Tema 2: Abstracción. Relación de problemas

1. Definir el T.D.A Servidor de red. Un servidor es un punto de red que se encuentra identificado por una dirección *ip.* Una dirección ip viene definida por cuatro dígitos que pueden tener valores que van desde 0 a 255. Se pide:
◦ Dar la especificación del tipo Servidor. Además establecer las operaciones que manejan al T.D.A
◦ Definir al menos dos *tipo rep.*
◦ Escoger uno de los *tipo rep* y para este establecer la función de abstracción e invariante de representación.

T.D.A Servidor:

- Especificación: representa un servidor de red con una dirección ip determinada compuesta por cuatro dígitos.
- Operaciones:
  - Servidor(): crea un servidor inicializado a 0.
  - Servidor(n): crea un servidor con la dirección n.
  - Servidor(otroServidor): crea un servidor a partir de otro ya existente.
  - GetIP(): devuelve la dirección ip del servidor.
  - setIP(m): asigna la dirección m al servidor.

- Tipo rep:

a) class Servidor{
    private:
        int digits[4];

b) class Servidor{
    private:
        int digitos;//los concatena en un unico entero

- Función de abstracción para el tipo rep a:
  x es un objeto de tipo rep a. $f_a(x) = x[0].x[1].x[2].x[3];$

- Invariante de representación a: sea $0 \leq i \leq 3$, $0 \leq digits[i] \leq 255$.

2.Definir el T.D.A Subred. Este T.D.A es una colección de servidores (s1,s2,...,sn), según se ha definido el T.D.A Servidor en el ejercicio anterior. En el T.D.A. Subred también se almacena si dos servidores están conectados (existe un enlace directo) entre ellos. Se pide:
◦ Dar la especificación del tipo Subred. Además establecer las operaciones que manejan a T.D.A
◦ Definir al menos dos *tipo rep*
◦ Escoger uno de los *tipo rep* y para este establecer la función de abstracción e invariante de representación

T.D.A Subred:

- Especificación: es una colección de servidores con una capacidad n que contiene m servidores.
- Operaciones:
  - SubRed(): crea una subred con una capacidad de 0 elementos.
  - SubRed(n): crea una subred con una capacidad n.
  - SubRed(otraSubred): crea una subred a partir de otra ya existente.
  - GetServidor(i): devuelve el servidor en la dirección i.
  - SetServidor(i, v): asigna el valor v al servidor situado en la dirección i.
  - GetCapacidad(): devuelve la capacidad de la subred.

- GetUtils(): devuelve el numero de servidores en la subred.
- Insertar(ele): inserta el servidor ele en la subred.
- Borrar(i): borra el servidor de la posición i de la subred.
- estanConectados(a, b): devuelve true si los servidores a y b están conectados y false en caso contrario.
- Conectar(a, b): conecta los servidores a y b.
- Desconectar(a, b): desconecta los servidores a y b.

- Tipo rep:
  a) class SubRed{
  private:
  Servidor* elementos;//array de elementos
  bool** conectados;//matriz de elementos conectados
  int capacidad;
  int utils;
  b) class SubRed{
  private:
  Servidor* elementos;
  bool* conectados;//las conexiones entre un servidor i con el resto se encuentran entre las posiciones utils*i y utils*i + utils
  int capacidad;
  int utils;
  int conectados;//tamaño del array conectados

- Función de abstracción para el tipo rep a:
  x es un objeto de tipo rep a. $f_a(x)$ = x.elementos[0], x.elementos[1],..., x.elementos[utils-1];
- Invariante de representación a: capacidad ≥ 0, utils ≥ 0.

3. Definir el T.D.A Punto Geográfico. Un punto geográfico se define por una latitud y longitud. La latitud es la distancia en grados desde la línea del Ecuador a los Polos. Su rango va desde -90º a 90º. La longitud es la distancia desde el meriadiano 0 al punto donde estamos. El rango de valores que adopta va desde -180º a 180º. Se pide
◦ Dar la especificación del tipo Punto Geográfico. Además establecer las operaciones que manejan a T.D.A
◦ Definir al menos dos *tipo rep*
◦ Escoger uno de los *tipo rep* y para este establecer la función de abstracción e invariante de representación

T.D.A Punto Geográfico:
- Especificación: es un par de dos elementos(latitud, longitud) que representan una ubicación geográfica.
- Operaciones:
  - PuntoGeográfico(): crea un punto geográfico con latitud y longitud igual a 0.
  - PuntoGeográfico(lat, lon): crea un punto geográfico con la latitud lat y la longitud lon.
  - PuntoGeográfico(otroPunto): crea un un punto geográfico a partir de otro ya existente.
  - GetLat(): devuelve la latitud del punto.
  - GetLon(): devuelve la longitud del punto.
  - SetLat(lat): asigna la latitud lat al punto.
  - SetLon(lon): asigna la longitud lon al punto.

- Tipo rep:
  a) class PuntoGeografico{
  private:
  float lat, lon;//valores en radianes
  b) class PuntoGeografico{
  private:
  int lon, lat;//valores en grados
- Función de abstracción para el tipo rep b:
  x es un objeto de tipo rep a. $f_a(x)$ = x.lat, x.lon;

- Invariante de representación b: $-90 \leq lat \leq 90$, $-180 \leq lon \leq 180$

4. Definir el T.D.A Ruta. Una ruta es una secuencia de puntos geográficos (ver ejercicio anterior). Se pide:
◦ Dar la especificación del tipo Ruta. Además establecer las operaciones que manejan a T.D.A
◦ Definir al menos dos *tipo rep*
◦ Escoger uno de los *tipo rep* y para este establecer la función de abstracción e invariante de representación

T.D.A Ruta:
- Especificación: es una colección de puntos geográficos con una capacidad n que contiene m puntos geográficos.
- Operaciones:
  - Ruta(): crea una ruta con una capacidad de 0 elementos.
  - Ruta(n): crea una ruta con una capacidad n.
  - Ruta(otraRuta): crea una ruta a partir de otra ya existente.
  - GetPunto(i): devuelve el punto geográfico en la dirección i.
  - SetPunto(i, v): asigna el valor v al punto situado en la dirección i.
  - GetCapacidad(): devuelve la capacidad de la ruta.
  - GetUtils(): devuelve el numero de puntos geográficos en la ruta.
  - Insertar(ele): inserta el punto geográfico ele en la ruta.
  - Borrar(i): borra el punto geográfico de la posición i de la ruta.
  - LongitudRuta(): devuelve la longitud de la ruta en kilometros.

- Tipo rep:
  a) class Ruta{
  private:
  PuntoGeográfico* ruta;
  int capacidad;
  int utils;

Implementación a partir de celdas enlazadas:
struct Celda{
PuntoGeográfico elemento;
Celda* siguiente;
}
  b) class Ruta{
  private:
  Celda* primera;
  int capacidad;//en este caso capacidad y elementos utiles vale lo mismo

- Función de abstracción para el tipo rep a:
  x es un objeto de tipo rep a. $f_a(x)$ = x.ruta[0], x.ruta[1],..., x.ruta[utils-1];

- Invariante de representación a: capacidad ≥ 0, utils ≥ 0.

5. Dar una especificación para la función que permite derivar un polinomio. Suponiendo que tenemos el TDA Polinomio, la cabecera de la función derivada sería asi:
void Derivar(const Polinomio & p_origen, Polinomio & p_derivada);

La función derivar toma como constante y pasando por referencia el polinomio p_origen, obtiene la derivada de cada una de sus componentes y las guarda en la variable p_derivada, generando un nuevo polinomio, siendo este la derivada total del polinomio origen.