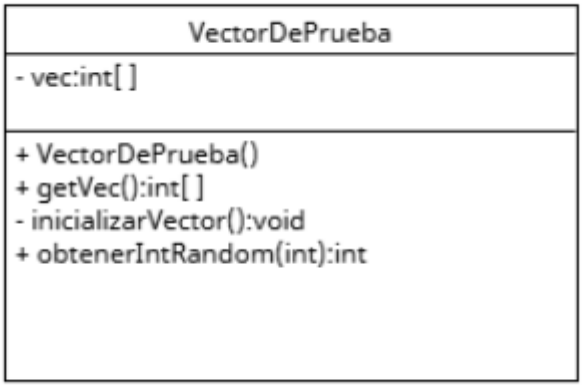
Ejercicios con arrays

Para los siguientes ejercicios se debe realizar el diagrama UML solo en los casos en que se solicite y en todos se debe escribir el código Java. Se deben desarrollar los métodos indicados por las funcionalidades y también cualquier otro que fuera necesario para cumplir con las mismas, en las clases en que corresponda.

1. Para la siguiente clase:



- Implementar en el constructor que se carguen datos aleatorios en el vector.

- Determinar cuántas veces está repetido el valor mayor y devolver el resultado.

- Devolver true o false si el promedio de los valores está almacenado en el vector o no.

- Realizar una rotación de una posición en el vector. Esto significa que el valor en la posición 0 pasa a la posición 1, el de la 1 a la 2 y así sucesivamente hasta que el de la posición 9 pasa a la 0.

- Realizar una rotación de n posiciones, donde n es un parámetro que recibe el método.

2. Una agencia de venta de automóviles usados tiene espacio para 30 vehículos. De cada vehículo se conoce su patente, su tipo (Sedan, Pick Up, Utilitario) y su precio de venta.

   Cuando se vende un vehículo se debe calcular e informar el importe a abonar. El mismo incluye el precio del vehículo más los impuestos, teniendo en cuenta las siguientes reglas de cálculo:

   - Para vehículos de valor superior a $12.000.000, se cobrará IVA de 21% y para los demás el IVA será del 10.5% .

   - Para vehículos tipo Sedan, con valor hasta $6.000.000, aplicar descuento equivalente al 50% del IVA cobrado.

   - Para vehículos tipo Pick Up y Utilitario, con valor superior a $20.000.000, aplicar sobrecosto por impuesto de rodamiento del 5% .

   - Para todos los vehículos, si el valor final es inferior a $20.000.000, aplicar descuento adicional del 5% del valor comercial

   No se podrán agregar vehículos para venta si no hay espacio en la agencia. Para agregar un vehículo se debe verificar que el mismo no se encuentre cargado en el sistema previamente.

   Cuando se vende un vehículo su espacio debe quedar disponible y se deben mostrar los datos del vehículo vendido discriminando precio y precio con impuestos.

   Se debe hacer lo siguiente, teniendo en cuenta las reglas definidas más arriba:

   - Realizar el diagrama UML.

   - Agregar un vehículo, devolviendo un valor booleano que indique si se pudo agregar o no.

   - Realizar una venta.

3. Se tiene una matriz de 6 x 4 que contiene números enteros. Primero tenemos que llenar la matriz con valores provistos por el método obtenerIntRandom de la clase VectorDePrueba.

   Una vez hecho esto, debemos desarrollar los siguientes métodos que recibirán a la matriz como parámetro:

   - Devolver la cantidad de impares que hay y a su vez reemplazar cada impar por -1.

   - Devolver un vector de 6 elementos que contenga en cada elemento cuantos -1 hay en cada fila.

   - Mostrar la posición (fila y columna) del menor valor de la matriz.

   - Dado un valor, devolver un vector de 6 elementos que contenga true o false indicando para cada fila si el valor se encuentra en la misma o no.

   - Devolver true si la cantidad de números positivos de las filas es mayor a la cantidad de números negativos de las columnas.

   - Intercambiar la fila i por la fila j siendo i y j dos valores que se reciben como parámetro.

4. Se tiene un club que tiene una lista de socios. De cada socio se tiene su nombre, el valor de su cuota y el detalle de cuotas de cada mes de un solo año, con el valor de las mismas y sus saldos pendientes.

   Cuando se carga un pago de un socio, se recibe el nombre del mismo y si el socio existe, se imputa el importe recibido a cada una de las cuotas pendientes, comenzando por la más antigua. Puede ser que el pago sea por un importe superior al valor de una cuota.

De cada socio se puede ver un detalle de su deuda mostrando para cada cuota pendiente, el número de mes y el importe que se adeuda (para los que tengan el valor de la segunda columna mayor a cero).

El club puede agregar socios (verificando que no existan antes), cargar pagos y pedir el detalle de la deuda de un socio en particular (en ambos casos de socios que existan).

En la clase PruebaClub se debería agregar socios, cargar pagos y consultar la deuda de un socio.

Además se debe realizar el diagrama UML

5. Una guardería náutica nos solicita la creación de un programa para el control de pago de las embarcaciones que guarda. Las embarcaciones se guardan en "camas", distribuidas de a cuatro camas por cada uno de sus dos niveles disponibles.

   De cada embarcación se sabe su patente, su fecha de ingreso a la guardería, la cama que ocupa y los doce pagos mensuales realizados en concepto de alquiler durante el año actual. Estos pagos están ordenados por mes. De cada pago se conoce el monto y el dni de la persona que lo efectuó. Para los meses aún no transcurridos o con pagos no efectuados se guarda null.

   Cuando se registra una nueva embarcación (alta en el sistema), se le asigna una cama. Estas se asignan de manera tal que se llena primero el nivel inferior (nivel 1) y recién cuando esté lleno se pasará a completar el nivel siguiente.

El número de cama es solo un número entero de dos cifras: la decena indica el nivel y la unidad la cama. Por ejemplo la ubicación 24 se refiere a la embarcación ubicada en el segundo nivel en la cuarta cama.

Se necesita poder emitir un listado con el estado de pagos de cada embarcación. El cliente nos pide expresamente que se pueda saber de la forma más eficiente posible si una embarcación debe alguna cuota o no.

Sabemos que al pagar siempre se cancela primero la cuota pendiente más antigua (nunca quedan cuotas sin pagar en el medio). Además nos dice que en general los mayores casos de morosidad se dan por no tener abonado el mes en curso.

Una embarcación que ingresó el año pasado y tiene todas las cuotas al día al mes de septiembre se vería de la siguiente forma (cada P es un pago):

Patente	Cama	Ingreso	Pagos del año
AA02	11	04/11/23	P P P P P P P P P null null null

Pero una embarcación que ingresó este año debe algunas cuotas al mes de septiembre se vería así (en rojo lo adeudado):

Patente	Cama	Ingreso	Pagos del año
BB21	12	10/03/24	null null P null null null null null null null null null

Resumen de funcionalidades requeridas:

- mostrar pagos por embarcación.
- registrar el pago de una cuota.
- registrar una nueva embarcación.

Además se debe realizar el diagrama UML y probar las clases desde la clase PruebaGuarderia.

Se provee la clase Fecha, que contiene métodos para gestionar objetos para el manejo de las fechas necesarias para el caso planteado.