
	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 1</p>

## INFORME DE LABORATORIO

### (formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	Fundamentos de la programación 2				
TÍTULO DE LA PRÁCTICA:	Arreglos estandar				
NÚMERO DE PRÁCTICA:	02	AÑO LECTIVO:	2024-B	NRO. SEMESTRE:	II
FECHA DE PRESENTACIÓN	27/09/2024	HORA DE PRESENTACIÓN	09:45:15		
INTEGRANTE (s) Mauro Snayder Sullca Mamani				NOTA (0-20)	
DOCENTE(s): Ing. Lino Jose Pinto Oppe					

RESULTADOS Y PRUEBAS	
<p><b>I. EJERCICIOS RESUELTOS:</b></p> <p><b>1. JUEGO DEL AHORCADO</b></p> <p>En este ejercicio se le solicita a usted implementar el juego del ahorcado utilizando el código parcial que se le entrega.</p> <p>Deberá considerar que:</p> <ul style="list-style-type: none"> <li>El juego valida el ingreso de letras solamente. En caso el usuario ingrese un carácter equivocado le dará el mensaje de error y volverá a solicitar el ingreso</li> <li>El juego supone que el usuario no ingresa una letra ingresada previamente</li> <li>El método <code>ingreseLetra()</code> debe ser modificado para incluir las consideraciones de validación. Puede crear métodos adicionales</li> </ul>	
<pre> 7  /** 8   * 9   * @author Mauro Snayder 10  */ 11  import java.util.*; 12 13  public class Actividad { 14      public static void main(String []args){ 15          String ahorc1 = " +---+ \n" + 16                        "       \n" + 17                        "       \n" + 18                        "       \n" + 19                        "       \n" + 20                        "       \n" + 21                        "===== "; </pre>	

```

String ahor2 = " +---+ \n"+
               " | | \n"+
               " O | \n"+
               " | \n"+
               " | \n"+
               " | \n"+
               "=====";

String ahor3 = " +---+ \n"+
               " | | \n"+
               " O | \n"+
               " | | \n"+
               " | \n"+
               " | \n"+
               "=====";

String ahor4 = " +---+ \n"+
               " | | \n"+
               " O | \n"+
               " /| | \n"+
               " | \n"+
               " | \n"+
               "=====";

String ahor5 = " +---+ \n"+
               " | | \n"+
               " O | \n"+
               " /|\ | \n"+
               " | \n"+
               " | \n"+
               "=====";

String ahor6 = " +---+ \n"+
               " | | \n"+
               " O | \n"+
               " /|\ | \n"+
               " / | \n"+
               " | \n"+
               "=====";

String ahor7 = " +---+ \n"+
               " | | \n"+
               " O | \n"+
               " /|\ | \n"+
               " / \ | \n"+
               " | \n"+
               "=====";

String [] figuras = {ahor1, ahor2, ahor3, ahor4, ahor5, ahor6, ahor7};
int contador=1;
String letra;
String [] palabras = {"programacion", "java", "indentacion", "clases","objetos",
                      "desarrollador", "pruebas"};



String palSecreta = getPalabraSecreta(palabras);

```

```

70 // PalabraInc: Almacena en un arreglo tipo "char" las letras correctas que ingresa el usuario.
71 Character[] palabraInc=new Character[palSecreta.length()];
72 // Ponemos subguiones a todos los espacios del arreglo, para que este luego sea completado.
73 for(int i=0;i<palabraInc.length;i++)
74     palabraInc[i]='_';
75 System.out.println(figuras[0]);
76 mostrarBlancos(palSecreta);
77 System.out.println("\n");
78 while(contador <= 6){
79     letra=ingreseLetra(palabraInc);
80     if (letraEnPalabraSecreta(letra, palSecreta)){
81         // Si la letra pertenece a la palabra secreta, entonces
82         // actualizamos el arreglo "palabraInc"
83         mostrarBlancosActualizados(letra,palSecreta,palabraInc);
84         // si ya no hay subguiones en el arreglo entonces gana el usuario.
85         if (palabraCompleta(palabraInc)) {
86             System.out.println("\nUsted gana");
87             break;
88         }
89     }
90     else{
91         System.out.println(figuras[contador]);
92         mostrarBlancosActualizados(letra,palSecreta,palabraInc);
93         contador=contador+1;
94     }
95 }
96 // El contador inicia con 1, y si despues falla sus 6 intentos entonces pierde el juego.
97 if (contador==7)
98     System.out.println("\nUsted perdio");
99 }
100 public static String getPalabraSecreta(String [] lasPalabras){
101     int ind;
102     int indiceMayor=lasPalabras.length -1;
103     int indiceMenor=0;
104     ind = (int) ((Math.random()*(indiceMayor-indiceMenor+1)+indiceMenor));
105     return lasPalabras[ind];
106 }
107 public static void mostrarBlancos(String palabra) {
108     for(int i=0; i< palabra.length(); i++)
109         System.out.print("_ ");
110 }
111 public static String ingreseLetra(Character[] palabraInc){
112     String laLetra;
113     Scanner sc=new Scanner(System.in);
114     System.out.print("\nIngrese letra: ");
115     laLetra=sc.next();
116     // Con este bucle y condicion, estamos validando que no se ingrese letras correctas que
117     // el usuario ya ingreso anteriormente.
118     for(int i=0;i<palabraInc.length;i++){

```

	<p style="text-align: center;"><b>UNIVERSIDAD NACIONAL DE SAN AGUSTIN</b>  <b>FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS</b>  <b>ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</b></p>	
<p style="text-align: center;"><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p><b>Aprobación:</b> 2022/03/01</p>	<p><b>Código:</b> GUIA-PRLE-001</p>	<p><b>Página:</b> 4</p>

```

119         if (laLetra.charAt(0) == palabraInc[i]) {
120             System.out.println("Error usted ya ingreso esta letra.");
121             laLetra = "-1";
122         }
123     }
124     // Estamos validando que el usuario no haya ingresado una cadena de mas de una letra o un numero.
125     while (laLetra.length() != 1 || Character.isDigit(laLetra.charAt(0))) {
126         System.out.print("Error, Ingrese letra: ");
127         laLetra = sc.next();
128     }
129     return laLetra;
130 }
131 public static boolean letraEnPalabraSecreta(String letra, String palSecreta) {
132     // Con charAt() accedemos a una letra de la palabra secreta y lo comparamos con la letra que ingreso el
133     // usuario, si son iguales retornamos True; caso contrario, False.
134     for (int j = 0; j < palSecreta.length(); j++) {
135         if (letra.charAt(0) == palSecreta.charAt(j))
136             return true;
137     }
138     return false;
139 }
140 public static void mostrarBlancosActualizados(String letra, String palSecreta, Character[] palabraInc) {
141     // Creamos un nuevo arreglo temporal para copiar el arreglo "palabraInc"
142     // Esto nos servirá para que no se pierdan el progreso cuando el usuario ingrese otra letra correcta.
143     Character[] actualizar = new Character[palSecreta.length()];
144     System.arraycopy(palabraInc, 0, actualizar, 0, palabraInc.length);
145     // Introducimos al arreglo temporal la letra que ingreso el usuario.
146     for (int j = 0; j < actualizar.length; j++) {
147         if (letra.charAt(0) == palSecreta.charAt(j))
148             actualizar[j] = letra.charAt(0);
149         // Imprimimos el arreglo con las palabras que el usuario ingreso.
150         System.out.print(actualizar[j] + " ");
151     }
152     // Actualizamos el arreglo "palabraInc" con la letra correcta que ingreso el usuario.
153     System.arraycopy(actualizar, 0, palabraInc, 0, actualizar.length);
154 }
155 public static boolean palabraCompleta(Character[] palabraInc) {
156     // Verificamos que no haya subguiones en el arreglo "palabraInc", para así
157     // llegar a la conclusion de que el usuario completó toda la palabra secreta.
158     for (int i = 0; i < palabraInc.length; i++) {
159         if (palabraInc[i] == '_')
160             return false;
161     }
162     return true;
163 }
164 }

```

## II. PRUEBAS

*¿Con que valores comprobaste que tu práctica estuviera correcta?*

*Ingresando cadenas con mas de una letra, cadenas con números, o letras correctas ya ingresadas anteriormente.*

*¿Qué resultado esperabas obtener para cada valor de entrada?*

*Se espera que las letras correctas revelen su posición en la palabra, las incorrectas aumenten el estado del ahorcado, y las entradas no válidas soliciten nuevamente una letra.*

*¿Qué valor o comportamiento obtuviste para cada valor de entrada?*

*Las letras correctas se revelan en la palabra, mientras que las incorrectas aumentan el estado del ahorcado. Las letras repetidas y los caracteres no válidos solicitan nuevamente una entrada, y al ganar o perder, se muestra el mensaje correspondiente.*

**Output - Mauro02 (run)**

```

+---+
|   |
|   |
|   |
|   |
=====
- - - - -


Ingrese letra: a
_ _ _ _ _ a _
Ingrese letra: e
_ _ e _ a _
Ingrese letra: i
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: o
_ _ e _ a _
Ingrese letra: o
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: u
_ _ u e _ a _
Ingrese letra: m
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: p
p _ u e _ a _
Ingrese letra: r
p r u e _ a _
Ingrese letra: b
p r u e b a _
Ingrese letra: s
p r u e b a s
Usted gana

```

 **Output**
**Output - Mauro02 (run)**

```

=====
Ingrese letra: q
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: h
+---+
|   |
|   |
|   |
|   |
=====
- - - - -


Ingrese letra: m
_ _ _ _ _ m _
Ingrese letra: b
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: v
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: x
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Usted perdio
BUILD SUCCESSFUL (total time: 2

```

 **Output**
**Output - Mauro02 (run)**

```

Ingrese letra: p
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: a
_ _ a _ _
Ingrese letra: e
_ _ a _ e _
Ingrese letra: i
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: o
_ _ a _ e _
Ingrese letra: o
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: u
+---+
|   |
|   |
|   |
|   |
=====
- - - - -

Ingrese letra: a
Error usted ya ingreso esta letra.
Error, Ingrese letra: l
Error, Ingrese letra: c
c _ a _ e _
Ingrese letra: l
c l a _ e _
Ingrese letra: s
c l a s e s
Usted gana
BUILD SUCCESSFUL (total time: 1 mi

```

### III. RUBRICA:

Contenido y demostración		Puntos	Checklis t	Estudiant e	Profeso r
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2		1	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4		1,5	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2		2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2		2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2		1,5	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2		2	
7. Ortografía	El documento no muestra errores ortográficos.	2		2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4		4	
TOTAL		20		16	



Tabla 2: Rúbrica para contenido del Informe y demostración

### CONCLUSIONES

*El uso de arreglos unidimensionales y arraycopy en Java optimiza la gestión y manipulación de colecciones de datos, mejorando tanto la eficiencia como la claridad del código.*

### METODOLOGÍA DE TRABAJO

*Lo primero que hice, es leer cada los enunciados de cada ejercicio y también tomar en cuenta las restricciones que nos da para así poder buscar una solución al problema. Después observar el código y entender la funcionalidad de cada uno y completar las partes que están incompletas. Y por último comprobar*

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p><b>Aprobación:</b> 2022/03/01</p>	<p><b>Código:</b> GUIA-PRLE-001</p>	<p><b>Página:</b> 7</p>

*nuestro código ingresando varias veces valores de prueba para ver que nuestro código está funcionando correctamente.*

## REFERENCIAS Y BIBLIOGRAFÍA

*M. W. Aedo López, Fundamentos de programación I: Java Básico, 1st ed. Arequipa, Perú: Universidad Nacional de San Agustín, Jul. 2019. ISBN: 978-612-4337-55-0. 116 p. [Enseñanza universitaria o superior]. Impreso, tapa blanda, 21 x 29.7 cm.*