Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Introducción a la Programación y Computación 2

Inga. Claudia Liceth Rojas Morales
Ing. Marlon Antonio Pérez Türk
Ing. Byron Rodolfo Zepeta Arevalo
Ing. Jose Manuel Ruiz Juarez
Ing. Edwin Estuardo Zapeta Gómez

Tutores de curso:
Alexander Raymundo Ixvalan Pacheco
Jackeline Alexandra Benitez Benitez
Viany Paola Juárez Hernández
Oscar Roberto Velásquez León
José Carlos Estrada Garcia

PROYECTO 1

OBJETIVO GENERAL

Se busca que el estudiante sea capaz de dar una solución al problema que se le plantea, mediante la lógica y conocimientos que se le han impartido durante la clase y que han sido aplicados en el laboratorio.

OBJETIVOS ESPECÍFICOS

- Que el estudiante sea capaz de aplicar abstracción a un problema dado.
- Implementar una solución utilizando el lenguaje de programación Python.
- Utilizar estructuras de programación secuenciales, cíclicas y condicionales.
- Que el estudiante aprenda a generar reportes con la herramienta Graphviz.
- Que el estudiante sea capaz de manipular archivos XML.
- Que el estudiante utilice los conceptos de TDA y aplicarlos a memoria dinámica.

A. Descripción del problema

El 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.

Los científicos, luego de hacer experimentos han logrado identificar patrones que pueden determinar si una enfermedad producirá una enfermedad leve, una enfermedad grave, o si el paciente morirá a causa de la enfermedad. La forma en que identifican los patrones que pueden llegar a determinar las características de la gravedad de la enfermedad es a través del uso de rejillas cuadradas con tejido del paciente, estas rejillas contienen una célula en cada celda y cada célula puede estar saludable o contagiada. Los científicos han identificado que las células contagiadas tienen un comportamiento que se evidencia periódicamente, este comportamiento se define de la siguiente forma:

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

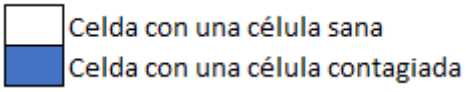
Los científicos, estudian las enfermedades tomando rejillas de células que contienen un patrón inicial de células sanas y células contagiadas, y luego de "N" períodos, evalúan los datos buscando uno de los siguientes casos:

a. Que el patrón inicial se repita siempre después de "N" períodos, 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 ésta será incurable.
b. Que algún patrón, distinto al patrón inicial, se repita luego de "N" períodos cada "N_1" períodos, 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 ésta será incurable, en caso de que "N_1" es mayor que 1, la enfermedad será grave.

Las rejillas son estructuras cuadradas de "M" por "M" celdas donde "M" toma valores en múltiplos de 10 a partir de 10.

B. Ejemplos:

Para los ejemplos, se considerará la siguiente notación:



Ejemplo No. 1

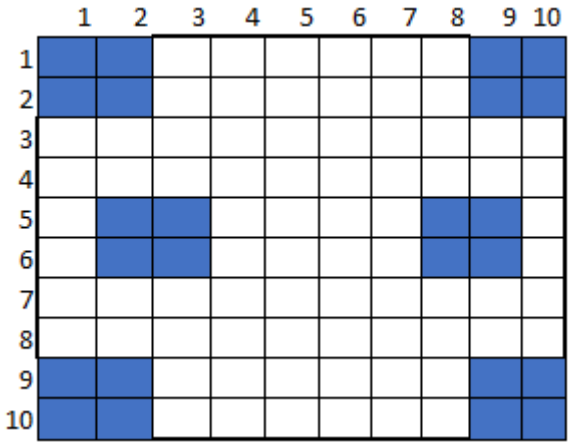


Fig. No. 1 – Rejilla inicial

En la Figura No. 1 se muestra la rejilla inicial con el tejido de un paciente, como se puede observar, luego de un período, aplicando las reglas de la Sección A, incisos 1 y 2, la rejilla continuará igual ya que cada celda que contiene una célula contagiada tiene exactamente 3 vecinos contagiados, por lo que en el siguiente período continuarán contagiadas; mientras que ninguna celda con una célula sana tiene 3 celdas vecinas con células contagiadas, por lo que continuarán sanas. Esto significa que el patrón se repite cada 1 períodos, es decir, N = 1, por lo tanto, el paciente morirá.

Ejemplo No. 2

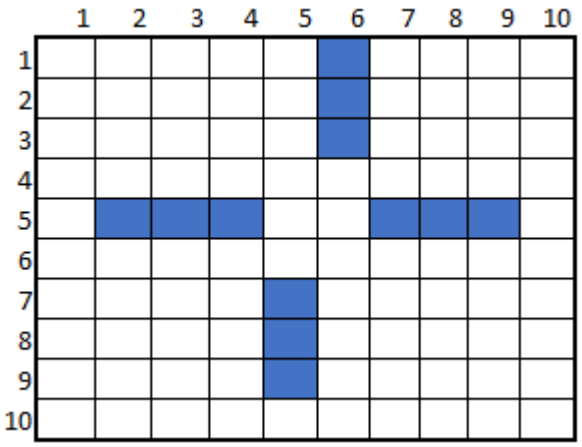


Fig. No. 2 – Rejilla inicial

En la figura No. 2, se muestra una rejilla inicial con el tejido de un segundo paciente, luego de un período, se aplicarán las reglas de la sección A, incisos 1 y 2, entonces, la rejilla se verá de la siguiente forma:

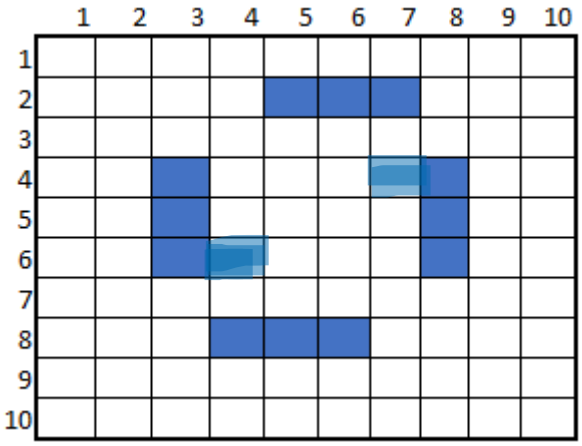


Fig. No. 3 – Rejilla luego de 1 período

La figura No. 3, muestra la rejilla con las células de los tejidos del 2do paciente luego de 1 período de espera. Si esperamos un 2do. Período, entonces, volveremos a la rejilla de la Figura No. 2, es decir, al patrón inicial, en este caso, N = 2, ya que luego de 2 períodos encontramos el patrón inicial y por lo tanto, se espera que este paciente tenga una enfermedad grave con un período N = 2.

Debe tomar en cuenta, que existe un caso especial, este se da cuando el patrón inicial no se repite, pero produce un patrón que eventualmente si se repite, en este caso, se tiene un patrón que se produce en el período N, y que luego se repite cada N_1 períodos. En este caso si N_1 es igual a 1 la enfermedad será mortal, de lo contrario será grave.

C. Programa a desarrollar

El laboratorio de investigación epidemiológica de Guatemala le ha contratado para construir una aplicación de software que permita a sus científicos ingresar la información de las rejillas en su sistema y que éste sea capaz de determinar si la enfermedad será grave o mortal para un paciente dado.

Límites

Tamaño máximo de la rejilla: 10,000 filas y 10,000 columnas

Cantidad de períodos máxima a evaluar: 10,000 períodos

Ingreso de datos iniciales

Los datos iniciales serán cargados en un archivo XML, este archivo contendrá los nombres de pacientes, el tamaño de la rejilla y la representación de la rejilla con el patrón inicial identificando en cada celda las células sanas (con un valor 0) y las células contagiadas (con un valor 1).

Funcionalidad de la aplicación a desarrollar

Se espera que la aplicación permita elegir un paciente y ejecutar los períodos uno a uno para ver gráficamente la forma en que se desarrollará la enfermedad en el tejido proporcionado en la rejilla inicialmente, además, deberá identificar el momento en que el patrón inicial se repita, indicando el período en que ocurre, o bien, si un patrón generado posteriormente se repite, deberá indicar el período en que apareció el patrón y en cuántos períodos empezó a repetirse. El usuario también dispondrá de una opción para ejecutar automáticamente todos los períodos necesarios hasta que se identifique si la enfermedad será mortal o grave, o bien se llegue al límite de períodos de la aplicación. En todo momento, el usuario deberá observar la cantidad de células sanas, la cantidad de células contagiadas y el período en que se encuentra la simulación.

El programa también proveerá una opción para generar una salida a un archivo XML. Este archivo contendrá la información de cada paciente, si desarrollará una versión grave, mortal o leve de la enfermedad y el período en que se obtendrá el patrón inicial, o bien, el período en que se tendrá un patrón que se repetirá, así como el período secundario necesario para repetir dicho patrón.

Ejemplo XML de entrada

```
<?xml version="1.0" encoding="UTF-8"?>
<pacientes>
  <paciente>
    <datospersonales>
      <nombre>$nombre</nombre>
      <edad>$edad</edad>
    </datospersonales>
    <periodos>5</periodos>
    <m>10</m>
    <rejilla>
      <celda f="1" c="1" />
      <celda f="1" c="2" />
      <celda f="2" c="1" />
      <celda f="2" c="2" />
      <celda f="6" c="6" />
      <celda f="6" c="7" />
```

```
        <celda f="7" c="6" />
        <celda f="7" c="7" />
      </rejilla>
    </paciente>
    <paciente>....</paciente>
</pacientes>
```

Ejemplo XML de salida

```
 <?xml version="1.0" encoding="UTF-8"?>
<pacientes>
  <paciente>
    <datospersonales>
      <nombre>$nombre</nombre>
      <edad>$edad</edad>
    </datospersonales>
    <periodos>5</periodos>
    <m>10</m>
    <resultado>leve</resultado>
  </paciente>
  <paciente>
    <datospersonales>
      <nombre>$nombre2</nombre>
      <edad>$edad2</edad>
    </datospersonales>
    <periodos>5</periodos>
    <m>10</m>
    <resultado>grave</resultado>
    <n>3</n>
  </paciente>
  <paciente>
    <datospersonales>
      <nombre>$nombre3</nombre>
      <edad>$edad3</edad>
    </datospersonales>
    <periodos>10</periodos>
    <m>10</m>
    <resultado>mortal</resultado>
    <n>1</n>
  </paciente>
  <paciente>
    <datospersonales>
      <nombre>$nombre3</nombre>
      <edad>$edad3</edad>
    </datospersonales>
    <periodos>1000</periodos>
    <m>10</m>
    <resultado>grave</resultado>
    <n>50</n>
```

```
    <n1>15</n1>
  </paciente>
  <paciente>
    <datospersonales>
      <nombre>$nombre3</nombre>
      <edad>$edad3</edad>
    </datospersonales>
    <periodos>1000</periodos>
    <m>10</m>
    <resultado>mortal</resultado>
    <n>99</n>
    <n1>1</n1>
  </paciente>
  <paciente>....</paciente>
</pacientes>
```

Nota:
N = Numero de periodo en donde se encuentra el nuevo patrón a repetir
N1= Numero de periodo en el que se repite el nuevo patrón encontrado

D. Consideraciones

Se deberán implementar listas enlazadas para resolver el problema, creadas por el estudiante, creando una clase Nodo y una o varias clases lista enlazada, de tal manera que al recorrer la lista se puedan realizar las operaciones con las muestras de tejido que contienen las rejillas.

El cómo se utilicen las estructuras anteriormente descritas para guardar los datos del archivo de entrada queda a discreción del alumno. Tomar en cuenta que al ingresar el archivo de entrada todas las rejillas con muestras de tejido de los pacientes dentro del archivo XML deben de ser cargados a las listas para luego tener la habilidad de simular los períodos de alguna de las muestras ingresadas, o bien, generar el archivo de salida.

Debe utilizarse versionamiento para el desarrollo del proyecto. Se utilizará la plataforma **Github** en la cual se debe crear un repositorio en el que se gestionará el proyecto. Se deben realizar 4 releases o versiones del proyecto (se recomienda realizar una por semana del tiempo disponible). Se deberá agregar a sus respectivos auxiliares como colaboradores del repositorio. El último release será el release final y se deberá de realizar antes de entregar el proyecto en la fecha estipulada.

DOCUMENTACIÓN

Para que el proyecto sea calificado, el estudiante deberá entregar la documentación utilizando el formato de ensayo definido para el curso. En el caso del proyecto, el ensayo puede tener un mínimo de 4 y un máximo de 7 páginas de contenido, este máximo no incluye los apéndices o anexos donde se pueden mostrar modelos y diseños utilizados para construir la solución. Es obligatorio incluir el diagrama de clases que modela la solución de software presentada por el estudiante.

RESTRICCIONES

- Solo se permitirá la utilización de los IDEs discutidos en el laboratorio.
- Uso obligatorio de programación orientada a objetos (POO) desarrollada por completo por el estudiante. De no cumplir con la restricción, no se tendrá derecho a calificación.
- El nombre del repositorio debe de ser **IPC2_Proyecto1_#Carnet.**
- El estudiante debe entregar la documentación solicitada para poder optar a la calificación.
- Los archivos de entrada no podrán modificarse.
- Los archivos de salida deben llevar la estructura mostrada en el enunciado obligatoriamente.
- Deben existir 4 releases uno por cada semana, de esta manera se corrobora el avance continuo del proyecto.
- Se calificará de los cambios realizados en el cuarto release. Los cambios realizados después de ese release no se tomarán en cuenta.
- Cualquier caso de copia parcial o total tendrá una nota de 0 y será reportada a escuela.
- Para dudas concernientes al proyecto se utilizarán los foros en UEDI de manera que todos los estudiantes puedan ver las preguntas y las posteriores respuestas.
- **NO HABRÁ PRÓRROGA.**

ENTREGA

- La entrega será el **05 de septiembre** a más tardar a las 11:59 pm.
- La entrega será por medio de la UEDI.
- La documentación debe estar subida en el repositorio en una carpeta separada.
- Para entregar el proyecto en UEDI se deberá subir un archivo de texto con el link del repositorio.