Universidad de los Andes
Ingeniería de Sistemas y Computación
ISIS1205 – Algorítmica y Programación 2
Hoja de trabajo teórica Nivel 8

Nombre: _____

1. Este es el código de la clase Entidad según el diagrama del mundo (no hay TODOs. Es para su referencia).

```
public class Entidad implements Serializable
{
    private String nombre;
    private String nit;

   /**
    * Constructor de la clase
    * @param Nombre del jugador
    */
   public String Entidad (String elNombre, String elNit) {
          nombre = elNombre;
          nit = elNit;
   }
   /**
    * Devuelve el nombre de la entidad
    * @return Nombre de la entidad
    */
   public String darNombre () {
       return nombre;
   }
   /**
    * Devuelve el nit de la entidad
    * @return Nit de la entidad
    */
   public String darNit () {
       return nit;
   }

   /**
    * Devuelve el nombre formateado de la entidad
    * @return Nombre formateado de la entidad
    */
   public String toString () {
       return nit + "-" + nombre;
   }
```

2. Escriba el código de la clase SistemaFinanciero según el diagrama del mundo completando los TODOs definidos

```
    public class SistemaFinanciero implements Serializable
    {
        /**
         * Contenedor con las entidades adscritas a Datacredito
         */
        private ArrayList entidades;

        /**
        * Constructor de la clase
        */
        public SistemaFinanciero( ) throws ExcepcionFormatoInvalidoArchivo,
    IOException, ClassNotFoundException
        {
            tiposAntecedente = new ArrayList( );
            personas = new ArrayList( );
            cargarSistemaFinanciero( "./data/SistemaFinanciero.data" );
            cargarTiposAntecedentes( "./data/TiposAntecedente.txt" );
      /* TODO:  Llame al método que carga las entidades.  Suponga que el nombre del
         Archivo es ./data/Entidades.txt  (2 puntos).

        verificarInvariante( );
    }

  /**
    * Lee de un archivo de texto las entidades adscritas según formato de archivo
    * definido en el enunciado de esta evaluación.
    * @param nombreArchivo El nombre del archivo de donde se van a extraer los datos.
    * @throws ExcepcionFormatoInvalidoArchivo En caso que el archivo no tiene el formato correcto
    * @throws IOException Se lanza está excepción en caso que se encuentre un caracter inválido.
    */

  /* TODO:  Complete la signatura del método según lo definido en el contrato (3 puntos)
   public void cargarEntidades
   {
      /* TODO: Complete el método (20 puntos) */
    }

  /**
    * Lee de un archivo de texto las personas y antecedentes que son reportadas por
    * una entidad.  Se usa el formato definido en el enunciado.
    * @param nombreArchivo El nombre del archivo de donde se van a extraer los datos.
    * @param laEntidad Entidad que está reportando.
    * @param fechaAntecedente Fecha del antecedente a registrar en el sistema
    * @throws ExcepcionFormatoInvalidoArchivo En caso que el archivo no tiene el formato correcto
    * @throws IOException Se lanza está excepción en caso que se encuentre un caracter inválido.
    * @throws NoExisteTipoAntecedenteException Se lanza está excepción en caso que
    * en que alguno de los antecedentes reportados en el archivo no exista en el sistema
    */
  public void cargarAntecedentesPorEntidad (String rutaNombre, Entidad laEntidad, Date
fechaAntecedente) throws ExcepcionFormatoInvalidoArchivo, IOException,
    NoExisteTipoAntecedenteException  {

      File archivo = new File (rutaNombre);
      /* TODO: Complete las siguientes 2 instrucciones (2 puntos)
       FileReader fr =
      BufferedReader br =
      /* TODO: Lea la primera linea del archivo.  Complete la instrucción (1 punto)
      String linea =
      while ( linea != null ) {
         String [] campos = linea.split(";");
         int cedula;
```

```
int edad;
try {
   /* TODO: En cedula obtenga la cedula leída del archivo convertida a entero.
      Haga lo mismo con edad.  Complete las instrucciones (2 puntos) */
     cedula = Integer.parseInt (  );
     edad = Integer.parseInt (    );

} catch (NumberFormatException e) {
   throw new ExcepcionFormatoInvalidoArchivo ("Cedula invalida");
}
/* TODO: En nombrePersona obtenga el nombre de la persona leída del archivo
Haga lo mismo con el genero, nombre y descripción del antecedente (2 puntos) */
 String nombrePesona =
 String generoPesona =
 String nombreTipoAntecedente =
 String descripcionAntecedente =
/* TODO: Si la cedula de la persona no existe en el contenedor de personas, cree
la nueva persona e incorpórela en el contenedor de personas (3 puntos)*/
// HINT: Invoque los métodos de esta clase que le hacen lo que se pide.
         Mire el diagrama de clases del modelo del mundo.

TipoAntecedente antecedenteABuscar = null;
```

```
        /* TODO: Haga una búsqueda lineal para encontrar el tipo de antecedente que
           coincida con nombreTipoAntecedente en el contenedor de tipos de antecedentes,
           dejando el resultado en la variable antecedenteABuscar. Si no existe, deje la
           variable en null  (5 puntos)*/
        for ( int i=0; i < tipoAntecedentes.size(); i++ ) {
            // Completar según TODO.

        }
        /* TODO: Si antecedenteABuscar es null, no existe un tipo de antecedente
           válido.  Lanzar la excepción según contrato (4 puntos)*/


        }
        Persona personaABuscar = null;
        /* TODO: Haga una búsqueda lineal para encontrar en el contenedor de personas, la
           persona que contenga la cedula guardada en la variable cedula, dejando el
           resultado en personaABuscar.  Como ud. ya la insertó (ver penúltimo TODO de
           la pagina 4), tiene la certidumbre de encontrarla (5 puntos)*/
        for ( int i=0; i < personas.size(); i++ ) {
            // Completar según TODO.




        }
        /* TODO: Invoque el método adecuado de esta clase para ingresar el antecedente
           a la persona encontrada (5 puntos)
        // HINT: Mire el diagrama de clases del modelo del mundo.



        /* TODO: Lea la siguiente línea del archivo.  Complete la instrucción (1 punto)
        linea =
    }
    br.close();
}
```