Nombre _____
ED-Ex2B-20180207

Haz un proyecto con la siguiente funcionalidad:
(2 puntos) En un TextArea a la izquierda de la ventana el usuario podrá escribir palabras (una por renglón). Cuando se pulse un botón situado debajo (con el texto Leer), revisará el contenido del TextArea quedándose de cada línea con las 15 primeras letras, eliminando las líneas vacías y las líneas repetidas (refrescando el TextArea con los valores depurados).

(2 puntos) Después, en un panel situado a la derecha del TextArea, pondrá una lista con tantos ComboBox<String> como palabras correctas tengamos, con el texto de cada una de ellas en cada línea del ComboBox. Podemos usar el constructor del ComboBox que permite construir el modelo pasando un vector de cadenas.

(2 puntos) Cuando pulsemos un botón situado debajo de esta lista de ComboBox, el programa comprobará que no hay selecciones repetidas en los ComboBox y que están en el orden impuesto por uno de los dos botones de radio situados debajo (alfabético ascendente y por longitud de la palabra descendente). Indicará el error (o el éxito) en una etiqueta situada más abajo. Podemos saber la cadena seleccionada en cada ComboBox usando el método getSelectedItem() y los componentes del panel con getComponents()

(2 puntos) Para trabajar el "modelo o negocio" te apoyarás en una clase que has de construir: Grupo que lleva todo el control de un ArrayList de String: cuántos elementos tiene (tamaño), añadir nuevos elementos (poner) depurándolos (si vacíos no lo añade y devuelve -2, si está repetido no lo añade y devuelve -1, si supera las 15 letras añade sólo las 15 primeras y devuelve 0, en otro caso añade y devuelve 0), vaciar el ArrayList, coger (tomar) el elemento situado en una posición del Array, comprobar si está ordenado alfabéticamente ascendente, comprobar si está ordenada por la longitud de las cadenas de forma descendente.

(2 puntos) Se harán las pruebas de la clase Grupo usando la clase GrupoTest generada con JUnit que se adjunta.

AYUDAS: el TextArea se usa como un TextField que permite más de una línea (separador de líneas: System.lineSeparator() )
La cadena que devuelve el método getText() podemos trocearla con el método split(System.lineSeparator())

```
public class GrupoTest {

   Grupo i1, i2, i4, i5, i6;
   public GrupoTest() {
      i1= new Grupo(); i1.poner("tres"); i1.poner("dos"); i1.poner("uno");
      i2= new Grupo(); i2.poner("cuatro"); i2.poner("dos"); i2.poner("tres");
```

```
    i4= new Grupo(); i4.poner("uno");
    i5= new Grupo();
    i6= new Grupo(); i6.poner("algo"); i6.poner("cinco");
                i6.poner("cuatro"); i6.poner("dos");

}

@Test
public void testTamaño() {
    System.out.println("tamaño");
    assertEquals(3, i1.tamaño());
    assertEquals(1, i4.tamaño());
    assertEquals(0, i5.tamaño());
}

@Test
public void testVaciar() {
    System.out.println("limpiar");
    Grupo i3= new Grupo(); i3.poner("cinco"); i3.poner("tres"); i3.poner("dos");
    assertEquals(3, i3.tamaño());
    i3.vaciar();
    assertEquals(0, i3.tamaño());
}


@Test
public void testPoner() {
    System.out.println("poner");
    Grupo i3= new Grupo(); i3.poner("cinco"); i3.poner("tres"); i3.poner("dos");
    assertEquals(-1, i3.poner("dos"));
    assertEquals(-2, i3.poner(""));
    assertEquals(0, i3.poner("cuatro"));
    assertEquals(0, i3.poner("012345678901234567890123456789012345"));
    assertEquals("012345678901234", i3.tomar(4));
    assertEquals("cuatro", i3.tomar(3));
}

@Test
public void testTomar() {
    System.out.println("tomar");
    assertEquals("tres", i1.tomar(0));
    assertEquals("uno", i1.tomar(2));
    assertEquals("", i1.tomar(3));
    assertEquals("", i1.tomar(-8));
}

@Test
public void testOrdenadoAlfabeticoAscendente() {
    System.out.println("ordenadoAlfabeticoAscendente");
```

```
        assertEquals(false, i1.ordenadoAlfabeticoAscendente());
        assertEquals(true, i6.ordenadoAlfabeticoAscendente());
        assertEquals(true, i4.ordenadoAlfabeticoAscendente());
    }

    @Test
    public void testOrdenadoLongitudDescendente() {
        System.out.println("ordenadoLongitudDescendente");
        assertEquals(true, i1.ordenadoLongitudDescendente());
        assertEquals(false, i2.ordenadoLongitudDescendente());
        assertEquals(true, i5.ordenadoLongitudDescendente());
    }

}
```