OTROS EJERCICIOS EXTRAS (Guía Colecciones)

Ejercicio1 (utilizar ArrayList)

Crea una clase Alumnos, donde tendrá como Atributos el nombre, apellido, la nacionalidad y su fecha de nacimiento. La nacionalidad podrá ser ARGENTINA – CHILENA – VENEZOLANA solamente.

Necesidades:

- Crear los métodos de **A – B -M** (Alta – Baja – modificación) que manipulen los objetos de este tipo.
- Crear una lista de 20 alumnos (hacer uso del constructor que recibe parámetros , para no hacer carga manual de los datos)
- Ordenar y mostrar la lista de alumnos ordenados por orden alfabético según su apellido de forma ascendente.
- Ordenar y mostrar la lista de alumnos ordenados por orden alfabético según su nombre de forma descendente.
- Con un alumno especifico (suponiendo que no existen dos alumnos con el mismo nombre y apellido), mostrar por pantalla su edad. Considerar, el manejo de fechas, utilizando el atributo fecha de nacimiento.
- Crear listas paralelas de Alumnos, según la nacionalidad.
- Realizar un reporte final de datos que informe:
    - Cuantos alumnos son mayores de 25 años.
    - Cuantos alumnos tiene su apellido que comienzan con letra L o P
    - Cuantos alumnos hay de nacionalidad ARGENTINA – CHILENA – VENEZOLANA.

Recordar:

- Hacer uso de paquetes y clases correspondientes para buena estructura de nuestro proyecto.
- Todos los métodos deben ser pensados para reutilizar y adaptarse a futuros cambios (por ejemplo, mañana quiero carga de 30 alumnos, y las instrucciones deben prevenir esta modificación)

Ejercicio2

Implementa un clase Ficha (de dominó) con su constructor y sus getters, con un toString() (que imprima el “seis-zero” como “6:0|” y con un método llamado “Ficha girarFicha()” que gire la ficha (el “6:0|” pasaría a ser “0:6|”). Implementa también el método “boolean esUnDoble()”.
A) Ahora implementa con ArrayLists una clase que genere todas las fichas desde el doble 0 al doble MAX_NUM. Después, el programa, actuando como si fuera un robot jugando al solitario, empezará por tirar el doble más grande e irá tirando fichas (las jugadas tienen que ser legales) hasta que haya tirado todas sus fichas (de su “mano” a la “mesa) o ya no pueda tirar ninguna ficha más. Tu ejecución puede ser diferente de las de los ejemplos dependiendo de cómo lo implementes.

```
run: (MAX_NUM = 2)
mano(5): 0:0|0:1|0:2|1:1|1:2|
mesa(1): 2:2|
mano(4): 0:0|0:1|1:1|1:2|
mesa(2): 0:2|2:2|
mano(3): 0:1|1:1|1:2|
mesa(3): 0:0|0:2|2:2|
mano(2): 1:1|1:2|
mesa(4): 1:0|0:0|0:2|2:2|
mano(1): 1:2|
mesa(5): 1:1|1:0|0:0|0:2|2:2|
mano(0):
mesa(6): 2:1|1:1|1:0|0:0|0:2|2:2|
BUIL SUCCESSFUL (total time: 0 seconds)
```

```
run: (MAX_NUM = 5)
mano(20): 0:0|0:1|0:2|0:3|0:4|0:5|1:1|1:2|1:3|1:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|
3:5|4:4|4:5|
mesa(1): 5:5|
mano(19): 0:0|0:1|0:2|0:3|0:4|1:1|1:2|1:3|1:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|
4:4|4:5|
mesa(2): 0:5|5:5|
mano(18): 0:1|0:2|0:3|0:4|1:1|1:2|1:3|1:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|
4:5|
mesa(3): 0:0|0:5|5:5|
mano(17): 0:2|0:3|0:4|1:1|1:2|1:3|1:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|4:5|
mesa(4): 1:0|0:0|0:5|5:5|
mano(16): 0:2|0:3|0:4|1:2|1:3|1:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|4:5|
mesa(5): 1:1|1:0|0:0|0:5|5:5|
mano(15): 0:2|0:3|0:4|1:3|1:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|4:5|
mesa(6): 2:1|1:1|1:0|0:0|0:5|5:5|
mano(14): 0:3|0:4|1:3|1:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|4:5|
mesa(7): 0:2|2:1|1:1|1:0|0:0|0:5|5:5|
mano(13): 0:4|1:3|1:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|4:5|
mesa(8): 3:0|0:2|2:1|1:1|1:0|0:0|0:5|5:5|
mano(12): 0:4|1:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|4:5|
mesa(9): 1:3|3:0|0:2|2:1|1:1|1:0|0:0|0:5|5:5|
mano(11): 0:4|1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|4:5|
mesa(10): 4:1|1:3|3:0|0:2|2:1|1:1|1:0|0:0|0:5|5:5|
mano(10): 1:5|2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|4:5|
mesa(11): 0:4|4:1|1:3|3:0|0:2|2:1|1:1|1:0|0:0|0:5|5:5|
mano(9): 2:2|2:3|2:4|2:5|3:3|3:4|3:5|4:4|4:5|
mesa(12): 0:4|4:1|1:3|3:0|0:2|2:1|1:1|1:0|0:0|0:5|5:5|5:1|
BUIL SUCCESSFUL (total time: 0 seconds)
```

Profe: Adri Bestilleiro

B) No obstante, como estamos todo el rato eliminando fichas de la “mano” y la mitad de las veces estás insertando fichas al principio de la “pila” sería más eficiente implementarlas con LinkedList.
Haz por tanto esta re-implementación con LinkedList.
Pista: Acuérdate que LinkedList incluye métodos como addFirst(), addLast(), getFirst(), getLast() que ArrayList no tiene.
Nota: Como en la “mano” no importa en que posición esté cada elemento, también podría ser un HashSet.
Pista: ¿Si te peta al eliminar una ficha de la “mano” con un java.util.ConcurrentModificationException que tienes que cambiar en como recorres la List y/o en como eliminas el elemento?

Ejercicio3(utilizar Set)

Crea un conjunto al que se le va a llamar jugadores. Inserta en el conjunto los jugadores del FC Barcelona.
Realiza un bucle sobre los jugadores del conjunto y muestra sus nombres.
Consulta si en el conjunto existe el jugador «Neymar JR». Avisa si existe o no.
Crea un segundo conjunto jugadores2 con los jugadores «Piqué» y «Busquets».
Consulta si todos los elementos de jugadores2 existen en jugadores.
Realiza una unión de los conjuntos jugadores y jugadores2.
Elimina todos los jugadores del conjunto jugadores2 y muestra el número de jugadores que tiene el conjunto jugadores2 (debería ahora ser cero)