ESTRUCTURAS DE DATOS Y DE LA INFORMACIÓN

Examen Práctico Febrero 2010 (tipo B)

Login: Password:
Apellidos y Nombre: _____
D.N.I.: _____
Titulación: _____
Profesor: _____

Instrucciones.

- Cada alumno tiene reservada una cuenta específica para el examen cuyo login y contraseña le serán entregados.
- Cualquier intento de acceso a Internet o a otros directorios fuera de la cuenta asignada para el examen provocará que éste sea invalidado. Todos los comandos quedarán almacenados en un registro de la cuenta que podrá ser examinado posteriormente. Cualquier intento de modificar o eliminar este registro supondrá la anulación del examen.
- Durante el examen, no se permitirá hablar con los compañeros ni mirar a sus ordenadores sin pedir permiso previo a alguno de los profesores responsables. El no cumplimiento de esta norma supondrá la expulsión de la sala de examen de manera inmediata.
- En el examen se proporcionan ficheros *.cpp* para probar lo que se le pide.
- Todas las clases del examen se crearán en un único directorio.
- Puede añadir los atributos y métodos privados que considere necesarios, además de los indicados en la especificación de cada ejercicio.
- Se valorará la claridad de código y los comentarios.
- **Nota de ayuda**: si no sabe hacer alguna de las funciones, al menos, declárelas y déjelas sin código para que el programa pueda compilar. Se evaluará la parte del programa que funcione.

1. La clase Carta: definición. (2 puntos)

Debe crear la clase Carta que representa una carta de un juego. La clase Carta tendrá atributos *palo* y *número*.
Cree la clase Carta con los siguientes requisitos:

- Nombre: Carta
- Ficheros: carta.h/.cpp
- Atributos privados:
  - _palo: string
  - _numero: int
- Métodos públicos:
  - Carta: constructor vacío.
  - Carta(p, n): constructor parametrizado que recibe el palo y el número.
  - Carta(c): constructor de copia.
  - get/setPalo: acceso y modificación del atributo _palo.

- get/setNumero: acceso y modificación del atributo _numero.

2. La clase Carta: operadores. (2 puntos)

Amplíe la clase Carta con los operadores abajo indicados.
Métodos públicos:

- Operador de asignación =
- Operadores de comparación ==, **!=**
- Operador de salida (flujo) **<<**: añade al flujo la cadena *"_numero de _palo"*. Ejemplo, *"4 de copas"*.

Ayuda: `ostream& operator<<(ostream& co, const class& C);`

3. La clase Baraja: definición. (2 puntos)

Debe crear la clase Baraja que representa un conjunto de cartas. La clase Baraja tendrá un atributo privado que contiene objetos de la clase Carta.
Cree la clase Baraja con los siguientes requisitos:

- Nombre: Baraja
- Ficheros: baraja.h/.cpp
- Atributos privados:
  - _lasCartas: Carta *
  - _numCartas: int
- Métodos públicos:
  - Baraja: constructor vacío.
  - Baraja(b): constructor de copia.
  - Baraja(nc): constructor parametrizado. El parámetro indica el número máximo de elementos que puede almacenar la baraja.
  - ~Baraja: destructor. Libera la memoria reservada.
  - size(): devuelve el tamaño de la baraja.
  - operator=: operador de copia.
  - operator[]: accede al elemento i-ésimo de tal forma que puede modificarse. Se controlan errores usando assert si se trata de acceder a un elemento incorrecto.

4. La clase Baraja: ficheros. (2 puntos)

Amplíe la clase Baraja con los siguientes métodos públicos destinados al manejo de ficheros:

- saveToFile(nombreFichero): guarda en un fichero de texto, de nombre dado por el parámetro *nombreFichero* (tipo char *), las cartas actualmente almacenadas en la baraja. En cada fila del fichero se escribirán los datos de una carta:
  `elPalo_i elNumero_i`

- loadFromFile(nombreFichero): carga en la baraja desde un fichero de texto, de nombre dado por el parámetro *nombreFichero* (tipo char *), las cartas

actualmente almacenadas en dicho fichero. Se asume que en cada fila del fichero están los datos de una carta: `elPalo_i elNumero_i`
Los datos cargados sobrescriben los que pueda contener el objeto, y en caso de que el número de cartas que contiene el fichero sea mayor que el tamaño actual de la baraja, se reservará el espacio necesario para que pueda almacenar todas las cartas del fichero.

5. La clase BarajaSTL: STL. (2 puntos)

Debe crear la clase BarajaSTL que representa una baraja de cartas. La particularidad de esta clase, es que el atributo privado que representa el conjunto de cartas de la baraja debe ser implementado haciendo uso de la clase *vector* de la biblioteca STL.

- Nombre: BarajaSTL
- Ficheros: barajaSTL.h/.cpp
- Atributos privados:
  - _lasCartas: vector STL.
- Métodos públicos: los mismos de la clase Baraja.

Ayuda: copie el contenido de sus ficheros baraja.h y baraja.cpp a dos nuevos ficheros llamados barajaSTL.h/.cpp. Modifique las partes correspondientes.