CLASES. Revisión

1.a.- Realiza una clase, de nombre Examen, para guardar información sobre los exámenes de un centro educativo. La información que se guarda de un examen es: el nombre de la asignatura, el aula, la fecha y la hora. Para guardar la fecha y la hora hay que realizar dos clases, Fecha y Hora.

La clase Fecha guarda día, mes y año. Todos los valores se reciben en el constructor por parámetro. Además, esta clase debe tener un método que devuelva cada uno de los atributos y un método toString() que devuelva la información de la fecha en forma de String.

La clase Hora guarda hora y minuto. También recibe los valores para los atributos por parámetro en el constructor, tiene métodos que devuelven cada uno de los atributos y un método toString().

1.b.- Realiza una aplicación que cree un objeto de tipo Examen, lo muestre por pantalla, modifique su fecha y hora y lo vuelva a mostrar por pantalla.

2.a.- Escribe una clase Punto () que contenga los siguientes métodos:

a) los constructores y los métodos getX(), getY(), setX(), setY() y distancia()

b) Un método toString() que devuelva la información del Punto de la siguiente manera (x,y).

2.b.- Utilizando la clase Punto del ejercicio anterior, escribe una clase Polígono. Esta clase tiene como atributo un array de objetos Punto. Se consideran adyacentes los objetos Punto que estén en celdas consecutivas del array y los puntos que están en la primera y última celda. Esta clase ha de tener los siguientes métodos:

a) El constructor, recibirá por parámetro un array con los objetos Punto que definen el Polígono.

b) trasladar(), recibe por parámetro el desplazamiento en la coordenada x y el desplazamiento en la coordenada y.

c) escalar(), recibe por parámetro el factor de escala para la coordenada x y el factor de escala para la coordenada y.

d) numVértices(), devuelve el número de vértices del Polígono.

e) toString(), devuelve la información de los puntos del Polígono, uno en cada línea.

f) perímetro(). Devuelve el perímetro del polígono.

2.c.- Escribe una aplicación en la que:

a) Cree un Polígono de cuatro vértices, que serán (0,0), (2,0), (2,2), (0,2) y de color rojo.

b) Muestra la información del polígono y su perímetro por pantalla.

c) Trasladar el polígono 4 en el eje x y -3 en el eje y.

d) Muestra la información del polígono por pantalla.

HERENCIA

3.a.- Escribe una clase Multimedia para almacenar información de objetos de tipo multimedia (películas, discos, mp3...). Esta clase contiene título, autor, formato y duración como atributos. El formato puede ser uno de los siguientes: wav, mp3, midi, avi, mov, mpg, cdAudio y dvd. El valor de todos los atributos se pasa por parámetro en el momento de crear el objeto. Esta clase tiene, además, un método para devolver cada uno de los atributos y un método toString() que devuelve la información del objeto. Por último un método equals() que recibe un objeto de tipo Multimedia y devuelve true en caso de que el título y el autor sean iguales a los del objeto que llama al método y false en caso contrario.

3.b.- Escribe una clase Película que herede de la clase Multimedia anterior. La clase Película tiene, además de los atributos heredados, un actor principal y una actriz principal. Se permite que uno de los dos sea nulo, pero no los dos. La clase debe tener dos métodos para obtener los dos nuevos atributos y debe sobrescribir el método toString() para que devuelva, además de la información heredada, la nueva información.

3.c.- Escribe una clase ListaMultimedia para almacenar objetos de tipo multimedia. La clase debe tener un atributo que sea un array de objetos Multimedia y un entero para contar cuántos objetos hay almacenados. Además, tiene un constructor y los siguientes métodos:

a) el constructor recibe por parámetro un entero indicando el número máximo de objetos que va a almacenar.

b) int size(): devuelve el número de objetos que hay en la lista.

c) boolean add(Multimedia m): añade el objeto al final de la lista y devuelve true, en caso de que la lista esté llena devolverá false.

d) Multimedia get(int posición): devuelve el objeto situado en la posición especificada.

e) int indexOf(Multimedia m): devuelve la posición del objeto que se introduce por parámetro, si no se encuentra, devolverá -1.

f) String toString() devuelve la información de los objetos que están en la lista.

3.d.- Escribe una aplicación donde:

a) Se crea un objeto de tipo ListaMultimedia de tamaño máximo 10.

b) Se pidan tres películas y se añadan a la lista.

c) Se muestre la lista por pantalla.

d) Se cree un objeto de tipo Película introduciendo el título y el autor de una de las películas de la lista. Para el resto de los argumentos se utilizan valores no significativos.

e) Busca la posición de este objeto en la lista.

f) Obtenga el objeto que está en esa posición y lo muestre por pantalla junto con la posición en la que se encuentra.

3.f.- Escribe una clase Disco que herede de la clase Multimedia ya realizada. La clase Disco tiene, aparte de los elementos heredados, un atributo para almacenar el género al que pertenece (rock, pop, punk, etc.). La clase debe tener un método para obtener el nuevo atributo y debe sobrescribir el método toString() para que devuelva, además de la información heredada, la nueva información.

3.g.- Escribe una aplicación donde:

a) Se cree un objeto de tipo ListaMultimedia de tamaño máximo 10.

b) Se creen tres discos y se añadan a la lista.

c) Se muestre la lista por pantalla.

d) Se cree un objeto de tipo Disco introduciendo el título y el autor de uno de los discos de la lista, para el resto de los argumentos se utilizan valores no significativos.

e) Busca la posición de este objeto en la lista.

f) Obtenga el objeto que está en esa posición y lo muestre por pantalla junto con la posición en la que se encuentra

ÁMBITOS Y VISIBILIDAD

4.a.- Escribe una clase Coche de la que van a heredar CocheCambioManual y CocheCambioAutomatico. Los atributos de los coches son la matrícula, la velocidad y la marcha. Para este ejercicio no se permite la marcha atrás, por tanto no se permiten ni velocidad negativa, ni marcha negativa. En el constructor se recibe el valor de la matrícula por parámetro y se inicializa el valor de la velocidad y la marcha a 0. Además tendrá los siguientes métodos:

getMatrícula(): que devuelve el valor de la matrícula.

getMarcha(): devuelve el valor de la marcha.

getVelocidad(): devuelve el valor de la velocidad.

acelerar(): recibe por parámetro un valor al acelerar el coche.

frenar(): recibe por parámetro un valor al frenar el coche.

toString(): devuelve en forma de String la información del coche.

cambiaMarcha(): recibe por parámetro la marcha a la que se tiene que cambiar el coche. Este método es protected, para que puedan acceder a él las clases que heredan de Coche, pero no las clases de otros paquetes.

4.b.- La clase CocheCambioManual sobrescribe el método cambiaMarcha() y lo hace público, para que pueda ser llamado desde cualquier clase. No permite que se cambie a una marcha negativa.

4.c.- La clase CocheCambioAutomatico sobrescribe los métodos acelerar() y frenar() para que cambie automáticamente de marcha conforme se va acelerando y frenando.

COMPATIBILIDAD DE TIPOS

4.d.- Escribe una aplicación que pida por teclado la matrícula de un coche y pregunte si el coche es con cambio automático o no. Posteriormente, debe crear un coche con las características indicadas por el usuario y mostrarlo. Acelerar el coche en 60 km/h, si es un coche con cambio manual, cambiar la marcha a tercera y volverlo a mostrar.

POLIMORFISMO

5.- Escribe una aplicación donde:

a) Se cree un objeto de tipo ListaMultimedia de tamaño máximo 10. ListaMultimedia del Ejercicio 10.

b) Se creen tres discos y se añadan a la lista.

c) Se creen tres películas y se añadan a la lista.

d) Trabajando con la lista y suponiendo que no se sabe en qué posiciones están los discos y las películas:

1. Se muestre la lista por pantalla.
2. Se calcule la duración total de los objetos de la ListaMultimedia.
3. Se muestre cuántos discos hay de rock.
4. Se obtenga cuántas películas no tienen actriz principal.

6.a.- Se va a implementar un simulador de Vehículos. Existen dos tipos de Vehículo: Coche y Camión.

a) Sus características comunes son la matrícula y la velocidad. En el momento de crearlos, la matrícula se recibe por parámetro y la velocidad se inicializa a 0. El método toString() de los vehículos devuelve información acerca de la matrícula y la velocidad. Además se pueden acelerar, pasando por parámetro la cantidad en km/h que se tiene que acelerar.

b) Los coches tienen además un atributo para el número de puertas, que se recibe también por parámetro en el momento de crearlo. Tiene además un método que devuelve el número de puertas.

c) Los camiones tienen un atributo de tipo Remolque que inicializa a null (para indicar que no tiene remolque). Además tiene un método ponRemolque(), que recibe el Remolque por parámetro, y otro quitaRemolque(). Cuando se muestre la información de un camión que lleve remolque, además de la matrícula y velocidad del camión, debe aparecer la información del remolque.

d) En esta clase hay que sobrescribir el método acelerar de manera que si el camión tiene remolque y la velocidad más la aceleración superan los 100 km/h se lance una excepción de tipo DemasiadoRapidoException.

e) Hay que implementar la clase Remolque. Esta clase tiene un atributo de tipo entero que es el peso y cuyo valor se le da en el momento de crear el objeto. Debe tener un método toString() que devuelva la información del remolque.

f) Implementar la clase DemasiadoRapidoException.

Implementar este diseño con la estructura más convincente.

6.b.- Utilizando las clases del ejercicio anterior, implementa una aplicación que haga lo siguiente:

a) Declare y cree un array de 4 vehículos

b) Cree 2 camiones y 2 coches y los añada al array.

c) Suponiendo que no se sabe en qué celdas los coches y en cuáles los camiones:

1. Ponga un remolque de 5000 Kg a los camiones del array.
2. Acelere todos los vehículos.
3. Escriba por pantalla la información de todos los vehículos.