Programación

Global Septiembre

Problema 1. (5 ptos.) → mínimo 2 ptos.

Escribe un programa que permita realizar apuestas de la "Lotería Primitiva".

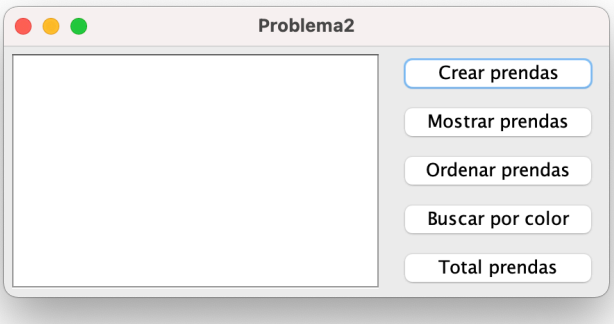
Indicaciones:

- Una apuesta consiste en 6 números DISTINTOS entre el 1 y el 49.
- Los 6 números de la apuesta se guardarán en un array.
- Una vez generada la apuesta se mostrará ordenada ascendentemente.
- Cuando se genere una apuesta se le preguntará al usuario si desea generar otra, en cuyo caso se iniciará el proceso de nuevo. Si la respuesta es negativa finalizará el programa.

NOTA: Realiza todas las validaciones necesarias y control de excepciones.

Problema 2. (5 ptos.) → mínimo 2 ptos.

Programa la siguiente ventana (proporcionada en el archivo **Problema2.zip**):



- **(0,25 ptos.)** Botón **Crear prendas.** Crea los siguientes objetos *Prenda* y almacénalos en un *ArrayList*:

Descripción	Talla	Color	Precio
Camiseta manga corta	S	Blanco	5,99
Pantalón vaquero	M	Azul medio	30
Camiseta dibujo	M	Amarillo	10
Vestido largo flores	S	Varios	50

- **(1 pto.)** Botón **Mostrar prendas**. Muestra en el área de texto todas las prendas almacenadas en el *ArrayList*.

- **(1,5 ptos.)** Botón **Ordenar prendas**. Ordena las prendas almacenadas en el *ArrayList* por descripción (orden natural) o por talla (orden total). El criterio de ordenación se elegirá mediante un cuadro de diálogo.

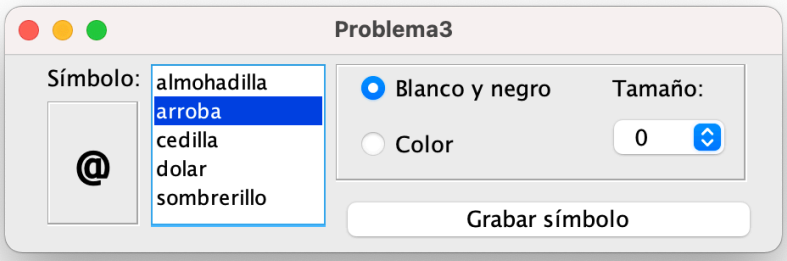
- **(1,25 ptos.)** Botón **Buscar por color.** Solicita al usuario un color a través de un cuadro de diálogo, y a continuación, muestra en el área de texto las prendas del *ArrayList* con color coincidente.

- **(1 pto.)** Botón **Total.** Muestra en un cuadro de mensaje el importe total de todas las prendas almacenas en el *ArrayList*.

NOTAS:
- La ventana se proporciona, por lo que no tienes que diseñarla tu.
- Diseña el bean *Prenda*.

Problema 3. (10 ptos.) → mínimo 4,5 ptos.

Crea un proyecto *Maven* (ventana proporcionada en el archivo **Problema3.zip**):



A. Diseña la clase SIMBOLO de acuerdo con las siguientes indicaciones:

- **(0,25 ptos.)** Un símbolo se representa por su nombre, carácter asociado, código ASCII, tamaño y color. Por ejemplo:
  - **nombre**: arroba
  - **carácter**: @
  - **ASCII**: 64
  - **tamaño**: 8
  - **color**: yellow (objeto de la clase Color)

- **(1 ptos.)** Para crear un símbolo será necesario proporcionar su nombre, carácter y ASCII asociado, en caso de que alguno de ellos se omita (ASCII carácter NULO), se lanzará una *Excepción* informando del error.

  Por defecto, un símbolo se creará de tamaño 10 y color *black*.

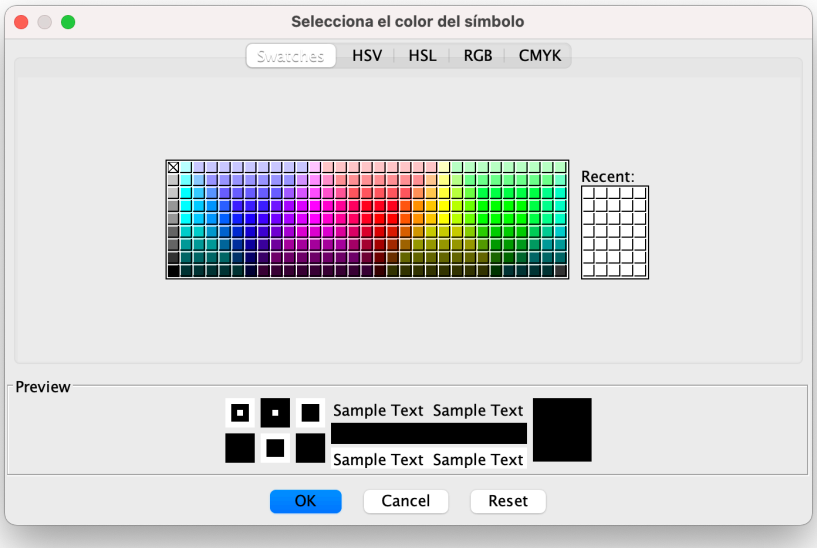
- **(2 ptos.)** Programa la funcionalidad de la clase según lo indicado en la interfaz **ElementoGráfico**.

B. Al abrirse la ventana:

- **(0,25 ptos.)** Carga el combo con los números del 0 al 99.
- **(1,25 ptos.)** Carga la lista con los objetos SIMBOLO que se encuentran en el fichero **simbolos.csv**.
- **(0,25 ptos.)** Configura los botones de opción para que sólo se pueda seleccionar uno de ellos.
- **(0,25 ptos.)** Configura la lista para que solo se pueda seleccionar un elemento.

C. (1,5 ptos.)

Si se selecciona la opción Color se permitirá al usuario modificar el color del símbolo a través de un diálogo de selector de colores como el siguiente (*JColorChooser)*:



D. (1,5 ptos.)

Al seleccionar un símbolo de la lista (*JList*), se mostrará el carácter asociado en la etiqueta:

Además al hacer doble clic en la etiqueta con el símbolo (*JLabel*) se mostrará su código ASCII a través de un cuadro de diálogo.

E. (1,75 ptos.)

Al pulsar el botón "*Grabar símbolo*" se insertará una nueva fila en la tabla *simbolos* de la base de datos **MySQL *simbolos***, sólo si se ha elegido un símbolo de la lista y el tamaño es superior a 0, en caso contrario se mostrará un mensaje de error al usuario.

Después de insertar el símbolo en la base de datos, se informará al usuario del nº de símbolos iguales (con el mismo nombre), que hay en la base de datos.

NOTA: La ventana se proporciona, por lo que no tienes que diseñarla tu.