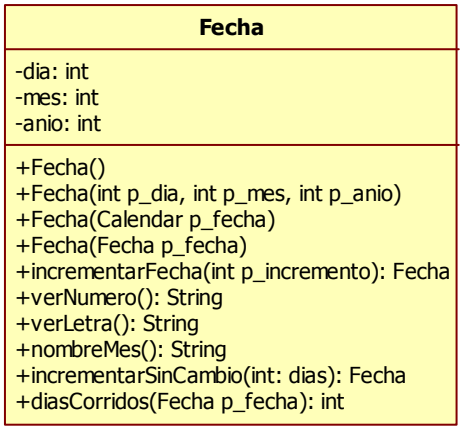
Objetivos

Que el alumno:

- Valore la reutilización del código, a través de la instanciación de clases **propias** previamente definidas
- Valore la reutilización del código, a través de la instanciación de clases **predefinidas de Java**
- Aprenda a manipular archivos de acceso secuencial y directo
- Aprenda a manipular datos de tipo fecha

1. Crear en java una clase que responda al siguiente diagrama de clase. Crear además una clase ejecutable que permita ingresar los números de una fecha para trabajar con la clase creada.



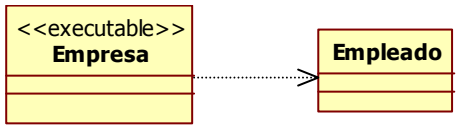
Sobrecargar el método constructor:
- inicializar el objeto con la fecha actual del sistema, si no se reciben parámetros.
- inicializar el objeto asignando los parámetros ingresados para cada atributo
- inicializar el objeto a partir de un objeto de tipo Calendar
- inicializar el objeto a partir de otro objeto de tipo Fecha

El método incrementarFecha(int), incrementa en nDias la fecha, **modifica el estado del objeto**, y devuelve la nueva fecha incrementada.
El método incrementarSinCambio(int) incrementa en nDias la fecha, **NO modifica el estado del objeto**, y devuelve un **nuevo** objeto de tipo Fecha con la fecha cambiada.
El método nombreMes() retorna el literal que indica el nombre del mes.

El método verNumero() permite visualizar la fecha en formato DD/MM/AAAA.
El método verLetra() permite visualizar la fecha en formato "DD de nombreMes de AAAA".
El método diasCorridos(Fecha) devuelve la cantidad de días corridos entre una fecha y la otra, devolviendo un valor negativo si la fecha recibida como parámetro es anterior a la fecha receptora. Considerar para este caso el método getTimeInMillis().

Atención: en todos los casos, al grabar utilizar los geter's de los atributos de las clases correspondientes.

2. Implementar una clase ejecutable Empresa que solicite los datos de sus empleados, de acuerdo a los atributos de la clase Empleado. El programa debe permitir el ingreso de datos hasta que el operador indique lo contrario. Con los datos ingresados, instanciar cada empleado y mostrarlo en pantalla con el método apropiado.

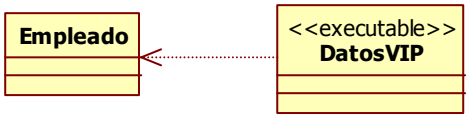


El programa además debe grabar en un archivo secuencial con el nombre "Empleado.dat" los datos de cada empleado, incluyendo el sueldo neto (utilizar el método definido a tal efecto). Utilizar los métodos apropiados de la clase Calendar para obtener, de la fecha de ingreso de cada empleado, los datos desagregados para grabar en el archivo (Dia, Mes, Año). El diseño de registro es el siguiente.

CUIL	Apellido	Nombre	Sueldo básico	Sueldo Neto	Día Ingreso	Mes Ingreso	Año Ingreso

.

3. You have to implement an executable class named DatosVip which allows generating a new sequential file, named "EmpleadoVIP.dat". Read the data from the file "Empleado.dat" and record only those employees who had more than 10 years working for the company. Record them with the same record design. You have to use the method which provides the class Employee.



4. Crear una clase ejecutable para generar el archivo de acceso directo "Laboratorio.dat". Solicitar y grabar los datos con el siguiente diseño de registro:

Cod.Laboratorio	Nombre	Domicilio	Telefono

El programa debe permitir el ingreso de datos hasta que el operador indique lo contrario.
Atencion!! A los efectos de acceder posteriormente en forma directa a los registros, Codigo de Laboratorio debe ser inicializado en 0 (cero) para el primer registro y luego se deberá incrementar para cada nuevo registro.

5. Una fundación desea imprimir las invitaciones del próximo evento que realizará. Los destinatarios de las invitaciones son leídos de un archivo secuencial de nombre "Invitados.dat", cuyo registro responde a los atributos de la clase Persona, y tiene el siguiente diseño:

DNI (int)	Apellido (String)	Nombre (String)	Día Nacimiento (int)	Mes Nacimiento (int)	Año Nacimiento (int)

A los invitados de 60 o más años se les realiza un 10% de descuento en la inscripción (utilizar método provisto por la clase para determinar la edad). El formato de la invitación es el siguiente:

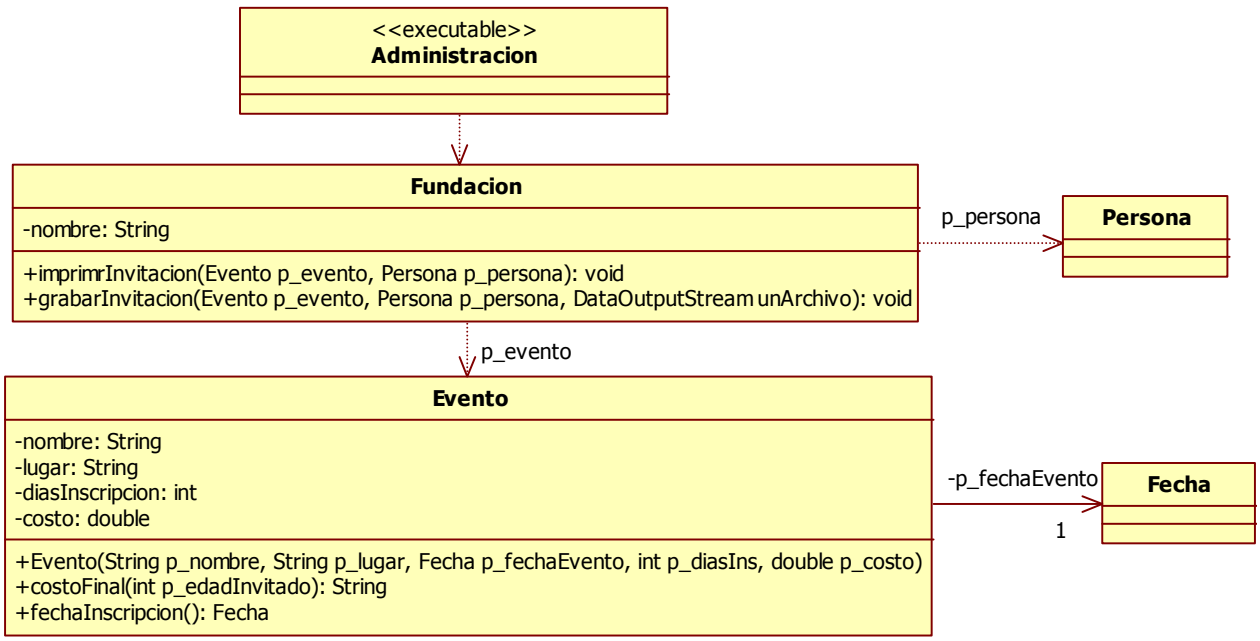
Estimado/a: ***José Martinez***

La Fundación ***Educar para Crecer***
Invita a Ud. al próximo evento: ***Congreso de Cardiopatías Congénitas****.*
Las inscripciones se realizan con ***10*** *días de anticipación.*
Las mismas se llevarán a cabo el día: ***17/10/2010***
El evento tendrá lugar en: ***Facultad Medicina UBA, el dia 27 Octubre 2010***
Costo: ***$270 (incluye descuento)***
Lo esperamos

Además se desea grabar los datos en un archivo secuencial con los siguientes datos:

DNI (int)	Apellido y Nombre (String)	Nombre Evento (String)	Lugar (String)	Costo (double)	Mes Evento (int)	Día Evento (int)

Los requerimientos se modelan en el siguiente diagrama de clases:



El atributo diasInscripcion indica cuántos días antes de la fecha del evento se realizará la inscripción.
El método fechaInscripción() retorna la fecha correspondiente.
En caso que el invitado se vea beneficiado con un descuento, éste se practicará al costo del evento, y se indicará con el mensaje "incluye descuento". Esto debe ser implementado en el método costoFinal().

Nota: El archivo "Invitados.dat" que se adjunta en el Aula Virtual contiene 3 registros, de los cuales 2 cumplen la condición para recibir descuento.

6. Un comercio mayorista de galletitas necesita conocer, al final del día, qué productos comprar y a qué proveedor hacerlo. Este comercio establece 2 políticas de compra:
   1- **Política por Faltante de Stock (FS)**: si el stock del producto es igual o menor al mínimo de compra requerido por el proveedor de dicho producto.
   2- **Política por Punto de Reposición (PR)**: si el stock del producto es menor o igual al punto de reposición del producto y, al día del análisis, faltan más de 5 días (corridos[1]) hasta el día de entrega pautado por el proveedor.

La empresa posee un archivo de acceso secuencial, donde están registrados todos los productos que maneja la distribuidora, llamado **productos.dat**; y un archivo de acceso directo, con los datos de los proveedores, llamado **proveedores.dat**. (Ver diseño de archivos, más abajo).

Al final del día se corre el proceso de "Solicitudes de Compra" en el que el sistema verifica la existencia de cada producto y, como resultado de dicho proceso, crea un archivo de acceso secuencial llamado **comprar_AAAAMMDD.txt**, en el que se registran las solicitudes de compras. Donde AAAAMMDD es la fecha en la que se corre el proceso.

Diseño de archivos

Solicitudes de compra: (comprar_AAAAMMDD.txt)

Descripción Producto (String * 30)	Nombre Proveedor (String * 30)	Cantidad compra (int)	Monto compra (double)	Razón de compra (char)

Razón de compra: F o P según se trate de un faltante de stock o de punto de reposición.

Productos: (productos.dat)

Código Producto (int)	Descripción (String * 30)	Existencia Mínima (int)	% Punto de Reposición (double)	Stock (int)	Precio (double)	Código Proveedor (int)

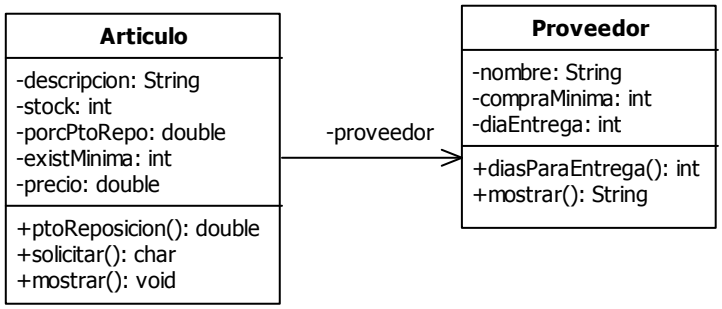
Existencia mínima: cantidad mínima del producto que debe existir en el depósito. Valor establecido por la distribuidora.
% Punto de Reposición: porcentaje por encima de la existencia mínima que indica que se debe solicitar la compra de dicho producto. Establecido por la distribuidora.

Proveedores: (proveedores.dat)

Código Proveedor (int)	Nombre (String * 30)	Compra Mínima (int)	Día de Entrega (int)

Compra mínima: cantidad mínima del producto que el proveedor vende en una orden de compra. Valor establecido por el proveedor.
Día de entrega: Día del mes pautado con el proveedor para la entrega del producto. Cada proveedor entrega solo una vez por mes. Ver **Notas Importantes (*)**.

El sistema se resume en el siguiente diagrama de Clases:



Implementar las Clases con sus respectivos constructores, getters y setters.

[1] Días corridos: considerar los 7 días de la semana. Opuesto: Días laborales (de lunes a viernes).

Clase Proveedor:

- *diasParaEntrega():* Retorna un entero, mayor que cero, que indica la cantidad de días corridos faltantes desde hoy hasta la próxima entrega del proveedor. Si hoy es el día 22 del mes y el proveedor pauta hacer entregas los días 22 de cada mes, entonces diasParaEntrega() = 30.

- *mostrar():* Retorna el String con la siguiente forma:
  Proveedor: Distribuidor Norte Galletitas

Clase Artículo:

- solicitar(): Retorna **N** si no hace falta comprar y, **F** o **P** si no se cumple la 1ra. o la 2da. política interna de compra respectivamente.

- *ptoReposicion():* Retorna el resultado de la siguiente formula
  ptoReposición = Existencia mínima * (1 + Porcentaje de Reposición / 100)

- mostrar(): Tiene la siguiente salida impresa:

  Producto: Surtido - Bagley
  Proveedor: Distribuidor Norte Galletitas

En la clase ejecutable, denominada *ReposicionAutomatica*, simular el proceso diario de armado de las solicitudes de compra. Al final del proceso se desea obtener el siguiente reporte:

Listado de productos a comprar al 22 de Septiembre de 2011

Producto: Surtido - Bagley
Proveedor: Distribuidor Norte Galletitas
Cantidad a comprar: 45 paquetes ***Día de entrega: 15 del mes***

Producto: Frutigram - Granix
Proveedor: Distribuidor Pedro Gomez
Cantidad a comprar: 45 paquetes ***Día de entrega: 1 del mes***

Cantidad de productos a reponer: 2
Importe total de la compra planificada: $ 1234,56
Cantidad de compras por faltante de stock: 1
Cantidad de compras por punto de reposición: 1

Archivo creado: comprar_AAAAMMDD.txt

Notas Importantes (*)
A modo de simplificar la tarea considerar:

- Que la cantidad a comprar es siempre la compra mínima.
- Que todos los meses tienen 30 días.
- Que todos los días son laborales.

7. Modificar el ejercicio anterior para que no considere que todos los meses tienen 30 días, sino que considere los días reales de cada mes. Tener en cuenta que Febrero tiene 29 días en año bisiesto.

8. Modificar el ejercicio 6 para que considere únicamente los días laborales (de lunes a viernes) en el cálculo de días para la entrega; así, si hoy es miércoles 17 y el proveedor entrega en 7 días, entonces los días para entrega serían 9 ya que 7 días laborales se convierten en 9 días corridos.