

---

## PROYECTO 1: ANÁLISIS DE ENFERMEDADES

---

202109715 – Pablo Josué Barahona Luncey

### Resumen

Un laboratorio de investigación epidemiológica en Guatemala ha estudiado cómo las enfermedades infectan las células del cuerpo humano, se propagan y causan enfermedades graves e incluso la muerte. Después de realizar los experimentos, los científicos identificaron patrones que podrían determinar si la enfermedad resultaría en una enfermedad leve o grave, o si el paciente moriría a causa de la enfermedad.

### Palabras clave

Enfermedades a través de células

### Abstract

*An epidemiological research laboratory in Guatemala has studied how diseases infect the cells of the human body, spread and cause serious illness and even death. After conducting the experiments, the scientists identified patterns that could determine whether the disease would result in mild or severe illness, or whether the patient would die from the disease.*

### Keywords

*Disease through cells*

## Introducción

La automatización es lo que se busca en los programas, debido a la necesidad de evaluar enfermedades, se realiza un programa el cual detecta a través de las células si el paciente infectado tendrá una enfermedad leve, mortal o grave. El programa se desarrolla en el lenguaje de programación Python y su objetivo principal es analizar patrones a través de rejillas infectadas.

## Desarrollo del tema

Este programa se desarrolló en el lenguaje de programación Python, el cual es un lenguaje de scripting independiente de plataforma y orientado a objetos, preparado para realizar cualquier tipo de programa, desde aplicaciones Windows a servidores de red o incluso, páginas web. Una de sus ventajas es que es un lenguaje interpretado, lo que significa que no necesita compilar el código fuente para poder ejecutarlo.

Para este proyecto se tiene como objetivo principal la utilización de clase lista simplemente enlazada y clase nodo. Este tipo de listas es una estructura de datos en la que cada elemento apunta al siguiente. De este modo, teniendo la referencia del principio de la lista podemos acceder a todos los elementos de la misma.

También se usaron listas nativas de Python y diferentes librerías, tales como numpy, la cual es una biblioteca para el lenguaje de programación Python que da soporte para crear vectores y matrices grandes

multidimensionales, junto con una gran colección de funciones matemáticas de alto nivel para operar con ellas, entre otros.

Así mismo, se tiene una clase main, la cual contiene la dirección del programa y el uso de los métodos de la clase lista.

El nodo únicamente almacena los datos recibidos, en otros lenguajes también se le conoce como constructor.

Para poder entender estos conceptos, se realizó un diagrama de clases, el cual puede encontrarse en la parte de Anexos como “Imagen 1”.

Para entender el programa desde un punto de vista técnico, iniciamos presentando el programa y el creador, en este caso soy yo (Pablo Josué Barahona Luncey) y las respectivas identificaciones.

Al inicio, se le pide al usuario ingresar una opción, de las cuáles son:

1. Ingresar datos
2. Salir

Si el usuario selecciona ingresar datos, se le abrirá una nueva ventana, el cuál es el explorador de archivos, en este explorador de archivos, se tiene por defecto que ingrese datos con extensión .xml, estos datos deben de tener una estructura como la siguiente:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <pacientes>
3    <paciente>
4      <datospersonales>
5        <nombre>Jaime</nombre>
6        <edad>24</edad>
7      </datospersonales>
8      <periodos>5</periodos>
9      <m>10</m>
10     <rejilla>
11       <celda f="1" c="1" />
12       <celda f="1" c="2" />
13       <celda f="2" c="1" />
14       <celda f="2" c="2" />
15       <celda f="6" c="6" />
16       <celda f="6" c="7" />
17       <celda f="7" c="6" />
18       <celda f="7" c="7" />
19     </rejilla>
20   </paciente>
21 </pacientes>
```

Este archivo xml debe de ser interpretado por el programa. Para entender el archivo, se tiene una lista de pacientes, los cuales tiene sus datos personales, luego el período de evaluación y el tamaño de la matriz a evaluar, se muestran en las celdas, las coordenadas de las células infectadas. Así se hará con todos los pacientes de la lista.

Posterior a entender el xml, el programa evalúa e interpreta estos datos. Se realizó a través de una librería llamada xml.dom o conocida también como minidom. Esta librería puede leer los archivos xml e interpretarlos según el gusto del programa.

Para empezar a usar ese archivo, se lee primero la lista de pacientes, luego de eso, con un ciclo for, se van leyendo los datos uno por uno y almacenando en nuestra clase nodo con la ayuda de un método de

nuestra clase lista. Luego de leer todo el archivo xml, pasamos a entender que se quiere hacer.

Luego de tener ya nuestros datos almacenados, se procede a usar los métodos de nuestra clase lista.

Primero usamos el método “mostrarpaciente(self)” este nos dará un resumen de los pacientes cargados en nuestro programa. Muy importante que los datos estén ingresados correctamente. A este método se le deben de llamar los datos de nuestra clase nodo.

Ya que mostramos todos los pacientes, procedemos a preguntarle al usuario que paciente desea evaluar, con esto debe de seleccionar el número correspondiente al paciente.

Acá usaremos el método pacienteseleccionado(self, numpaciente), este método toma como parámetro el dato ingresado por el usuario.

Ya que el usuario seleccionó el paciente, podremos desplegar todos los datos y detalles del mismo. El usuario podrá seleccionar que hacer con el usuario, ya que muestra la rejilla original sin ningún patrón modificado.

Para esto volveremos a preguntar al usuario que debe de hacer. Las opciones son:

1. Analizar muestra
2. Salir del programa

Si el paciente selecciona analizar muestra, podrá analizar la muestra e ir agregando cuantas muestras guste y esté en el parámetro permitido.

Para esto, se creó un algoritmo en un método llamado mostrarmatrizenferma(self), el siguiente algoritmo funciona de la siguiente manera:


1. Se evalúan a través de if, los vecinos de cada célula, si encuentra algún vecino infectado, se concatena en una variable.

2. Luego de concatenar las variables, se evalúa según los parámetros iniciales. Si una célula infectada tiene 2 o 3 células vecinas infectadas, esta seguirá con su mismo estado. Si una célula tiene exactamente 3 células vecinas contagiadas, entonces esta se contagiara y cambiará su estado de “sano” a “infectado”
3. Luego de ya tener su estado, se procede a efectuar los cambios y establecer esa matriz como patrón n y se evaluará el siguiente patrón sobre el patrón n.
4. El algoritmo podrá ser evaluado la cantidad de veces que esté indicado en el archivo xml, puede ser de 2 a 100000 veces, eso dependerá del evaluador.
5. Al usuario siempre se le pregunta si desea seguir analizando la muestra o salir del programa.
6. Los datos de matriz, celdas contagias y celdas sanas, son proporcionadas por el usuario.
7. Las matrices generadas se van guardando en una lista que posteriormente se utilizará.

Luego de utilizar el algoritmo, se procederá a evaluar la enfermedad. Para ello utilizaremos el método listaduplicada(self), este método funciona de la siguiente forma:

Se evalúa a través de la lista los resultados y se comparan entre ellos, si se repiten una vez, muchas veces o si no se repiten, de eso dependerá el diagnóstico. La enfermedad puede ser leve, grave o mortal.

Ya que se evaluó la enfermedad, se genera un archivo xml, este archivo xml debe de tener la siguiente estructura:



```
1 <pacientes>
2   <paciente>
3     <datospersonales>
4       <nombre>AriGameplays</nombre>
5       <edad>24</edad>
6     </datospersonales>
7     <periodos>12</periodos>
8     <resultado>grave</resultado>
9   </paciente>
10 </pacientes>
```

Este será el fin del programa.

## Conclusiones

1. EL proyecto ayuda a profundizar sobre la idea de las listas simples y como utilizar POO para problemas de la vida real.
2. EL utilizar lenguajes como Python, ayuda a profundizar sobre los temas de automatización y POO para que el usuario se sienta cómodo con su uso y se tenga una fácil interacción.
3. Los algoritmos son creados para poder evaluar y ejecutar los programas de una manera automática, su principal objetivo es reducir líneas de código y en este programa si lo pudo hacer ya que existían 56 diferentes combinaciones para poder evaluar si existían celdas vecinas contagiadas.

## Referencias bibliográficas

- Barahona, P. (5 de septiembre de 2022). *Lucidchart*.  
Obtenido de Lucidchart:  
[https://lucid.app/documents?canInterruptWithPayments=1#/templates?folder\\_id=home](https://lucid.app/documents?canInterruptWithPayments=1#/templates?folder_id=home)
- Desarrollo Web. (5 de septiembre de 2022).  
*desarrolloweb.com*. Obtenido de  
desarrolloweb.com:  
<https://desarrolloweb.com/articulos/1325.php>
- Desconocido. (5 de septiembre de 2022). *Listas  
enlazadas simples*. Obtenido de Listas  
enlazadas simples:  
[http://www.it.uc3m.es/java/2011-12/units/pilas-colas/guides/2/guide\\_es\\_solution.html#:~:text=Una%20lista%20enlazada%20simple%20es,los%20elementos%20de%20la%20misma](http://www.it.uc3m.es/java/2011-12/units/pilas-colas/guides/2/guide_es_solution.html#:~:text=Una%20lista%20enlazada%20simple%20es,los%20elementos%20de%20la%20misma).
- Stefunko, M. (2018). *Acción humana*. Guatemala:  
The Scholar's edition.

## Anexos

### Clase UML

PABLO JOSUE BARAHONA LUNCEY | September 6, 2022

