Informe de Laboratorio 02

Tema: Laboratorio 02

Nota

Estudiante	Escuela	Asignatura
Victor Mamani Anahua vmamanian@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de la Programación II Semestre: II Código: 20230489

Laboratorio	Tema	Duración
02	Laboratorio 02	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 10 Setiembre 2023	Al 17 Setiembre 2023

1. Tarea

- En este ejercicio se le solicita a usted implementar el juego del ahorcado utilizando el código parcial que se le entrega.
- Deberá considerar que:
- 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
- El juego supone que el usuario no ingresa una letra ingresada previamente
- El método ingreseLetra() debe ser modificado para incluir las consideraciones de validación
- Puede crear métodos adicionales

2. Equipos, materiales y temas utilizados

- Sistema Operativo Ubuntu GNU Linux 23 lunar 64 bits Kernell 6.2.v
- Sistema Operativo Windows 11.
- Visual Studio Code.
- OpenJDK 64-Bits 19.0.7.
- Git 2.39.2.

- Cuenta en GitHub con el correo institucional.
- Programación Orientada a Objetos.
- Actividades del Laboratorio 01

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/VictorMA18/fp2-23b.git`
- URL para el laboratorio 01 en el Repositorio GitHub.
- `https://github.com/VictorMA18/fp2-23b/tree/main/Fase01/Lab02`

4. Actividades del Laboratorio 02

4.1. Ejercicio Ahorcado

- En el primer commit en el metodo mostrarBlancosActualizados lo modificamos para el uso de que se compare cada letra ingresada con cada caracter de la palabra secreta y al final de esta "guardarla en el array fill para su posterior uso".
- El codigo y el commit seria el siguiente:

Listing 1: Commit

```
$ git commit -m " Modificamos el metodo mostrarBlancosActualizados y completamos el
    metodo letraEnPalabraSecreta y agregamos un Mensaje de perdio y cuantos intentos
    uso"
```

Listing 2: Las lineas de codigos del metodo modificado:

```
public static void mostrarBlancosActualizados(String letra, String palSecreta, String[]
    fill){ // Agregamos al dominio un string y un string[] para su funcionamiento del
    metodo
    //COMPLETAR
    for(int z = 0; z < palSecreta.length(); z++){
        if(letra.charAt(0) == palSecreta.charAt(z)){
               fill[z] = letra.substring(0,1);
        }else{
            if(fill[z] == null){
                fill[z] = "_ ";
            }
        }
    }
    for(int a = 0; a < fill.length; a++){
        if(a == fill.length - 1){
            System.out.print(fill[a] + "\n");
        }else{
            System.out.print(fill[a]);
        }
    }
}
```

4.2. Ejercicio Ahorcado

- En el primer commit tambien completamos el metodo letraEnPalabraSecreta el cual nos da retorna un valor booleano el cual funciona como un indicador su la palbra ingresada es igual al caracter de la palabra secreta
- El codigo y el commit seria el siguiente:

Listing 3: Commit

```
$ git commit -m " Modificamos el metodo mostrarBlancosActualizados y completamos el
    metodo letraEnPalabraSecreta y agregamos un Mensaje de perdio y cuantos intentos
    uso"
```

Listing 4: Las lineas de codigo del metodo completado;

```
public static boolean letraEnPalabraSecreta(String letra, String palSecreta){
    //COMPLETAR
    int count = 0;
    for(int x = 0; x < palSecreta.length(); x++){
        if(letra.charAt(0) == palSecreta.charAt(x)){
            count++;
        }
    }
    return count > 0; // Agregamos el contador para sacar un valor booleano de la
        letra ingresada
}
```

4.3. Ejercicio Ahorcado

- En el primer commit tambien completamos añadimos un if que se va a dar cuando el usuario llego a la ultima imagen y se imprime un mensaje de error y cuantos intentos uso en el ciclo
- El codigo y el commit seria el siguiente:

Listing 5: Commit

```
$ git commit -m " Modificamos el metodo mostrarBlancosActualizados y completamos el
    metodo letraEnPalabraSecreta y agregamos un Mensaje de perdio y cuantos intentos
    uso"
```

Listing 6: Las lineas de codigo de lo añadido:

```
while(contador <= 6){
    letra = ingreseLetra();
        if (letraEnPalabraSecreta(letra, palSecreta)){
            mostrarBlancosActualizados(letra,palSecreta,fill);
            if(finish(fill, palSecreta,contador)){ // ESTRUCTURA DE CONTROL USADA Y
                TAMBIEN EL METODO CREADO QUE ES FINISH
                break;
            }
        }else{
            System.out.println(figuras[contador]);
            if(figuras[contador] == ahor7){
```

```
                    System.out.println("Usted a perdido y el numero de intentos es : " +
                        contador); // MENSAJE DE PERDIO Y CUANTOS INTENTOS USO
                }
                contador = contador +1;
            }
        }
    }
```

4.4. Ejercicio Ahorcado

- En el segundo commit creamos el metodo arraynew que nos serviria para crear un array de tipo string que hace una comparacion con el array fill debido que en este metodo te retornara un array con la palabra secreta y este se comparara con el array fill para su comparacion
- El codigo y el commit seria el siguiente:

Listing 7: Commit

```
$ git commit -m "Creamos los metodos arraynew y finish y tambien ponemos una estructura
    de control en el ciclo while"
```

Listing 8: Las lineas de codigo de lo creado:

```
public static String[] arraynew(String palSecreta){ // Creamos un metodo que cree un array
    para poder usarlo como un comparador que se ve en el metodo finalizar
    String[] arraysecret = new String[palSecreta.length()];
    for(int x = 0; x < palSecreta.length(); x++){
        arraysecret[x] = palSecreta.charAt(x) + "";
    }
    return arraysecret;
}
```

4.5. Ahorcado

- En el segundo commit creamos el metodo finish en la que usamos el metodo arraynew y tambien verifcamos que en fill no tenga un string (-)
- Tambien agregamos el mensaje de haber ganado con el numero de intentos que se dio
- El codigo y el commit seria el siguiente:

Listing 9: Commit

```
$ git commit -m "Creamos los metodos arraynew y finish y tambien ponemos una estructura
    de control en el ciclo while"
```

Listing 10: Las lineas de codigo de lo creado:

```
public static boolean finish(String[] fill,String palSecreta, int intentos){//METODO
    CREADO PARA SABER SI GANO Y CUANTOS INTENTOS USO
    String[] arraysecret = arraynew(palSecreta);
    int count = 0;
```

```
    for (int i = 0; i < fill.length; i++) { //VERIFICAMO LA EXISTENCIA DE UN "_ "
        if (fill[i].equals("_ ")){
            count++;
        }
    }
    System.out.println("------------------------------");
    if (Arrays.equals(fill, arraysecret) && count == 0) { // COMPARAMOS LOS ARRAY Y
        VERIFICAMOS SI SON IGUALES Y QUE NO EXISTA UN "_ "
        System.out.println("Usted a ganado");
        System.out.println("El numero de intentos : "+ intentos);
        return true;
    }else {
        return false;
    }
}
```

4.6. Ahorcado

- En el segundo commit creamos un if dentro del while con el metodo finish para que este ciclo while se finalize
- El codigo y el commit seria el siguiente:

Listing 11: Commit

```
$ git commit -m "Creamos los metodos arraynew y finish y tambien ponemos una estructura
    de control en el ciclo while que nos servira como un indicador que si se esta
    llegando a igualar estas 2 arrays para su posterior finalizacion con un break"
```

Listing 12: Las lineas de codigo de lo creado:

```
    while(contador <= 6){
        letra = ingreseLetra();
            if (letraEnPalabraSecreta(letra, palSecreta)){
                mostrarBlancosActualizados(letra,palSecreta,fill);
                if(finish(fill, palSecreta,contador)){ // ESTRUCTURA DE CONTROL USADA Y
                    TAMBIEN EL METODO CREADO QUE ES FINISH
                    break;
                }
            }else{
                System.out.println(figuras[contador]);
                if(figuras[contador] == ahor7){
                    System.out.println("Usted a perdido y el numero de intentos es : " +
                        contador); // MENSAJE DE PERDIO Y CUANTOS INTENTOS USO
                }
                contador = contador +1;
            }
    }
```

4.7. Ahorcado

- En el tercer commit creamos un metodo validateletter que nos permita saber si la letra ingresada no sea un numero y si lo es este nos retornara true y sera usado en un ciclo para que se ingrese una letra adecuada

- El codigo y el commit seria el siguiente:

Listing 13: Commit

```
$ git commit -m "Cambiamos el nombre del archivo al adecuado que es Ahorcado y tambien
    anadimos un metodo validateletter y completamos el metodo ingreseletra"
```

Listing 14: Las lineas de codigo de lo creado:

```
public static boolean validateletter(String letra){ // METODO CREADO PARA EL INGRESO DE
    UNA LETRA PARA SU COMPROBACION SI ES UN NUMERO NOS DARA TRUE Y SI NO FALSE
   try {
      Integer.parseInt(letra);
      return true;
   } catch (NumberFormatException e) {
      return false;
   }
}
```

4.8. Ahorcado

- En el tercer commit completamos el metodo ingresarletra que seria poner un ciclo while que se daria cuando la letra ingresada es mayor a 1 o esta letra sea un numero y se le pidira que ponga una letra adecuada
- El codigo y el commit seria el siguiente:

Listing 15: Commit

```
$ git commit -m "Cambiamos el nombre del archivo al adecuado que es Ahorcado y tamien
    anadimos un metodo validateletter y completamos el metodo ingreseletra"
```

Listing 16: Las lineas de codigo del completado:

```
public static String ingreseLetra(){// COMPLETANDO EL METODO
   String laLetra;
   Scanner sc = new Scanner(System.in);
   System.out.println("Ingrese letra: ");
   laLetra = sc.next();
   while(laLetra.length() != 1 || validateletter(laLetra)){
      System.out.println("No ingreso una letra especifica vuelva a ingresar una letra");
      System.out.println("Ingrese letra: "); //COMPLETAR PARA VALIDAR
      laLetra = sc.next();
   }
   return laLetra;
}
```

4.9. Estructura de laboratorio 02

- El contenido que se entrega en este laboratorio es el siguiente:

4.10. Ahorcado

- El Ejecucion seria la siguiente::

Listing 17: Las lineas de codigo del completado:

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

java
Ingrese letra:
j
j- - -
------------------------------
Ingrese letra:
w
 + - - - +
  |     |
  O     |
        |
        |
        |
=========
Ingrese letra:
q
 + - - - +
  |     |
  O     |
  |     |
        |
        |
=========
Ingrese letra:
w
 + - - - +
  |     |
  O     |
  /|     |
        |
        |
=========
Ingrese letra:
q
 + - - - +
  |     |
  O     |
  /|\    |
        |
        |
=========
Ingrese letra:
```

```
w
 + - - - +
  |       |
  O       |
  /|\     |
  /        |
          |
=========
Ingrese letra:
a
ja_ a
------------------------------
Ingrese letra:
v
java
------------------------------
Usted a ganado
El numero de intentos : 6
```

4.11. Estructura de laboratorio 02

- El contenido que se entrega en este laboratorio es el siguiente:

```
/Lab02
|----Ahorcado.class
|----Ahorcado.java
+----Latex
   |----img
          |----logo_abet.png
          |----logo_episunsa.png
          |----logo_unsa.jpg
          +----pseudocodigo_insercion.png
   |----InformeLab02.aux
   |----InformeLab02.fdb_latexmk
   |----InformeLab02.fls
   |----InformeLab02.log
   |----InformeLab02.out
   |----InformeLab02.pdf
   |----InformeLab02.synctex.gz
   |----InformeLab02.tex
   +----src
      +----Ahorcado01.java
```

5. Rúbricas

5.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

5.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos lo items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	1	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	1	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	0.5	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	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	X	2	
Total		20		12.5	

6. Referencias

- https://drive.google.com/file/d/19hjCsMtViypEj3cOPRF_cflH1r8r9vBn/view