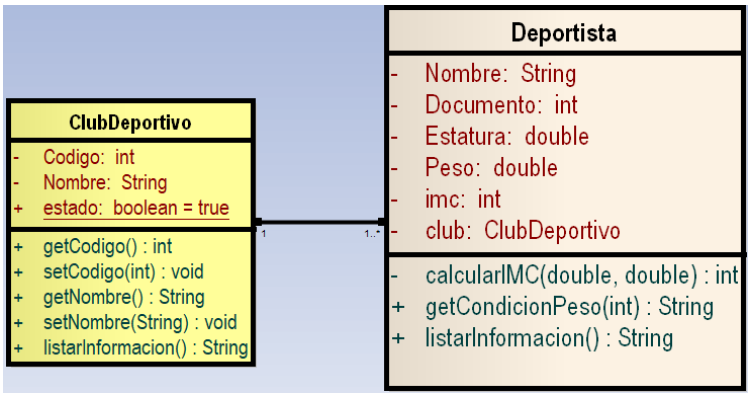
Centro de Servicios y Gestión Empresarial	ACTIVIDAD:	Aplicar listas en JAVA para almacenar objetos		
	PROGRAMA:		COMPETENCIA:	
	FICHA:		INSTRUCTOR:	
	FECHA:		APRENDIZ:	

Crear un proyecto llamado ***Prueba*** y agregar al proyecto 2 paquetes, llamados ***ejercicio1*** y ***ejercicio2***. Dentro de cada paquete debe crear un archivo ejecutable llamado ***Admin1*** y ***Admin2***

1) Realice un programa que permita crear una lista y almacenar la información de cada deportista, sin que se repita el documento de identidad del deportista en la lista, de a acuerdo a la siguiente imagen:



Se debe tener en cuenta las siguientes consideraciones:

- Aplique encapsulamiento para los atributos documento, nombre, estatura, peso, club.
- Métodos **get** para el atributos **imc.**
- Crear los constructores que considere necesarios.
- Crear una lista de tipo deportista (**ArrayList <Deportista> lista = new ArrayList()**), donde cada posición de la lista representaría la información de un deportista, es decir, los atributos del deportista.
- La fórmula para determinar el IMC es la siguiente:

$$IMC = \frac{\text{Peso (kg)}}{\text{Estatura}^2\text{(Mts.)}} \quad \frac{120\ kg}{1.65 \times 1.65} = \frac{120\ kg}{2.7225} = 44$$

- Si la persona tiene IMC menor a 18 está **bajo de peso**, si el IMC está entre 18 y 24 entonces esta **normal de peso**, si el IMC está entre 25 y 27 entonces la persona tiene **sobre peso**, de lo contrario la persona esta **obeso**

El programa debe ofrecer el siguiente **menú** al iniciar el aplicativo:

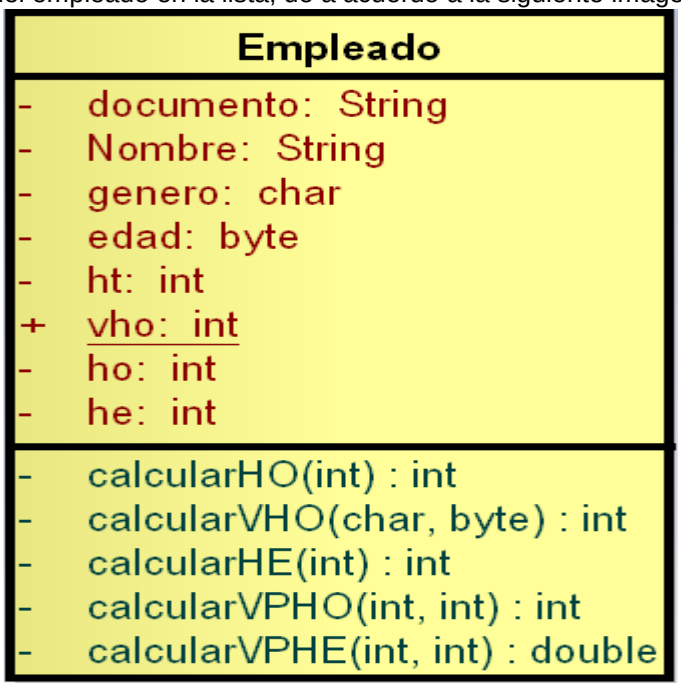
Seleccione una Opción:
1: Registrar deportista
2: Mostrar deportista
3: Modificar deportista
4: Eliminar deportista
5: Modificar Club deportivo;
6: Salir del programa

El programa debe cumplir las siguientes funcionalidades:

1. Si el usuario ingresa el numero **1** (opción *Registrar deportista*), debe ejecutar un ***método*** que capture los datos del deportista, incluyendo los del club al cual pertenece (objeto *CludDeportivo*) y el crear el objeto deportista, posteriormente agregue el deportista a la lista. Finalmente muestre el mensaje "***Deportista registrado exitosamente***".
   Nota: Si existe algún deportista con el documento debe mostrar el mensaje, "***Deportista existe en la lista***"

2. Si ingresa el numero **2** (opción *Mostrar deportista*), debe ejecutar un ***método*** que capture el documento de un deportista, lo busque en la lista y si la encuentra muestre los datos del deportista (*Documento, nombre, peso, estatura, imc, nombre del club*) y su condición de peso, de lo contrario muestre el mensaje "***Documento no se encontró".***
   Si no existe información en la lista muestre el mensaje "***La lista está vacía".***
3. Si ingresa el numero **3** (opción *Modificar deportista*), debe ejecutar un ***método*** que capture el documento de un deportista, lo busque en la lista y si la encuentra permita actualizar el *peso* o la *estatura*; de lo contrario muestre el mensaje "***Documento no se encontró"***.
   Si no existe información en la lista muestre el mensaje "***La lista está vacía".***
4. Si ingresa el numero **4** (opción *Eliminar deportista*), debe ejecutar un ***método*** que capture el documento de un deportista, lo busque en la lista y si lo encuentra, elimine la información del deportista de la lista y muestre el mensaje "***Deportista eliminado exitosamente".***
5. Si ingresa el número **5** (opción *Cambiar Club deportivo*), debe ejecutar un ***método*** que capture un nombre de un club y modifique el nombre del club al cual pertenece el deportista.
6. Solo cuando ingrese el número **6** (opción *Salir del programa*), debe terminar la ejecución del programa.

2) Realice un programa que permita crear una lista y almacenar la información de cada empleado, sin que se repita el documento de identidad del empleado en la lista, de a acuerdo a la siguiente imagen:



- Crear una lista de tipo empleado (**List <Empleado> lista = new ArrayList()**), donde cada posición de la lista representaría la información de un empleado, es decir, los atributos del empleado.
- Aplique encapsulamiento para los atributos documento, nombre, género, edad y ht.
- Métodos **get** para el atributos **vho, ho,he.**
- El atributo **vho** debe ser público y estático.

Se debe tener en cuenta las siguientes consideraciones:

- El gerente de la empresa TecniSystem Ltda. Liquida la nómina de sus empleados quincenalmente, para lo cual requiere de un programa que muestre durante su ejecución lo siguiente:
  - **A.** Documento Nombre, edad y género del empleado (**Hombre**, **Mujer**).
  - **B.** La cantidad de horas trabajadas (**HT**) quincenalmente.
  - **C.** La cantidad de horas Ordinarias (**HO**) que tendría que pagar a un empleado

D. El valor de una ***hora ordinaria*** (**VHO**) que devengará un trabajador.
E. La cantidad de horas Extras (**HE**) que tendría que pagar a un empleado.
F. El Valor que pagaría a cada empleado por horas ordinarias trabajadas (**VPHO**).
G. El Valor que pagaría a cada empleados por horas extras (**VPHE**)

- El **valor de una *hora ordinaria*** (**VHO**) dependerá del ***género y la edad*** del empleado, es decir, si el empleado es hombre y tiene menos de 18 años, el *valor de una hora ordinaria* será de $ **5000,** si el empleado es hombre y tiene 18 o más años, el *valor de una hora ordinaria* será de $ **8000,** si el empleado es mujer y tiene menos de 18 años, el *valor de una hora ordinaria* será de $ **7000,** si el empleado es mujer y tiene 18 o más años, el *valor de una hora ordinaria* será de $ **10000**

- También se sabe que cuando las horas que trabaje (***HT***) un empleado (hombre o mujer) quincenalmente, sean menores o igual a **80** se consideran ***horas ordinaria*** (***HO***). Si las horas trabajadas exceden de **80**, se consideran las primeras 80 como ***horas ordinaria*** y el resto se convierte en ***horas extras (HE)*** y se pagan con un recargo del 35% del valor de una ordinaria.

El programa debe ofrecer el siguiente **menú** al iniciar el aplicativo:

Seleccione una Opción:
1: Registrar Empleado
2: Mostrar Empleado
3: Modificar Empleado
4: Eliminar Empleado
5: Modificar VHO;
6: Salir del programa

El programa debe cumplir las siguientes funcionalidades:

1. Si el usuario ingresa el numero **1** (opción *Registrar empleado*), debe ejecutar un ***método*** que capture los datos del empleado y crear el objeto empleado, posteriormente agregue el empleado a la lista. Finalmente muestre el mensaje "***Empleado registrado exitosamente***".
   Nota: Si existe algún empleado con el documento debe mostrar el mensaje, "***Empleado existe en la lista***"
2. Si ingresa el numero **2** (opción *Mostrar empleado*), debe ejecutar un ***método*** que capture el documento de un empleado, lo busque en la lista y si la encuentra muestre los datos personales del empleado y los que corresponde a su nómina, de lo contrario muestre el mensaje "***Documento no se encontró".*** Si no existe información en la lista muestre el mensaje "***La lista está vacía".***
3. Si ingresa el numero **3** (opción *Modificar empleado*), debe ejecutar un ***método*** que capture el documento de un empleado, lo busque en la lista y si la encuentra permita actualizar la cantidad de horas trabajadas o el genero, de lo contrario muestre el mensaje "***Documento no se encontró"***. Si no existe información en la lista muestre el mensaje "***La lista está vacía".***
4. Si ingresa el numero **4** (opción *Eliminar empleado*), debe ejecutar un ***método*** que capture el documento de un empleado, lo busque en la lista y si lo encuentra, elimine la información del empleado de la lista y muestre el mensaje "***Empleado eliminado exitosamente".***
5. Si ingresa el número **5** (opción *Cambiar VHO*), debe ejecutar un ***método*** que capture el valor de la hora y modifique para todos los empleados el valor de la hora ordinaria sin importar el género o la edad del empleado.
6. Solo cuando ingrese el número **6** (opción *Salir del programa*), debe terminar la ejecución del programa.