Es obligatorio entregar esta hoja junto con el examen.
Todas las preguntas tienen la misma ponderación (25%).

1. Suponer que tenemos definidos los TAD **Natural** y **Cadena**. Queremos definir un tipo abstracto de datos **TorneoFutbol** que almacena los resultados de un torneo de fútbol. El número de equipos participantes en el torneo no está limitado. Podemos crear torneos y añadir los resultados de los partidos. Construir una especificación formal, usando el método axiomático, del TAD **TorneoFutbol**. El tipo tendrá las siguientes operaciones:

   - **crear**: esta operación crea un nuevo torneo inicialmente vacío (es decir, sin equipos ni partidos introducidos).
   - **partido**(*torneo*, *equipo1*, *goles1*, *equipo2*, *goles2*): dado un *torneo*, añade un nuevo resultado de un partido. *equipo1* y *equipo2* son cadenas de texto; *goles1* y *goles2* son naturales. Si ya existía el partido de *equipo1* contra *equipo2* en el *torneo*, cambia el resultado por *goles1* y *goles2* (es decir, los partidos no pueden estar repetidos).
   - **elimina**(*torneo*, *equipo1*, *equipo2*): dado un *torneo*, si se ha añadido previamente algún resultado de *equipo1* contra *equipo2* lo elimina. En otro caso, no hace nada.
   - **puntos**(*torneo*, *equipo*): devuelve un natural con el número actual de puntos de *equipo* en el torneo. Por cada partido ganado se suman 3 puntos, por empate 1 punto y por derrota 0 puntos.
   - **golesAFavor**(*torneo*, *equipo*), **golesEnContra**(*torneo*, *equipo*): calcula el número total de goles a favor y en contra, respectivamente, del *equipo* dado.

   Escribir las cuatro partes de la especificación axiomática (nombre, conjuntos, sintaxis y semántica). Se pueden añadir otras operaciones, que deberán ser especificadas también.

No entregar esta hoja con el examen. Te la puedes llevar.
Todas las preguntas tienen la misma ponderación (25%).

2. Tenemos una tabla de dispersión cerrada, con $B = 10$ cubetas, y con función de dispersión $h(k) = k \bmod B$. Insertamos los siguientes elementos: 72, 49, 99, 10, 32, 62, 45, 9 y 21. Para resolver las colisiones se utiliza una redispersión doble.

a) Definir una estrategia de redispersión doble que pueda ser válida para este caso.
b) Mostrar la inserción de los elementos en la tabla. Se debe mostrar la secuencia de búsqueda de cada elemento.
c) Razonar sobre la eliminación de los elementos 32 y 49. Mostrar las tablas resultantes de cada eliminación.
d) Supongamos que seguimos insertando elementos; la tabla se llena y quedan elementos que insertar. ¿Cómo lo resolvemos? Poner un ejemplo, mostrando la tabla resultante.

3. Dado un árbol trie, escribir una función que encuentre eficientemente todas las palabras del trie de longitud *l* y las imprima por pantalla. Para ello, se deben utilizar las operaciones genéricas sobre nodos trie que sean necesarias: *Consulta (n: trie, c: carácter): trie* (devuelve el hijo del nodo trie *n* para el carácter *c*, o NULO si no existe), y un iterador del tipo *para cada carácter c hijo del nodo n hacer*. Además, también se supone que existen funciones esVocal(c: carácter): booleano y esConsonante(c: carácter): booleano.

   La función debe tener una cabecera del tipo:

   ListarLongitud (raiz: trie; l: entero)

   Nota: si es necesario, se pueden definir funciones auxiliares.

4. Una red social está compuesta por personas y relaciones de amistad entre ellas. Hay *n* personas en total. Cada relación de amistad tiene asignada un número: el tiempo que tardan en ponerse en contacto las dos personas. La matriz **T**, de tamaño *n* x *n*, indica en cada **T**[*a*, *b*] el tiempo que tardan las personas *a* y *b*; **T**[*a*, *b*] = **T**[*b*, *a*]; y **T**[*a*, *b*] = infinito si *a* y *b* no son amigos. Dos personas cualesquiera pueden ponerse en contacto a través de amigos; el tiempo que se tarda será la suma de los tiempos de los contactos correspondientes.

   Las personas pueden ser rubias o morenas. El array **R** de booleanos, de tamaño *n*, indica en **R**[*a*] si la persona *a* es rubia (valor *true*) o morena (valor *false*).

   Dadas dos personas concretas, *a* y *b*, queremos saber si tardarán menos tiempo en comunicarse a través de amigos rubios o a través de amigos morenos. Escribir un algoritmo que calcule de manera eficiente el tiempo mínimo que tardarán en comunicarse *a* y *b* usando solo los amigos rubios y usando solo los amigos morenos.