Diseño y Programación Orientados a Objetos
1er. Semestre 2021

Tarea 3: Simulador Gráfico en C++ de la Evolución de una Pandemia

Lea detenidamente la tarea. **Si algo no lo entiende, consulte. Si es preciso, se incorporarán aclaraciones en otro color.**

1.- Objetivos

- Ejercitar la compilación y ejecución de programas en lenguaje C++ desde una consola de comandos.
- Ejercitar la configuración de un ambiente de trabajo para desarrollar aplicaciones en lenguaje C++, se pide trabajar con un IDE.
- Reconocer clases y relaciones entre ellas en lenguaje C++.
- Diseñar y programación de clases en C++.
- Ejercitar la entrada de datos en C++.
- Ejercitar la preparación y entrega de resultados de software (creación de makefiles, readme, documentación, manejo de repositorio GIT).
- Desarrollar aplicaciones gráficas usando
- Uso de metodología de desarrollo "iterativa" e "incremental".

1.- Descripción General

Esta tarea busca programar en C++ algunas funcionalidades de las tareas 1 y 2. La lógica de la simulación es esencialmente la misma de las tareas previas. Por completitud, se detalla nuevamente.

Se considera un número dado de “N” individuos dispersos que se mueven aleatoriamente y de manera confinada en una región plana rectangular que llamaremos **comuna**. Los N individuos se desglosan en: “S” individuos susceptibles a infectarse e “I” individuos ya infectados. En este ejercicio, los individuos infectados pasan a estado “R”, de recuperado, luego de I_time segundos. Los parámetros N, I, e I_time son ingresados vía un archivo y, durante la ejecución, pueden ser cambiados a través de la interfaz antes de dar inicio a una nueva simulación. Notar que éstos son los únicos parámetros modificables a través de la interfaz gráfica. Individuos recuperados no son susceptibles de ser infectados nuevamente. Durante la simulación, la interfaz gráfica para cambiar sus parámetros permanece deshabilitada.

La comuna es rectangular de dimensiones (width, length), ambos parámetros ingresados desde el archivo de entrada.
El movimiento aleatorio de los individuos es simulado con movimientos de rapidez constante seleccionada entre 0.9*Speed y 1.1*Speed (así se simula un individuo de caminar calmado y otros ágiles -más jóvenes-). La dirección θ de la velocidad varía en cada Δt a un valor aleatorio tomado de entre $\theta-\Delta\theta$ y $\theta+\Delta\theta$. “Speed”, Δt y $\Delta\theta$ son parámetros de la simulación ingresados vía archivo. θ inicial es un valor aleatorio para cada individuo. Cuando un individuo tiende a salir de la comuna, éste cambia su dirección como si estuvieran rebotando en el borde de ésta.

Cuando un individuo susceptible de infectarse se acerca a una distancia inferior a d [m] de uno infectado, existe una probabilidad P de infectarse por cada segundo en contacto cercano. Esta probabilidad de contagio depende de si ninguno (P=p0), sólo uno (P=p1) o ambos (P=p2) usan mascarilla. p0, p1 y p2 son parámetros de la simulación ingresados por archivo. El parámetro M indica la fracción de individuos de cada tipo que usan mascarilla. Por ejemplo, si M=0,2, significa que 1/5 de los individuos infectados usan mascarilla y 1/5 de los susceptibles de infectarse también la usan. M es otro parámetro de la simulación ingresado por archivo.

Finalmente, en la comuna se activan NumVac vacunatorios a los VacTime segundos de iniciada la simulación. Los vacunatorios son representados como zonas cuadradas fijas, de lado VacSize, y ubicados aleatoriamente en la comuna. Cuando un individuo susceptible de infectarse ingresa al área

de un vacunatorio, éste es vacunado, pasa a estado V y deja de ser susceptible de infectarse. Individuos infectados o recuperados no son vacunados en esta simulación.

El formato para el archivo de entrada -que es pasado como un argumento de la ejecución del programa- es rígido:
<N> <I> <I_time [s] >
<width [m]> <length [m] >
<Speed [m/s] > <Δt [s] > <$\Delta\theta$ [rad] >
<d [m] > <M> <p0> <p1> <p2>
<NumVac> <VacSize [m]> <VacTime [s] >

Por ejemplo este archivo puede tener valores como (es posible que haya valores más interesantes de considerar):
1 0 5
1000 100
1.4 0.2 0.4
2 0.5 0.3 0.2 0.1
2 10 100

Como salida, interesa conocer la evolución de las variables S, I, R y V en el tiempo. Para esto usted generará un gráfico, ver Figura 1, en pantalla de áreas apiladas donde la primera área es el número de individuos vacunados, luego el número de infectados, recuperados y finalmente los susceptibles de infectarse, todos como función del paso del tiempo. Este gráfico se debe actualizar regularmente conforme la simulación progresa.

Además de la salida gráfica de la Figura 1, interesa conocer la evolución de las variables S, I, R y V en el tiempo. Para esto usted enviará la salida a pantalla con el siguiente formato por cada línea (el tiempo avanza en unidades cercanas a un segundo). La primera línea es de encabezado para describir cada columna, luego vienen líneas según la duración de la simulación. Cada vez que se corra una nueva simulación se vuelve a mostrar por pantalla la primera línea.
Time, Vac, Inf, Rec, Sus
<t>, <V>, <I>, <R>, <S>

Interfaz
La interfaz Qt debe tener una apariencia como la de la Figura 1.

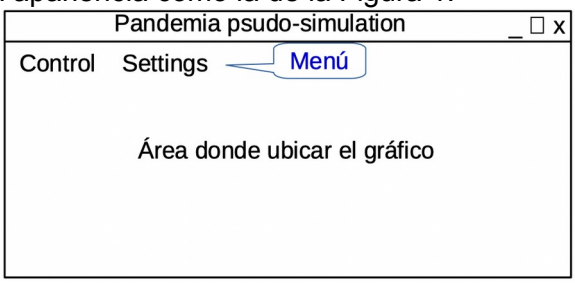

Figura 1: Interfaz principal

La opción del menú “Control” muestra dos opciones: Start y Stop. El programa parte con la simulación detenida y se activa al seleccionar Start. Cada vez que se presiona Start, la simulación se reinicia desde cero con los nuevos valores ingresados. Es decir el gráfico parte en t=0. La simulación correrá en “tiempo real”: un segundo de simulación será un segundo de tiempo real. La aplicación termina al cerrar su ventana principal.

No se recomienda refrescar la interfaz cada Δt. Programe la frecuencia de actualización de la interfaz a algún valor entre 3 y 5 veces por segundo.

La opción Settings sólo está activa en estado de simulación detenida. Ésta muestra una segunda ventana que sólo permite cambiar los parámetros N, I, e I_time de la simulación. Esto se pide así para no complicar la tarea.
Usted puede elegir el recurso gráfico cambiar los parámetros. El valor inicial para cada parámetro será el definido hasta ese momento en la simulación, partiendo por los valores tomados desde el archivo de entrada.

La ejecución de su tarea en la línea de comando sería:
$ pandemia *"archivo de entrada.txt"*

2.- Desarrollo en Etapas
Para llegar al resultado final de esta tarea usted debe aplicar una metodología de desarrollo de software de tipo "Iterativa e Incremental". Usted y su equipo irán desarrollando etapas donde los requerimientos son abordados gradualmente. En cada etapa usted y su equipo obtendrá una solución que funciona para un subconjunto de los requerimientos finales o bien es un avance hacia ellos. **Su equipo deberá entregar una solución para cada una de las etapas,** aún cuando la última integre las primeras. **El readme y archivo de documentación deben ser preparados solo para la última etapa. Prepare y entregue un makefile para cada una de las etapas.** Esto tiene por finalidad, practicar en la metodología iterativa e incremental.

2.1.- Primera Etapa: Un individuo se mueven aleatoriamente
Es esta primera etapa no hay vacunatorios, nadie usa mascarilla y nadie se contagia. El objetivo de esta etapa es verificar el funcionamiento de algunas clases y la lógica de desplazamiento de un peatón cuando que se mueve aleatoriamente en la región comuna.
Para verificar el correcto funcionamiento de esta etapa, su programa mostrará por pantalla:
<t>, <x>, <y> /* *x, y son las coordenadas del individuo en el tiempo t.* */

El peatón se mueve hasta detener su ejecución a través de su IDE.

Use y/o complete las clases Comuna, Individuo, Simulador y Stage1 que serán provistos próximamente aquí. Usted no está obligado(a) a usar estos códigos. Están para su conveniencia y en caso que le resulten útiles. Otras formas de estructurar el resultado para cada etapa también son válidos.

2.2.- Segunda Etapa: Individuos se mueven aleatoriamente y pueden contagiarse
Esta etapa es similar a la previa (sin vacuna, sin mascarilla), excepto que se crean N individuos los cuales pueden ser almacenados en un "vector <Pedestrian>". Al iniciar el programa se crean y ubican I individuos infectados y (N-I) individuos susceptibles de infectarse.
Como salida se espera un archivo de salida similar al indicado en la descripción general de la tarea, excepto que no hay columna V. En esta etapa no hay interfaz gráfica y el programa termina a través de al IDE.

2.3.- Tercera Etapa: Individuos se mueven aleatoriamente y algunos usan mascarillas
Esta etapa modifica la forma como los individuos pueden contagiarse dependiendo del uso de mascarilla. Haga los cambios en la lógica del programa de la etapa previa para reflejar este comportamiento.
Esta etapa debe generar una interfaz gráfica como la Figura 1. En esta etapa no se incorpora aún el menú Settings. La interfaz ofrece la opción de iniciar y detener la simulación, pero no podrá cambiar los parámetros. La opción Stop equivale aquí a pausar la simulación. El objetivo principal de esta etapa es desarrollar la interfaz gráfica y mostrar el gráfico resultante de la evolución de la pandemia sin vacunación.

2.4.- Cuarta Etapa: Individuos se mueven aleatoriamente, algunos usan mascarillas y se inicia la vacunación.
En esta última etapa se espera alcanzar la funcionalidad descrita en la sección 1.

2.5.- Extra crédito: Uso de → y ← acelerar y ralentizar la simulación
Esta parte es voluntaria, su desarrollo otorga 5 puntos adicionales (la nota final se satura en 100).
Para acelerar la evolución de la simulación, durante la simulación, el usuario puede presionar la tecla → , la cual multiplicará el avance del tiempo por 2. Al presionar la techa ← el avance del tiempo se reduce a la mitad.

3.- Elementos a considerar en su documentación
Entregue todo lo indicado en “Normas de Entrega de Tareas”, excepto makefile.
Para su entrega deje en GitLab los archivos de su proyecto Qt. Se espera que usted omita archivos no esenciales. Para eso haga uso de archivo de configuración “.gitignore”.
En su archivo de documentación (pdf o html) incorpore el diagrama de clases de la aplicación (etapa 4).

4.- Sobre la entrega de la tarea
El formato de la entrega seguirá la siguiente norma:

- La sección de entrega de la tarea en AULA será exclusivamente para agregar el link al repositorio Git donde esté alojada la solución de la tarea.
- Esta entrega mencionada anteriormente sigue estando regida por los plazos establecido en la tarea.
- No olvidar que la tarea debe ser subida al repositorio Git respetando estos mismos plazos.
- Cada grupo debe incorporar a su ayudante con el rol “reporter” a su proyecto Git
- La no entrega a tiempo, implica un descuento diario.