element el cual representa un solo nodo en este árbol. Las interacciones con todo el documento (leer y escribir en/ desde archivos) se realizan en el nivel ElementTree. Ahora bien, las interacciones con un solo elemento XML y sus subelementos se realizan por medio del nivel Element.

Para el manejo de la información se determinaron dos reglas importantes:

1. Toda célula contagiada, continúa contagiada si tiene 2 o 3 células contagiadas en las celdas vecinas, de lo contrario sanará para el siguiente periodo.
2. Cualquier célula sana que tenga exactamente 3 células contagiadas vecinas se contagiará para el siguiente periodo.

También se determinaron dos reglas para determinar si una enfermedad sería grave o mortal:

1. Que el patrón inicial se repita siempre después de “N” periodos, en este caso la enfermedad produce un caso grave. Si “N” es igual a 1, entonces el paciente morirá a causa de la enfermedad, ya que esta será incurable.
2. Que algún patrón distinto a la inicial se repita luego de “N” periodos cada “N_1” periodos, en este caso la enfermedad producirá un caso grave. Si “N_1” es igual a 1, entonces el paciente morirá a causa de la enfermedad ya que esta serpa incurable, en caso de que “N_1” sea mayor que 1 la enfermedad será grave.

Con esta información clara y luego de tomar los datos necesarios del archivo con formato XML se almacenó la información. Para esto se hizo uso de listas enlazadas ya que una consideración importante es que el software haga un uso óptimo de los recursos que el equipo ofrece.

Dando un breve concepto de las listas enlazadas, estas son unas de las estructuras de datos fundamentales la

cual puede ser usada para implementar otras estructuras de datos. Estas consisten en una secuencia de nodos, en los que se guardan campos de datos arbitrarios y una o dos referencias, enlaces o punteros al nodo anterior y posterior. En este caso en específico únicamente se utilizaron punteros hacia el nodo siguiente ya que el caso se podía manejar perfectamente de esa manera y era ideal para optimizar el rendimiento del equipo.

Además, el principal beneficio de las listas enlazadas respecto a los vectores convencionales es que el orden de los elementos enlazados puede ser diferente al orden de almacenamiento en la memoria o el disco, permitiendo que el orden de recorrido de la lista sea diferente al de almacenamiento. Es decir, que a pesar de que los elementos enlazados se encuentren uno a otro en la lista, esto no quiere decir que se almacenen contiguamente en la memoria física de la computadora, sino que, cada puntero hace referencia a una posición en memoria específica para recuperar la información correspondiente al nodo que se desea.

También es importante destacar que una lista enlazada es un tipo de dato autor referenciado porque contienen un puntero o enlace a otro dato del mismo tipo. Las listas enlazadas permiten inserciones y eliminación de nodos de cualquier punto de la lista en tiempo constante, pero no permiten un acceso aleatorio. Existen diferentes tipos de listas enlazadas: listas enlazadas simples, listas doblemente enlazadas, listas enlazas circulares y listas enlazadas doblemente circulares.

Estas listas pueden ser implementadas en muchos lenguajes. Aunque también es importante recalcar que muchos lenguajes ya implementan este tipo de listas, por lo tanto, no es necesario volver a diseñarlas.

Ahora que quedó claro el concepto de qué es una lista enlazada y sabiendo que la información recibida de los archivos con formato XML se almacenó de esta forma se mostrará a continuación la forma en que se diseñaron las clases del programa:

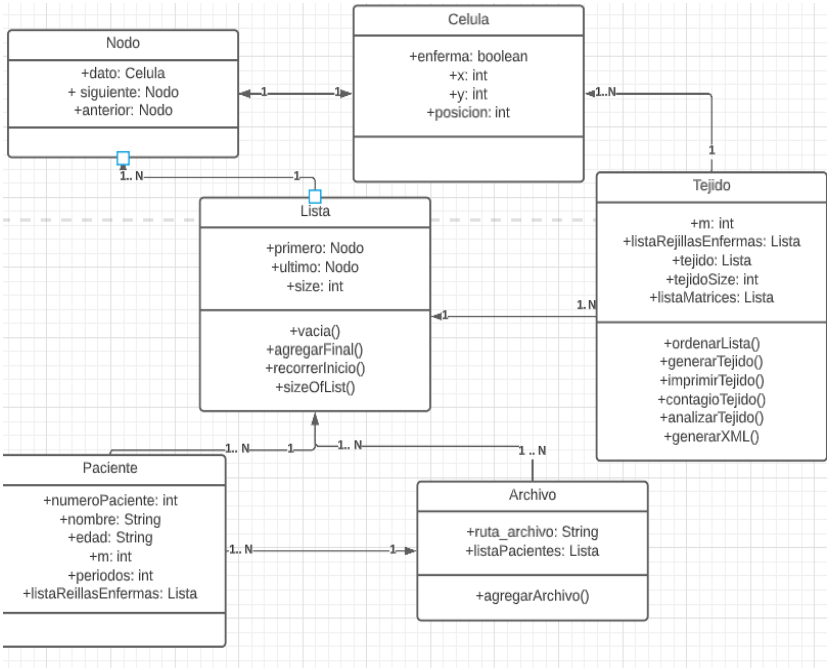

Figura 1. Diagrama de clases.

Fuente: elaboración propia.

Tal como se muestra en el diagrama de clases utilizado, podemos observar que para darle solución al problema fue necesario implementar seis clases. A continuación, se explicará la función que cumple dentro del sistema cada una de estas.

1. Clase Nodo: Esta se encarga de almacenar la información de cualquier dato que se ingrese en una

lista. Cuenta con los dos punteros, uno para el nodo anterior y otro para el nodo siguiente.

2. Clase Lista: Esta se encarga de enlazar cada uno de los nodos que se vayan creando. Además, permite eliminar nodos, así como recorrerse para mostrar cada uno de los nodos que la conforman.

3. Clase Célula: Almacena los datos referentes a las células, tales como si está enferma o no y su posición dentro del tejido.

4. Clase Tejido: Permite almacenar una cantidad determinada de células por medio de la clase Lista.

5. Clase Paciente: Almacena la información correspondiente a cada paciente.

6. Clase Archivo: Se encarga de almacenar la información de cada uno de los pacientes. Esta información la recibe por medio de un archivo XML.

Funciones o métodos principales

1. Función agregarArchivo: Esta función recibe el archivo XML, lo desestructura y toma cada uno de los datos de interés. Dichos datos son almacenados por medio de las listas enlazadas creadas anteriormente.

2. Función ordenarLista: Esta función se encarga de ordenar los datos recibidos de menor a mayor. Esto lo realizada por medio del método Bubble Sort.

3. Función generarTejido: Se encarga de acceder a la lista de las rejillas contagiadas del tejido a analizar. Con dicha información, esta genera el tejido inicial.

4. Función imprimirTejido: Esta función se encarga de mostrar en consola el tejido durante cada uno de los periodos indicados.

5. Función clearConsole: Se encarga de borrar el contenido de la consola para ir mostrando dinámicamente cada uno de los periodos del tejido a analizar.

6. Función contagioTejido: Posiblemente es la función más importante ya que se encarga de analizar el estado de las células vecinas de cada célula a analizar para indicar si esta célula se contagiará o no. Esto lo hace según los criterios indicados.

7. Función analizarTejido: Se encarga de verificar si un patrón se repite para así realizar el diagnóstico respectivo y decidir si la enfermedad del paciente es leve, grave o mortal.

8. Función prettify: Se encarga de darle la estructura correspondiente al archivo XML de salida.

9. Función generadorXML: Se encarga de generar el archivo XML con los datos del tejido del paciente.

Así, el funcionamiento en conjunto de todas estas clases permite obtener un resultado certero acerca de la enfermedad que tenga el paciente, permitiendo observar cómo se comportara a lo largo de los periodos que el usuario decida.

Anexos

0,0	(1,0)	(2,0)	(3,0)	(4,0)	(5,0)	(6,0)	(7,0)	(8,0)	(9,0)
0,1	(1,1)	(2,1)	(3,1)	(4,1)	(5,1)	(6,1)	(7,1)	(8,1)	(9,1)
0,2	(1,2)	(2,2)	(3,2)	(4,2)	(5,2)	(6,2)	(7,2)	(8,2)	(9,2)
0,3	(1,3)	(2,3)	(3,3)	(4,3)	(5,3)	(6,3)	(7,3)	(8,3)	(9,3)

Figura 2. Simulación de matriz

Fuente: elaboración propia.

1	x = 0	y = 0
2	x = 1	y = 0
3	x = 2	y = 0
4	x = 3	y = 0
5	x = 4	y = 0
6	x = 5	y = 0
7	x = 6	y = 0
8	x = 7	y = 0
9	x = 8	y = 0
10	x = 9	y = 0
11	x = 0	y = 1
12	x = 1	y = 1
13	x = 2	y = 1
14	x = 3	y = 1
15	x = 4	y = 1
16	x = 5	y = 1
17	x = 6	y = 1
18	x = 7	y = 1
19	x = 8	y = 1
20	x = 9	y = 1
21	x = 0	y = 2
22	x = 1	y = 2
23	x = 2	y = 2
24	x = 3	y = 2
25	x = 4	y = 2
26	x = 5	y = 2
27	x = 6	y = 2
28	x = 7	y = 2
29	x = 8	y = 2
30	x = 9	y = 2

Figura 3. Simulando comportamiento de una matriz con una lista enlazada

Fuente: elaboración propia.

posicion = (m*y)+(x+1)

Figura 3. Fórmula para hallar posición de las células dentro de la matriz

Fuente: elaboración propia.

Conclusiones

Finalmente, se puede decir que la implementación de todas estas técnicas de programación y estructuras de datos proporcionan una solución adecuada para el problema presentado. Además, esta brinda un resultado certero para poder agilizar el proceso de determinar el tratamiento adecuado para una enfermedad en específico.

Referencias bibliográficas

Dimeisons, N. (2014, 29 julio). *Listas Enlazadas*. Monografias.com. Recuperado 5 de septiembre de 2022, de https://www.monografias.com/trabajos101/las-istas-enlazadas/las-istas-enlazadas

Fuentes, J. (2021, 8 septiembre). *¿Qué son las estructuras de datos y por qué son tan útiles?* OpenWebinars.net. Recuperado 5 de septiembre de 2022, de https://openwebinars.net/blog/que-son-las-estructuras-de-datos-y-por-que-son-tan-utiles/

xml.etree.ElementTree — La API XML de ElementTree — documentación de Python - 3.10.6. (s. f.). Recuperado 5 de septiembre de 2022, de https://docs.python.org/es/3/library/xml.etree.elementtree.html