

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

Boletín de Arrays y Colecciones

Ejercicios básicos: No es necesario comentarlos.

1. Realiza los métodos **estáticos** que se citan a continuación. Realiza en el programa principal una prueba de los mismos con un vector de tamaño 10 (Ojo, el tamaño 10 es solo en el principal, las funciones han de ser genéricas y válidas para vectores de cualquier tamaño).

a) Función que crea y devuelva un vector con **n** números enteros con valores aleatorios entre 1000 y 5000 ambos incluidos (**n** es un parámetros de la función).

b) Función a la que se le pasa un vector cualquiera de números enteros y muestra sus elementos por pantalla. Debes usar un for mejorado mostrando elemento a elemento.

c) Función a la que se le pasa un vector cualquiera de enteros (no tiene porque ser el del apartado a) y **devuelve** el valor máximo que contiene. Usa for clásico.

d) Función a la que se le pasa un vector cualquiera de enteros y devuelve el valor mínimo que contiene. Usa for mejorado.

e) Función a la que se le pasa como parámetros un vector cualquiera de enteros y dos índices (son dos números enteros) e intercambia los datos que hay en las posiciones indicadas por los índices. Si hay un error de rango devuelve *false* si no devuelve *true*. Por ejemplo si se le pasa (v, 2, 5) debe intercambiar los datos v[2] con v[5] siempre que existan dichas posiciones.

2. Repite el ejercicio anterior pero adaptándolo a una matriz de tamaño **n** filas y **m** columnas y que guarde caracteres (char) aleatorios entre 'A' y 'Z'. En el main pruébalo con una tabla de 3x4 y luego una de 4x3. Ten en cuenta que:

- Para sacar un carácter aleatorio, debes sacar un numero aleatorio en el rango unicode de los caracteres deseados y castearlo a char.
- En el apartado (b) debes mostrar el array con formato de tabla, además cada carácter de una fila debe tener una separación de al menos dos espacios con los de al lado. Debes mostrar también como cabecera el número de fila y de columna. Usa printf para la ordenación en columnas, no uses \t.

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

Ejemplo:

	0	1	2	3
0	D	R	A	Y
1	J	O	P	C
2	H	P	A	S

- El apartado (e) lógicamente tendrás que pasar 4 índices para intercambiar 2 posiciones.

3. Repite el primer ejercicio adaptándolo a un **ArrayList<Integer>**. Pruébalo con una colección de 10 elementos en el programa principal.

Añade una función que se le pase un valor como parámetro y elimine de la colección todos los elementos mayores que dicho parámetro.

Ejercicios estándar: Se deben comentar las funciones para validar.

4. Crea una clase con los siguientes métodos estáticos públicos que serán realizados **usando sólo los métodos charAt() y length()**:

a) **muestraEnLinea**: Método que muestre cada letra de una cadena que se pasa como parámetro en una línea distinta.

b) **subCadena**: Método al que se le pase una cadena, una posición de inicio y una cantidad de caracteres y devuelve el fragmento indicado. Si se le pasan parámetros no válidos (índices fuera de rango o cadena null) devuelve cadena vacía.

c) **muestraCentrado**: Función que se le pasa un string y lo muestra centrado en a consola (supón consola de 80 caracteres de ancho).

d) **cadenaAVector**: Función a la que se le pasa una cadena y devuelve un vector de char con cada uno de los caracteres de la cadena.

e) **alReves**: Función a la que se le pasa una frase y la devuelve en orden inverso.

f) **pasoAMayusculas**: Función a la que se le pasa un número indeterminado de cadenas como parámetros y las devuelve concatenadas y todo en mayúsculas

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

dentro del ASCII estándar de 7 bits (es decir, no tengas en cuenta vocales con tilde ni diéresis ni la letra Ñ), además si la cadena tiene un guión bajo (_) colocará un espacio en su lugar. Revisa el **Apéndice II** para hacer este apartado.

5. a) Crea una clase denominada Ventas que como mínimo conste de los siguientes miembros (ninguno es estático):

- Un vector público de 12 posiciones de números enteros que representaran las ventas de cada mes del año.
- Un entero privado que representa el año. Con set y get. El set comprueba que el año sea menor que el actual (busca cómo obtener el actual). Si es mayor o igual se guarda el año anterior.
- Un constructor que inicialice el vector con números aleatorios entre 0 y 999 (ambos incluidos) y el año con un parámetro.
- Un constructor sobrecargado con dos parámetros: el año y el array de tamaño 12 con el que inicializa el vector de ventas. Si no fuera de tamaño 12 lo crea automáticamente como en el otro constructor.
- Un método denominado **media** que devuelve la media de valores del vector.
- Un método denominado **grafica** que muestra en pantalla un gráfico de barras de forma que cada 100 unidades vendidas aparezca un nuevo bloque # aumentado a la barra de ese mes. Será algo así (fíjate que quede bien formateado):

```
Año 2021:
Mes 1 (10): #
Mes 2 (220): ###
Mes 3 (920): #####
Mes 4 (170): ##
...
```

b) En el programa principal (**en otra clase**) se realizan las siguientes tareas:

Puede existir un archivo denominado *ventas.txt* que tenga en la primera línea un año y en la segunda 12 valores enteros separados por punto y coma (;). Si existe lo lee y con esos datos crea un objeto Ventas llamando al constructor de dos parámetros. Para saber si existe usa la función *File.exists(nombreArchivo)*.

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

Si no existe crea un objeto de la clase Ventas con el año anterior al actual (averigua como obtener el actual y réstale 1). A continuación se muestra el gráfico de barras, luego se muestra la media con dos decimales.

Antes de acabar guarda los datos en un archivo denominado *ventasXXXX.txt*. La XXXX es el año, por ejemplo *ventas2024.txt*. Guardará el año en la primera línea y en la segunda los datos del vector separados por punto y comas (;).


c) Modifica el main de forma que si se le pasa desde la línea de comando el nombre del archivo ya trabaje con dicho archivo en vez de *ventas.txt*. Por ejemplo se podría hacer:

```
java Principal ventas2024.txt
```

Mira el **Apéndice III** de los apuntes para más información.

6. Realizar una clase denominada Matriz con los siguientes miembros:

- Una propiedad que es matriz bidimensional de números enteros.
- Un constructor que se le pase un parámetro entero N de forma que inicialice la matriz a tamaño NxN con valores entre 0 y 10 aleatorios.
- Función **estática** que muestra una matriz que se le pasa como parámetro en formato tabla. Deben aparecer los índices como cabeceras de filas y columnas. Todo correctamente formateado.
- Función **suma** con las siguientes **sobrecargas**:
 - Sin parámetros: devuelve la suma de todos los elementos de la matriz
 - Un parámetro boolean: Si está a true devuelve la suma de los elementos de la diagonal principal de la matriz (la que va de 0,0 a n,n). Si está a false la suma del resto de los elementos (todos menos la diagonal).
Trata de hacerlo con solo un bucle para la diagonal y ninguno para los que no son de la diagonal (si no te sale hazlo con varios bucles, se te dará pista en la validación).
 - Un parámetro entero: suma de los elementos de la fila indicada por el parámetro. Si dicha fila no existe devuelve -1.

	RAMA:	Informática	CICLO:	DAM				
	MÓDULO	Programación					CURSO:	1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:			
	AUTOR	Francisco Bellas Aláez (Curro)						

- Función denominada **borraFila** a la cual se le pasa un entero y devuelve la matriz pero sin la fila indicada por el número del parámetro (debes crear una matriz con una fila menos). Si el número está fuera del rango válido devuelve la matriz completa.

En el programa principal (en otra clase) se inicializa un objeto de la clase Matriz y se plantea un menú con las opciones que llaman a las funciones correspondientes:

- Mostrar matriz.
- Mostrar suma de todos los elementos.
- Mostrar suma de la diagonal.
- Mostrar suma de los elementos salvo diagonal
- Mostrar suma de elementos de una fila: la fila la escoge el usuario.
- Mostrar matriz sin una fila: la fila la escoge el usuario.
- Salir. No sale del menú hasta que se selecciona esta opción.

(Opcional) Añade una función denominada **borraFilas** a la cual se le pase una colección de números enteros que contiene números de filas: Devuelve un array como el original pero sin las filas indicadas en la colección (si hay números fuera del rango simplemente que no los tenga en cuenta).

7. Realiza una simulación de aciertos en el juego de la lotería primitiva usando colecciones. Para ello haz las siguientes funciones:

Realiza una función (denominada **rellenaCol**) que tenga como parámetro una colección de enteros. Debe limpiarla y rellenarla con 6 valores aleatorios distintos entre 1 y 49 ambos inclusive.

Otra función (denominada **compara**) a la que se le pasan dos colecciones de enteros y comprueba cuantos elementos de una colección están en la otra. Devuelve dicho valor.

En el main el usuario introducirá 6 números diferentes entre 1 y 49 separados por comas y se guardarán en una colección (se deben hacer las comprobaciones pertinentes). Si quieres haz alguna función más para modularizar esta parte.

A continuación lanza un millón de loterías mediante la función **rellenaCol** y por cada una comprueba los aciertos que ha tenido el usuario mediante **compara**. Usa un array de 7 posiciones para ir incrementando cada una de ellas según los aciertos.

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

Es decir, si compara devuelve 0 incrementas la posición 0 de ese array, si compara devuelve 3 pues rellenas la posición 3. De esa forma al final de la simulación tienes la cantidad de veces que han salido 0 aciertos, 1 acierto, 2, aciertos, etc. Muestra estos datos por pantalla y comprueba que efectivamente no vale la pena jugar.

8. Diseño de una colección (ArrayList) de videojuegos.

Se parte de una clase muy sencilla denominada **Videojuego** con los campos Título, año y fabricante. En los set de título y fabricante se hará que los textos se guarden en mayúsculas, y en el de año si este es anterior a 1950 se pondrá el año actual.

En otra clase denominada **Coleccion** tendrá una propiedad privada denominada videojuegos que será un ArrayList de objetos Videojuego.

Dispondrá de la función **menu** en la que se debe permitir al usuario las siguientes opciones:

- Insertar nuevo videojuego (se permite decidir al usuario al principio o al final de la colección si hubiera ya algún elemento).
- Visualizar toda la lista de videojuegos: Se muestra en cada fila un videojuego bien formateado en columnas incluido el índice en la colección y con cabeceras. Si el título o el fabricante ocupa más de 23 caracteres lo trunca a tamaño 20 y añade puntos suspensivos (...).
- Buscar videojuegos: En este punto el usuario mete el principio del título y mostrar todos los títulos que empiecen por dicho fragmento.
- Eliminar videojuego (por posiciones). Se pide un índice y borra el videojuego de dicha posición.
- Borrar videojuegos de un año determinado. Se pide un año al usuario y borra todos los videojuegos de ese año.
- Salir del programa. Finaliza la aplicación y guarda todos los datos en un archivo denominado *videojuegos.txt* con el formato que se explica más abajo.
-

Al ejecutar el programa se debe inicializar la colección a partir de un archivo denominado videojuegos.txt. Si el archivo no existe la colección comienza vacía (Usa File.exists()). Dicho archivo tiene en cada línea los tres datos del videojuego separados por punto y coma (;). Por ejemplo:

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

Pacman;1980;Capcom
Undertale;2015;Toby Fox
Ghost of Tsushima;2020;Sucker Punch
Hollow Knight;2017;Team Cherry

Debe haber una comprobación básica de cada línea leída de que tiene tres datos y que el del medio son 4 dígitos (puedes usar `Character.isDigit()`). Si alguna no cumple la saltas. Esto **hazlo en una función** a la que le pasas un string con la línea leída del archivo y te devuelve un objeto tipo Videojuego con los datos o null si la línea es incorrecta.

Debes hacer otras funciones para modularizar mejor el código y evitar repetir subrutinas.

Finalmente en una tercera clase denominada **Principal** estará el main en el que se instancia un objeto tipo **Coleccion** y llama a su función **menu**.

Por supuesto se debe hacer un programa claro para el usuario informando o salvando los posibles errores que se puedan producir (eliminar título no existente, que introduzca con mayúsculas o minúsculas, que introduzca espacios al principio o al final de la cadena, etc.)

9. En un instituto se desea realizar una serie de estadísticos con las notas de los alumnos. Para ello y como fase inicial de un futuro proyecto se pide la realización de un programa de simulación de notas de dicho instituto.

Se realizarán 3 clases como se describe más abajo: Clase Aula, clase Menu y clase Principal.

Como idea general se dispondrá de una tabla de notas de alumnos por varias materias y habrá un menú con las opciones siguientes:

- Ver tabla de notas.
- Ver notas de un alumno.
- Ver notas de una materia.
- Calcular la media de notas global.
- Media de un alumno.
- Media de una materia.
- Nota máxima y mínima de una asignatura.

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

- Modificación de una nota.
- Salir

En todos los casos pertinentes **se mostrarán los datos con nombres de alumnos y/o materias.**

Clase Aula:

La estructura del programa será orientada a objetos de forma que habrá una clase **Aula** que contiene como propiedades una matriz de notas privado y dos vectores de cadenas, uno con los nombres de los alumnos y otro con los nombres de las materias.

El array de notas es privado, por tanto tiene get normal y el set tendrá dos sobrecargas:

Una a la cual se le pasa una tabla de enteros y la guarda tal cual (set estándar).

Otra a la cual se le pasan 3 datos: la posición (fila y columna) y el valor a introducir.


Los vectores de nombres tienen get pero no set, son por tanto de solo lectura.

Tendrá un constructor que tiene como parámetros dos vectores de string. Uno con los nombres de los alumnos y otro con los nombres de las asignaturas. Guarda ambos parámetros en las propiedades correspondientes e inicializa el array de notas con tantas filas como alumnos y tantas columnas como asignaturas.

Rellena el array de notas con notas aleatorias entre 0 y 10 sin decimales. Se deben ponderar de forma que los porcentajes de notas sean: 0, 1 y 2 → 5% cada una; 3 → 10%; 4, 5 y 6 → 15% cada una; 7 y 8 → 10% cada una; 9 y 10 → 5% cada una.

Un segundo constructor sobrecargado tiene como parámetros cantidad de alumnos y cantidad de asignaturas. Funciona igual que el anterior pero en este caso los nombres de las asignaturas son ASIG1, ASIG2, ASIG3,... y el de los alumnos Alumno 1, Alumno 2, Alumno 3,...

El resto de la clase diseñalo tú. En ella meterás todos los métodos necesarios que actúen sobre el array a nivel de proceso es decir, sin interfaz de usuario: medias, máximos, mínimos, ...

	RAMA:	Informática	CICLO:	DAM					
	MÓDULO	Programación						CURSO:	1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:				
	AUTOR	Francisco Bellas Aláez (Curro)							

Por tanto, en esta clase Aula no habrá **nada de interfaz de usuario** (ni entrada ni salida. Ni Scanner ni prints).

Clase Menu:

Otra clase será **Menu** donde estará todo el interfaz de usuario.

Debe tener al menos una propiedad que sea un objeto de la clase Aula, un método que implemente el menú y un método por cada una de las opciones del menú que serán llamadas desde el switch.

Hazlo claro para el usuario: Cuando pidas un alumno o asignatura, muestra la lista de alumnos/asignaturas y que se seleccione con índice.

Cuando muestres datos hazlo de forma que se vea claramente lo que estás mostrando: de qué alumno o signaturas muestras los datos, que esté todo bien alineado en caso de las tablas, etc.

Si quieres puedes tener los nombres de los alumnos y de las asignaturas en archivos.

Clase Principal:

En la clase Principal estará un main muy sencillo. Ahí simplemente creas un objeto de la clase Menu y ejecuta el método donde se encuentra el menú principal.

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

Ejercicios opcionales: En estos ejercicios se plantean unas ideas generales. Como ves son un montón, por lo que no se trata de hacerlos todos ni en el orden indicado. Lo interesante es que selecciones uno que te motiva y lo amplíes con tus propias ideas. También puedes inventarte una aplicación nueva.

10. Realiza el **juego de la Oca**. Ten en cuenta que aunque visualmente es una espiral, el tablero se puede representar como un array unidimensional con reglas en ciertas casillas. Además puedes hacer que la CPU juegue, pues sería una IA muy sencilla, con lo que puedes plantear uno o dos jugadores. Puedes usar el ejemplo dado en teoría como base inicial para el juego.

11. Realiza el **juego del ahorcado**. Las palabras inicialmente estarán en una colección. Trabaja sólo con mayúsculas. Debe aparecer el dibujo correspondiente a cada fallo. Debe tener la posibilidad de ampliar la colección de palabras. También puedes intentar leer las palabras de un archivo de texto.

12. Realiza una aplicación para trabajo con **matrices matemáticas** y que se pueda realizar distintas tareas con las mismas: multiplicaciones, sumas, determinantes, transposiciones, inversiones, etc.

13. Realizar algún **otro juego de tablero** para 2 ó más personas (Ajedrez, damas, parchís, go, othelo, etc.). Si te apetece puedes, en una segunda fase, hacerlo para un jugador, pero se complica bastante pues suele haber múltiples reglas.

14. Realiza un programa que permita resolver **sudokus**. Si se te complica el algoritmo de creación de un sudoku, puedes partir de soluciones ya hechas que estén en archivos.

15. Crucigramas: Debes tenerlos en archivos tanto las soluciones como las preguntas.

16. Realizar el juego **hundir la flota** (1 jugador contra el ordenador). El ordenador colocará en un tablero de 8x8 cinco barcos: 1 de dos casillas, 3 de 3 casillas, 1 de 4 casillas en horizontal o vertical. Todo esto de forma aleatoria. También el usuario podrá colocar los barcos en su tablero. Una vez terminada la colocación ordenador y jugador irán disparando coordenadas por turnos hasta que uno de los dos haya hundido todos los barcos del otro. Si uno acierta en un barco del otro, podrá repetir tirada.

Introduce algunas reglas de IA de forma que la CPU no repita tiradas o si acierta que trate de seguir el barco.

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

17. Realiza un programa para hacer **cálculos básicos desde la línea de comando**. A dicho programa se le pasará como primer argumento un símbolo de operación (+, -, *, /) y a continuación una ristra de números. En el caso de la suma y la multiplicación operará con todos los números que se le pasa. En el caso de la resta al primer número le resta los siguientes y en el caso de la división divide el primer número entre los siguientes. En el caso de que aparezca un 0 en alguna posición de la división simplemente lo ignora y pasa al número siguiente.

Si lo deseas puedes ampliarlo con otras operaciones como raíz cuadrada, potencias... diseña tú el formato que debe tener la línea de comando.

Ejemplo: \$ java calcula + 5 6 4 10

y muestre el 25. Por supuesto la lista de números será indeterminada.

18. Juego campo de minas. El jugador parte de la esquina inferior izquierda de una región de la pantalla. Tiene que llegar a la parte superior derecha. En la pantalla hay un número de minas distribuidas de forma aleatoria y que no se ven. La cantidad dependerá del nivel de dificultad elegido. En pantalla, el jugador dispondrá de un rudimentario radar que indica el número de minas que tiene alrededor. Dispondrá de tres vidas. Los movimientos son arriba, abajo, izquierda y derecha.

Una vez que llegue a su destino, el juego comenzará de nuevo con una mina más.

Se puntúa algo por cada movimiento y un bonus por llegar al final de fase (La fase se debe indicar en pantalla). También se realizará una bonificación por tiempo.

La estructura principal será un array del tamaño de la pantalla. Utiliza cierta codificación para indicar lo que hay en cada casilla, por ejemplo: 0 vacía, 1 humano, 2 mina,... incluso valores superiores para otros objetos buenos o malos que se te ocurran. Luego el interfaz de usuario es dibujar distintas cosas en pantalla según el contenido del array.

19. Realiza una **aventura conversacional** sencilla. Establecerás un array bidimensional de objetos tipo "escenario" de forma que definas en cada escenario colecciones de objetos y personajes además de una descripción y distintas tareas que puedes usar. Puedes omar como referencia algún juego conversacional clásico como el Cobra's Arc, Zork o La guerra de las Vajillas. A continuación unos enlaces con videos o en el caso del Zork también el juego on-line.

Cobra's Arc: <https://www.youtube.com/watch?v=fNY4B9KdKxQ>

La guerra de las vajillas: <https://www.youtube.com/watch?v=GKEACSpW8Ks>

COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	DAM		
	MÓDULO	Programación				CURSO: 1º
	PROTOCOLO:	Boletín de ejercicios	AVAL:		DATA:	
	AUTOR	Francisco Bellas Aláez (Curro)				

Zork: <https://www.youtube.com/watch?v=64-VQjygTGc>

Zork y otros para jugar on-line: <http://www.web-adventures.org>

20. Juego Mastermind. El ordenador saca una combinación de 4 símbolos de 7 posibles y el usuario tiene que adivinarla. En cada turno el usuario da una combinación y el ordenador le dice cuantos aciertos de símbolos hay y cuantos aciertos de símbolo y posición hay. Usa vectores de 4 elementos. Se deben guardar todas las tiradas en una colección para realizar un informe final: número de tiradas, momento de mayor acierto (salvo el final), momento de menor acierto y otras que se te ocurran. Más información en: <http://es.wikipedia.org/wiki/Mastermind>

21. Realiza un juego de cartas. La baraja originalmente está en una colección. Busca o invéntate algoritmos de barajado de las cartas, etc.