1. TAD Complejo

Se necesita diseñar un TAD Complejo para trabajar con números complejos. La interfaz del TAD debería proporcionar la siguiente funcionalidad:

- Crear un número complejo a partir de un par de valores de tipo float (parte real e imaginaria del número complejo).
- Eliminar un número complejo previamente creado.
- Modificar la parte real de un número complejo creado previamente, asginándole un nuevo valor que se le pasa como parámetro.
- Modificar la parte imaginaria de un número complejo, asginándole un nuevo valor que se le pasa como parámetro.
- Obtener la parte real de un número complejo, sin alterar su valor.
- Obtener la parte imaginaria de un número complejo, sin modificar su valor.
- Determinar si dos números complejos son iguales.
- Copiar un número complejo en otro.
- Imprimir el número complejo en un flujo de salida en su representación carteseiana.

a) Proporcione, en pseudocódigo, la descripción funcional de la interfaz pública del TAD Complejo especificando el nombre de las funciones, su retorno, parámetros de entrada y si se modifica alguno de ellos en el transcurso de su ejecución.

b) Proporcione el fichero *complejo.h* con la interfaz pública del TAD Complejo en el lenguaje de programación C.

c) Proporcione el fichero *complejo.c* cuando el TAD Complejo se implementa con la siguiente estructura de datos en C.

```
struct _Complejo {
   float re,
   float im
}
```

d) Proporcione el codigo C de la función de prototipo

```
Status sumaCmpl (Complejo *s1, const Complejo *s2, Complejo *suma)
```

que crea un número complejo con el valor de la suma de dos complejos que se le pasan como prámetros. Considere la función como: (a) una función primitiva y (b) una función derivada. Discuta las ventajas e inconvenientes de implementar una función como función derivada o primitiva.

e) Suponga que un usuario (cliente del TAD) utiliza la implementación anterior del TAD para crear una aplicación (main.c). Posteriormente el diseñador decide modificar la implementación del TAD Complejo. Indique los ficheros que deberán ser modificados y cuales deberán ser compilados y/o linkados de nuevo.

f) Enuncie las ventajas de aislar la especificación del TAD de su la implementación.

g) Proporcione el fichero *complejo.c* cuando se implementa el TAD Complejo con la siguiente estructura de datos en C.

```
struct _Complejo {
    float z[2];
}
```

donde en el primer elemento del array se almacena la parte real y en el segundo la parte imaginaria.

2. TAD Racional.

Se necesita desarrollar una aplicación para evaluar expresiones fraccionarias. Para ello se decide diseñar el TAD Racional para manipular expresiones que involucren operaciones con números

racionales. La interfaz del TAD deberá tener la siguiente funcionalidad:

- Crear un número racional a partir de un par de dos valores de tipo entero (numerador y denominador).
- Eliminar un número racional previamente creado.
- Modificar el numerador de un número racional creado previamente, asginándole un nuevo valor que se le pase como parámetro.
- Modificar el denominador de un número racional creado previamente, asginándole un nuevo valor que se le pase como parámetro.
- Consultar el denominador de un número racional, sin alterar su valor.
- Consultar el numerador de un número racional, sin alterar su valor.
- Determinar si dos números racionales son equivalentes. Recordad que dos números racionales a/b y c/d son equivalentes si a*d == b*c.
- Copiar un número racional en otro.
- Obtener la forma reducida (simplificada) de un número racional.
- Crear el número racional inverso de un número dado.
- Imprimir el número racional en un flujo de salida expresándolo como una fracción a/b en su forma reducida.

a) Proporcione, en pseudocódigo, la descripción funcional de la interfaz pública del TAD Racional especificando el nombre de las funciones, sus retornos, parámetros de entrada y si se modifica alguno de ellos en el transcurso de su ejecución.
b) Proporcione el fichero *racional.h* con la interfaz pública del TAD Racional en el lenguaje de programación C.
c) Proporcione el fichero *racional*.c cuando el TAD Racional se implementa con la siguiente estructura de datos en C.

```
struct _Racional {
        int num;
        int den;
}
```

Para hallar la forma reducida de un número racional es necesario obtener el máximo común divisor (mcd) de del numerador y el denominador. Para ello utilice utilice el algoritmo de Euclides. Suponed que la función que calcula el mcd es un función privada (y por tanto, solo accesible a las funciones de la interfaz).

d) Proporcione el código C de la función derivada sumaRacional de prototipo

```
Racional *sumaRacional (const Racional  *s1, const Racional *s2)
```

3. TAD Conjunto:

Sabiendo que un conjunto se define coma una colección de elementos sin duplicidad, especificar el Tipo Abstracto de Datos Conjunto. La interfaz del TAD Conjunto debe proporcionar la siguiente funcionalidad:

- Crear un Conjunto vacío.
- Eliminar un conjunto.
- Obtener la cardinalidad de un conjunto (número de elementos del conjunto).
- Determinar si un elemento pertenece al conjunto.
- Insertar un elemento en el conjunto.
- Extraer un elemento del conjunto
- Imprimir en un flujo de salida los elementos de un conjunto
- Crear una copia de un conjunto dado.
- Determinar si dos conjuntos son iguales.
- Realizar la operación matemática de la unión de dos conjuntos. Esta operación devuelve un conjunto con los elementos comunes y no comunes a ambos conjuntos.
- Realizar la operación matemática de la intersección de dos conjuntos. Esta operación devuelve un conjunto con los elementos comunes a los dos conjuntos.

Suponer pare ello que se dispone de la especificación del TAD Elemento.

a) Proporcione, en pseudocódigo, la descripción funcional de la interfaz del TAD Conjunto. Esto es, el retorno, los parámetros de entrada y si se modifica alguno de ellos en el transcurso de su ejecución, para cada una de las primitivas del TAD Conjunto.
b) Proporcione el fichero conjunto.h con la especificación del TAD Conjunto en el lenguaje de programación C.
c) Proporcione el fichero conjunto.c asumiendo que se dispone de una implementación que utiliza un array estático no-ordenado como estructura de datos. Por ejemplo:

```
#define MAXCARD 100

struct _Conjunto {
        Elemento *elems[MAXCARD];
        int card;
};
```

d) Evalúe la eficiencia (medida en números de accesos) de cada una de las primitivas del TAD Conjunto cuando se utiliza para su implementación la estructura de datos anterior.
e) Supóngase que el cliente (o usuario) del TAD Conjunto quisiese aumentar la funcionalidad del TAD implementando una nueva función derivada que realizase la operación matemática de la diferencia de dos conjuntos2.¿Sería posible para el cliente implementar eficientemente dicha función con la interfaz del TAD?. Justifique la respuesta.
f) Suponga que el TAD Conjunto incluye en su interfaz, además de las primitivas reseñadas las siguientes primitivas que permiten acceder secuencialmente a los elementos del conjunto a partir de uno dado:

```
/**
    @brief : Implementa el método next de un Iterador. Considera al conjunto como una
    secuencia de elementos, devolviendo la diercción del siguiente elemento de la secuencia a partir de uno dado.
    No reserva memora
    @param:cnj Puntero al conjunto
    @param: ele Puntero a la dirección del elemento
    @return Devuelve la referencia a la dirección del siguiente elemento del  conjunto no accedido; NULL sino
    hubiese mas elementos en la secuencia o ERROR
*/
Elemento *nextEleCnj (const Conjunto *cnj, const  Elemento *ele);

/**
@brief Considera un conjunto como una secuencia de elementos. Devuelve el primer elemento de la secuencia.
@param cnj Puntero al conjunto
@return devuelve la dirección de la referencia al primer elemento del conjunto;
 NULL sino estuviese mas elementos o ERROR
*/
Elemento *firstEleCnj (const Conjunto *cnj);


/**
@brief Considera un conjunto como una secuencia de elementos. Devuelve el último elemento de la secuencia.
@param cnj Puntero al conjunto
@return devuelve la dirección de la referencia al primer elemento del conjunto;
 NULL sino estuviese mas elementos o ERROR
*/
Elemento *lastEleCnj (const Conjunto *cnj);
```

g) Proporcione el código C de la función derivada diff que devuelve la diferencia de dos conjuntos (recordad que esta operación devuelve un conjunto con los elementos del primer conjunto que no están en el segundo) asumiendo que disponemos de las primitivas del apartado anterior.

```
Conjunto *diffCnj(const Conjunto *cn1, const Conjunto *cnj2);
```

h) La estructura de datos del apartado tres limita el número de elementos que puede albergar el conjunto a MAXCARD. Para evitar este inconveniente se propone utilizar como estructura de

datos un array "dinámico". De forma que cuando se cree el conjunto se reservará memoria suficiente para alojar MAXCARD elementos. Si en algún momento al insertar un elemento el conjunto ya estuviese lleno, se aumentaría el tamaño del array con otro bloque adicional de memoria de tamaño MAXCARD. Proporcione el código de las primitivas:

```
Conjunto *creaCnj(void);
Elemento *insertEleCnj (Conjunto *cnj, Elemento *ele);
```

cuando se utiliza la siguiente estructura de datos:

```
#define MAXCARD 100

struct _Conjunto {
        Elemento **elems;
        int card;
};
```

i) Supongase que se ha desarrollado una aplicación para trabajar con conjuntos de tipo entero y que ahora se decide re-utilizar la aplicación para trabajar con conjuntos de números complejos. ¿Dependería la aplicación del tipo de elemento almacenado en el conjunto?. Explique como se consigue aislar en C la implementación del TAD Conjunto del TAD Elemento.

4. TAD Elemento

Se quiere diseñar un TAD Elemento con la siguiente funcionalidad:

- Crear un Elemento asignanándole un valor que se le pasa como argumento
- Modificar el valor de un Elemento ya creado asignanándole un valor que se le pasa como parámetro.
- Eliminar un Elemento.
- Imprimir en un flujo de salida un Elemento.
- Crear una copia de un Elemento dado sin reservar memoria para la copia
- Determinar si dos elementos son iguales.

a) Proporcione, en pseudocódigo, la descripción funcional de la interfaz del TAD Elemento. Esto es, nombre de la funcoión, retornos, parámetros de entrada y si se modifica alguno de ellos en el transcurso de su ejecución, para cada una de las primitivas de la interfaz del TAD.
b) Proporcione el fichero elemento.h con la especificación del TAD Elemento en el lenguaje de programación C.
c) Suponga que se dese implementar el TAD Elemento utilizando el tipo de dato int del lenguaje C. ¿Cúal debería ser la estructura de datos apropiada?. Proporcione el fichero elemento.c.
d) Suponga que ahora se dese implementar el TAD Elemento utilizando el tipo de dato Complejo del Problema 1. ¿Que ficheros había que modificar? Justifique la respuesta. Proporcione el código del fichero elemento.h y elemento.c

5. Biblioteca.

Se desea realizar una determinada aplicación para gestionar los libros que hay en una biblioteca. Supóngase un modelo de biblioteca “simplificado” con las siguientes características:

- Cada libro de la biblioteca se identifica por su título (cadena de caracteres de menos de 80 caracteres) y autor.
- El número de ejemplares de cada título que hay en la biblioteca es diferente, por e.g del libro “El peor viaje del mundo” de Apsley Cherry-Garrad hay 10 ejemplares pero sólo 5 ejemplares de “Anatomía de un instante” de Javier Cercas.
- La biblioteca puede comprar libros en lotes de diferente tamaño, por e.g en un determinado instante puede añadir 5 ejemplares más a sus fondos del libro “Retrato de un hombre inmaduro” .
- Nunca habrá más 25 ejemplares de ninguno de los titulo y la biblioteca nunca tendrá más de 25.000 títulos diferentes.
- Los usuarios “sacan en préstamo” libros de la biblioteca (obviamente siempre que la librería tenga el título en sus fondos y existan ejemplares disponibles) y “devuelven” los libros prestados a la biblioteca.

- Los usuarios puede tener indefinidamente los ejemplares sacados en préstamo y tampoco se registran a los usuario que tienen los libros en préstamo.
- La biblioteca imprime periódicamente un catálogo con los libros que hay en sus fondos.

a) Especifique los TAD que considere necesarios para diseñar la aplicación (Nota: Supóngase que disponemos ya de los TADs: BOOL, STATUS, CC y que no es necesario modelizar a los clientes que no es necesario modelizar a los clientes que acceden a los ejemplares de la biblioteca).
b) Escriba en código C las EdD adecuadas para los TAD anteriores.
c) Proporcione el código C de los ficheros .h de los TADs anteriores.
d) Proporcione el código C de la función primitiva:

```
/**
@name obtieneEjemplares
@brief obtiene el numero de ejemplares totales que hay de un libro
@param lib, libro
@return ejemplares del libro. -1 si ERROR
 */

int obtieneEjemplares (const LIBRO *lib);
```

e) Proporcione el código C de la función:

```
/**
@name modificaEjamplaresLibro
@brief Actualiza los fondos de una biblioteca. Si el titulo no existe se crea el libro
@param bib, biblioteca
@param nejemplar, Numero de ejmaplares que se añaden a los fondos de un titulo
@param titulo, titulo del libro que se busca
@param autor, autor del libro que se busca
@return OK/ERR
*/

STATUS modificaEjemplaresLibro (BIBLIO *bib, const int nejemplar, const char *titulo, const char *autor);
```

¿En que fichero debería crearse las funciones anteriores? Proporcione el encabezamiento de los ficheros.

6. Cesta de la compra.

7. Reserva de vuelos

Una compañía de vuelos precisa de un sistema de gestión de reserva de vuelos que incluya al menos las siguientes operaciones:

- Crear un sistema de reserva vacío, sin información.
- Añadir un nuevo vuelo al sistema de reservas.
- Registrar una reserva para un pasajero en un vuelo ya existente, con la condición de que un
- mismo pasajero no pueda hacer mas de una reserva en un mismo vuelo.
- Comprobar si un vuelo ya no admite mas reservas.
- Determinar el número de asiento asignado a un pasajero en un vuelo, suponiendo que tal asignación se realiza por orden de reserva empezando por el uno, y devolviendo cero para los pasajeros sin reserva.
- Calcular el número de reservas realizadas en un vuelo.
- Anular un vuelo en el sistema de reservas, al mismo tiempo que se eliminan todas las posibles reservas para ese vuelo
- Obtener una lista de todos los vuelos en orden de hora de salida.
- Considerar que todo vuelo se identifica por el nombre de la compañía que lo opera (por e.g IBERIA) y su número de vuelo (por e.g. UX 9117)

Especificar los TAD para el sistema de reserva e implementarlo