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.

Aplicación de simulación de tejidos

202000869 – Alan Misael Carpio Garcia

Resumen

Un laboratorio de investigación epidemiológica de Guatemala ha estado investigando la forma en que las enfermedades infectan las células del cuerpo humano y se expanden produciendo enfermedades graves e incluso la muerte. Luego de cierto tiempo se identificaron patrones que pueden determinar si una enfermedad producirá una enfermedad leve, grave y mortal si el caso de ser mortal el paciente morir la característica de la gravedad de la enfermedad es a través del uso de rejillas cuadradas con tejido del paciente en estas rejillas contienen células sanas y células contagiadas esta información de proporcionara por medio de un archivo XML

Palabras clave
Simulación, célula, paciente, periodos, rejilla

Abstract

An epidemiological research laboratory in Guatemala has been investigating how diseases infect cells in the human body and spread, causing serious illness and even death. After a certain time, patterns were identified that can determine if a disease will produce a mild, severe and fatal disease, if the case is fatal, the patient will die, the characteristic of the severity of the disease is through the use of square grids with the patient's tissue. These grids contain healthy cells and infected cells. This information will be provided through an xml file.

Keywords
Simulation, cell, patient, periods, grid

Introducción

Programa que simula la infección de células en el cuerpo humano y que se expanden produciendo enfermedades leves, graves y mortales. Por medio de comparaciones el programa es capaz de identificar esas enfermedades leves, graves y mortales, estas rejillas contienen una célula en cada celda y cada célula puedes estar saludable o contagiada y estas células tendrán un comportamiento que se evidenciara periódicamente por medio de periodos estos datos se darán por medio un archivo XML estos datos se extraerán en la aplicación y se visualizaran por medio de una tabla en la aplicación donde uno podrá seleccionar a ese paciente para poder hacer un análisis de ese paciente

Desarrollo del tema

Para poder resolver este proyecto se utilizo el lenguaje de Python se utilizó programación orientada a objetos para poder crear los pacientes que se mandaban por medio del archivo XML esos pacientes se pueden visualizar por medio de un tabla al momento de cargar el archivo en es objeto se mostrara el nombre, edad, tamaño de la rejilla, periodos a evaluar, las células contagiadas.

```
class paciente(object):
    def __init__(self,_Nombre,_Edad,_Periodo,_m,_lista) -> None:
        self.__Nombre = _Nombre
        self.__Edad = _Edad
        self.__Periodo = _Periodo
        self.__m = _m
        self.__lista = _lista
        self.__contador_periodo = 0
        self.__n = 0
        self.__n1 = 0
        self.__estado = ""
        self.__imprimir = 0
        pass
    def get_lista(self):···
    def get_m(self):···
    def insertar_lista(self,lista):···
    def get_nombre(self):···
    def get_edad(self):···
    def get_periodo(self):···
    def get_rejilla(self):···
    def aumentar_contador(self,number):···
    def get_contador(self):···
```

Clase paciente.

Las células contagiadas y sanas se insertan dentro de una estructura de datos esa estructura forma una matriz el tamaño varía según el tamaño de la rejilla que tenga el paciente a analizar

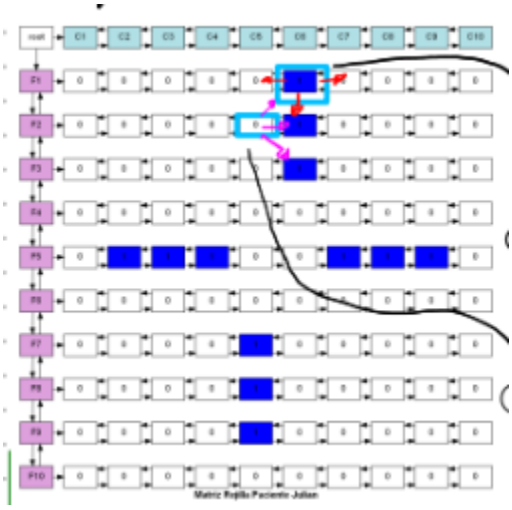


Imagen de cómo es casi la estructura de los nodos

Esta estructura tiene cuatro punteros en cual al momento de crear un nodo este estará conectado con el abajo el de arriba el de la izquierda y derecha la cual me permite poder comparar ese nodo en especifico con sus vecinos de una forma más eficiente.

Se utilizo una clase nodo y una clase lista doblemente enlazada en esa lista se insertan los datos dentro de esta estructura de nodos luego con una función compara los nodos con sus vecinos y súcibamente con los siguientes nodos hasta llegar con el ultimo nodo luego ese tejido comparado se almacena dentro de una lista dentro del objeto del paciente que se esta comparando luego ese tejido comparado se vuelve a comparar según los periodos que solicita el paciente

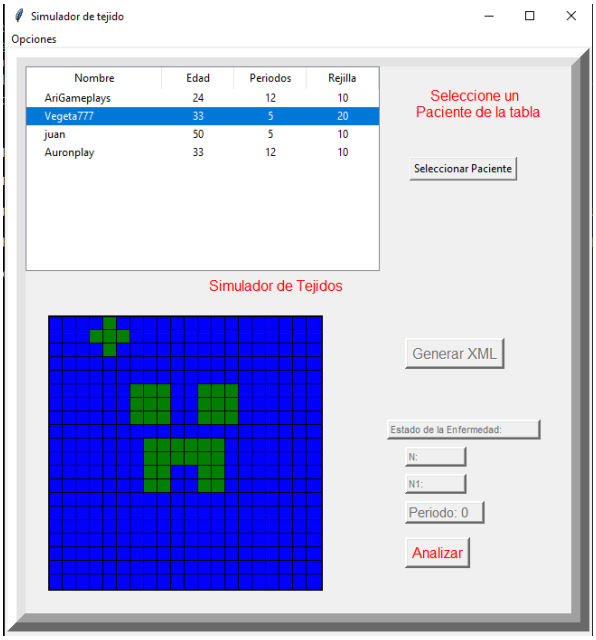
```python
from Paciente import paciente as p
class Nodo(object):
    def __init__(self,_dato) -> None:
        self.dato = _dato
        self.siguiente = None
        self.anterior = None
        self.superior = None
        self.inferior = None
        pass
class ListaEnlazada:

    def __init__(self,_sizeMatriz) -> None:
        self.primero = None
        self.final = None
        self.temp = None
        self.size = 0
        self.size2 = 0
        self.size3 = 0
        self.sizeMatriz = int(_sizeMatriz)
    def insertar(self,dato):···
    def sumador(self,com):···
    def comprabaciones_superiores(self,com):···
    def comprabaciones_inferiores(self,com):···
    def comprobacion_siguiente(self,com):···
    def comprobacion_anterior(self,com):···
    def comprobacion(self,m,lol):···
    def imprimir(self):···

```

Clase Nodo y lista Enlazada

Se utilizo canvas y tkinter para creación de la interfaz de la aplicación el canvas se utilizo para graficacion del tejido donde se visualizará en tiempo real como va infectándose las células o sanándose.
Se utilizo tkinter para la creación de la interfaz
También tiene la opción de generar un archivo XML donde estarán los datos del paciente o pacientes que se les realizo un análisis.



Interfaz y graficacion de la aplicación

Conclusiones

En la elaboración del proyecto se tubo la dificultad al momento de comparar ese nodo con sus vecinos ya que en internet hay muy poca o casi nada de información de como elaborar esa estructura de datos con este proyecto me ayudo a mejorar más mi lógica al momento de programar habían mas formas de comparar los nodos pero eran muy ineficientes y tardaba mas en comparar los datos este métodos es uno de los mas eficientes para poder comparar más rápido esos datos.

Referencias bibliográficas

https://www.geeksforgeeks.org/python-programming-language/?ref=shm

Extensión: de cuatro a siete páginas como máximo

Adicionalmente, se pueden agregar apéndices con modelos, tablas, etc. Que complementan el contenido del trabajo.