Práctica 6. Mallas regulares

Sesiones de prácticas: 2 (entrega tardía 23-dic, defensa en Enero)

Objetivos

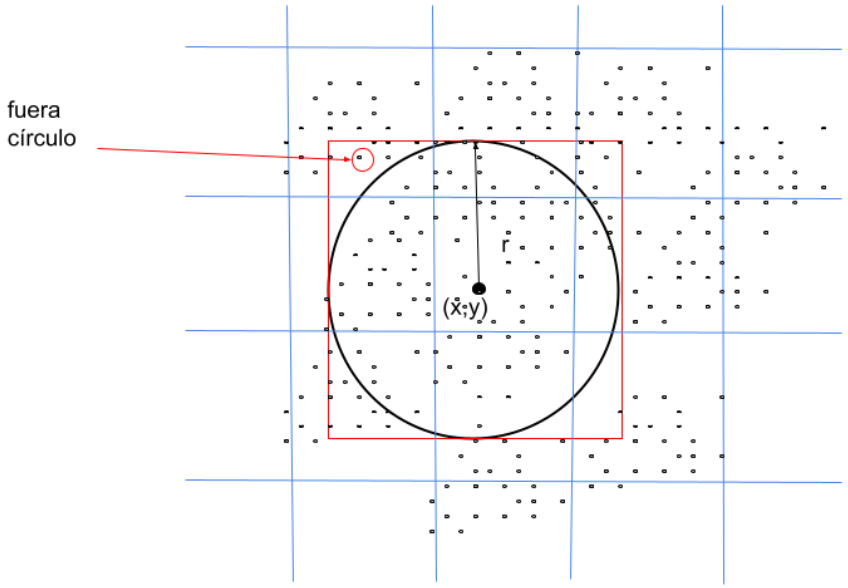
Utilizar mallas regulares para minimizar el tiempo de búsqueda del más cercano.

Descripción de la EEDD

La aplicación desarrollada hasta ahora ha usado las estructuras de datos más adecuadas en cada proceso, excepto en la búsqueda del centro de vacunación más cercano que se ha realizado mediante una búsqueda lineal. Para solucionar este problema se va a utilizar una malla regular (Lección 17-18).

```
  template <class T>
   class MallaRegular {
            …
   public:
      MallaRegular(float aXMin, float aYMin, float aXMax, float aYMax, int
nDiv);
….
      vector<T*> buscarRadio(float xcentro, float ycentro, float radio);
      unsigned maxElementosPorCelda();
      float promedioElementosPorCelda();
   }
```

Esta clase tiene la funcionalidad de la Malla Regular dada en la lección 17-18, pero añadiendo las funciones definidas arriba. La función buscarRadio() devuelve todos los usuarios dentro de un *radio* centrado en un punto con coordenadas *(xcentro,ycentro)*. Podemos asumir que todo tipo T con el que se instancie la clase tiene los métodos getX() y getY() implementados, que en nuestro caso son respectivamente la latitud y la longitud en coordenadas UTM. Para realizar esta operación, se considera el cuadrado de búsqueda que engloba al círculo y luego se calcula la distancia al centro. Se deben considerar sólo los puntos dentro del círculo, como se indica en el dibujo.



Descripción de la práctica:

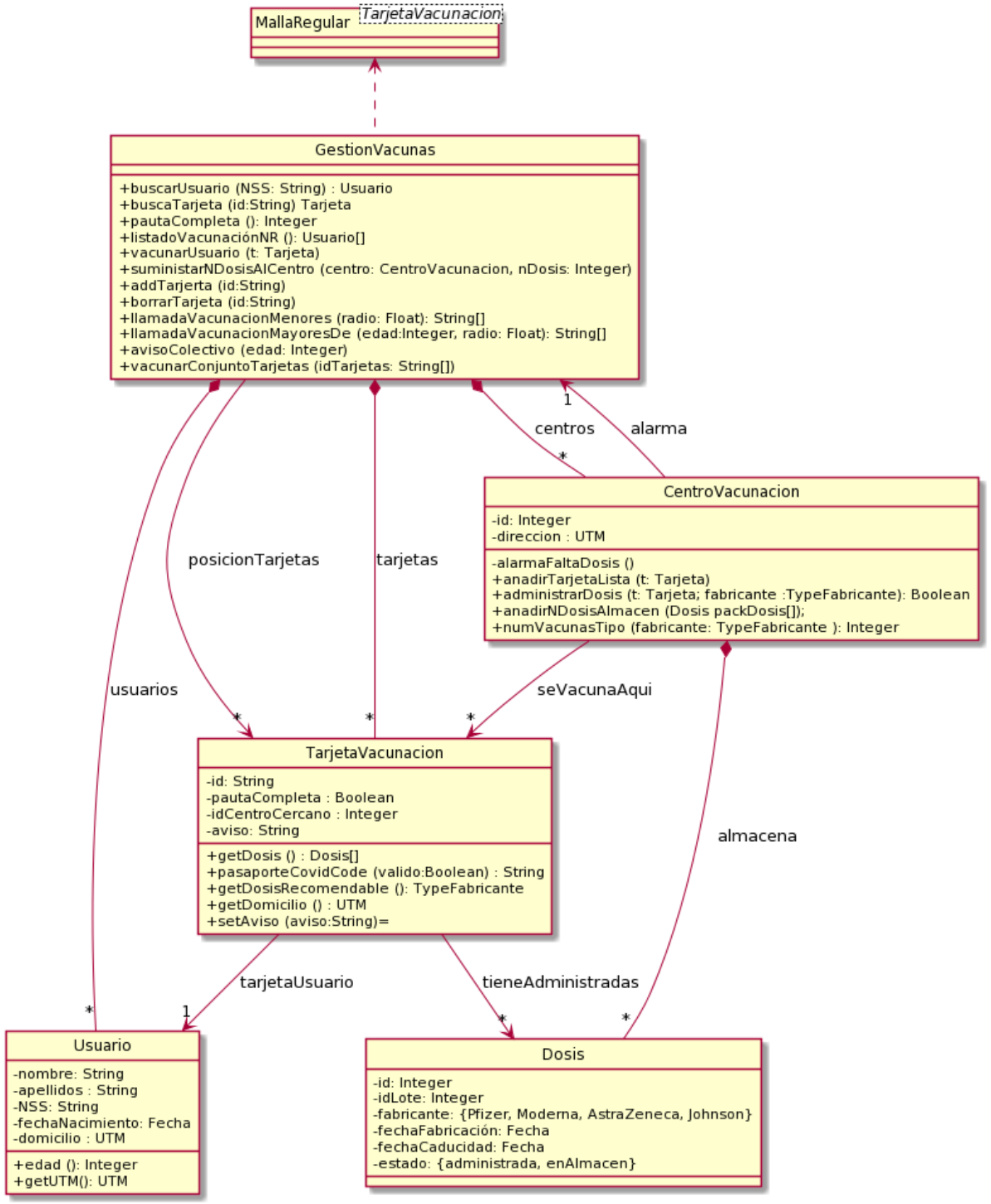
La malla se crea a partir de las tarjetas asociadas a los usuarios, siendo la posición UTM de éste la que se utiliza como coordenadas. Se consideran (aXmin,aYmin) las coordenadas más pequeñas del fichero y (aXmax, aYmax) las coordenadas máximas del recuadro, que se pueden obtener mientras se va rellenando el conjunto de usuarios. El número de casillas en la malla vendrá determinado por el promedio de usuarios por casilla, debiendo estar entre 15-20 usuarios/tarjetas. Calcular también el número máximo de usuarios en una casilla.

En el siguiente esquema UML se representa la nueva funcionalidad. En vista de las nuevas directrices de la Agencia Europea del Medicamento, los gestores de vacunas van a hacer una llamada a la vacunación a los siguientes colectivos: (A) menores de entre 5 y 11 años y a (B) los mayores de 60 para que de nuevo vayan a vacunarse. Para ello el gestor de vacunas añade dos nuevas funciones llamadaVacunacion…(), cada una de ellas dirigidas a los colectivos anteriores. Estas funciones lo que hacen es:

- Para cada uno de los centros
    - Para todas las tarjetas cuyos usuarios pertenezcan al colectivo (A o B) y que estén en un radio de XXX metros del centro
        - Añadir un aviso a la Tarjeta::aviso para que le aparezca al usuario el mensaje correspondiente y siempre y cuando no haya sido llamado por

otro centro (string no nulo). Ejemplo de mensaje: “<<idCentro>>:Acuda al centro más cercano para vacunarse”.
        - En caso de generar el aviso, asociar también la tarjeta de vacunación del usuario al centro de vacunación (relación *seVacunaAquí)*.
        - Si una persona tiene un aviso, no se cambia ni se añade al centro.
        - Cuando una persona es vacunada, se elimina su aviso de la tarjeta.
    - Para vacunar a los usuarios llamados se usará la función: *GestionVacunas::vacunarConjuntoTarjetas()*.

La función *avisoColectivo()* es para parejas y se explica abajo.



Programa de prueba:

Crear un programa de prueba con las siguientes indicaciones:

- Implementar la misma funcionalidad de la Práctica 5.

- Implementar la malla regular como un template según la Lección 17-18 añadiendo la función *buscarRadio()* descrita anteriormente.
- Realizar vacunación masiva con la primera dosis a todos los usuarios mayores de 12 años menos a los que tenga NSS acabado en 5 o 7.
- Vacunar de nuevo a todos los usuarios mayores de 25 años.
- Comprobar cuántos usuarios tienen pasaporte covid (pauta completa) hasta el momento con las condiciones actuales: dos dosis en mayores de 12 años.
- Cambiamos las nuevas condiciones de pasaporte covid: ahora deben ser todos los mayores de 5 años con 2 dosis y los mayores de 60 con 3 dosis y comprobamos cuantas personas obtendrían el pasaporte covid ahora sin nuevas vacunaciones. Hacer nuevas versiones de las funciones *pautaCompleta()* y *pasaporteCovid()*
- Hacer la llamada a la vacunación en todos los centros y para todas las tarjetas de usuarios de entre 5 y 11 años en un radio de 0.35 grados.
- De todos los llamados, vacunar automáticamente a todos los que tengan al menos un 5 en su Id de tarjeta.
- Llamad a la vacunación a los mayores de 60 en un radio de 0.5 grados de los centros de vacunación.
- De todos los llamados, vacunar automáticamente al 90% sin importar el orden.
- Comprobad de nuevo el número de personas con pauta completa considerando que relajamos la condición para los niños de entre 5 a 11 años, bastando con una sola dosis. Hacer por tanto nuevas versiones de las funciones asociadas

Para parejas:

Ha habido una fiesta donde acudieron todos los usuarios de 18 años y ha habido contagio masivo. Avisad a todos estos usuarios y a sus contactos cercanos en un radio de 0.075 grados. Implementar dicha funcionalidad en la función avisoColectivo(float rango), devolviendo todos los NSS de los usuarios alertados, en este caso el centro usado será el guardado en la tarjeta com más cercano.

VOLUNTARIO: Visualización 2D con la clase Img

Para poder visualizar el resultado de forma gráfica, se adjunta a esta práctica la clase **Img**. Esta clase es en realidad una matriz de píxeles, creada tomando como parámetro el tamaño de un recuadro en número de píxeles en (tamaFilas,tamaColumnas). Esta clase es capaz de dibujarse como imagen de modo que cada consulta generará un fichero imagen resultado con *Img::guardar()*. Ejecutar la función *main()* que aparece junto al código para comprobar el funcionamiento y visualizar la imagen resultante.

Para que esta clase sea útil en nuestro caso, hacemos coincidir las esquinas de la imagen con las del cuadro de trabajo donde se incluyen todas las coordenadas de los usuarios. Consideramos pues que la esquina inferior izquierda de nuestros datos es: (longitud, latitud) = (-9,99443, 35,86688) y la esquina

superior derecha es: (longitud, latitud) = (3,98926, 43,272616). En el ejemplo de prueba se pinta un recuadro y se pinta también un pixel azul.

Utilizar esta clase para dibujar con distinto color los usuarios cercanos a cada centro, en una imagen de 600*600.

Estilo y requerimientos del código:

1. El código debe ser claro, tener un estilo definido y estar perfectamente indentado, para ello se pueden seguir algunos de los estilos preestablecidos para el lenguaje C++ (http://geosoft.no/development/cppstyle.html).

2. Deben comprobarse todas los posibles errores y situaciones de riesgo que puedan ocurrir (desbordamientos de memoria, parámetros con valores no válidos, etc.) y lanzar las excepciones correspondientes, siempre que tenga sentido. Leer el tutorial de excepciones disponible en el repositorio de la asignatura en docencia virtual.

3. Se valorará positivamente la calidad general del código: claridad, estilo, ausencia de redundancias, etc.