Debug

Resolución de Katas en Java y Depuración

El propósito de esta actividad es fortalecer tus habilidades de programación en Java mediante la solución de tres problemas de katas y aprender a utilizar las herramientas de depuración en el entorno de desarrollo Eclipse.

Instrucciones

Solución de Katas

Resuelve los siguientes tres problemas (katas) en Java. Cada solución debe estar en un archivo Java independiente dentro del mismo proyecto de Eclipse:

Kata 1

<u>FizzBuzz</u>

Kata 2

Tamagotchi v1.0

Depuración en Eclipse

- Agrega **al menos dos puntos de interrupción (breakpoints)** en cada kata en las líneas del código que consideres clave para verificar la lógica.
- Utiliza el modo de depuración de Eclipse para ejecutar el programa y analizar el flujo de ejecución.
- Observa el comportamiento del programa en la consola y en las herramientas de depuración.

Entregables

1. Código Fuente:

Entrega los dos archivos Java con las soluciones a las katas, correctamente documentados y funcionales.

2. Capturas de Pantalla del Código:

Adjunta imágenes que muestren el código completo de cada kata en Eclipse.

3. Capturas de Pantalla del Proceso de Depuración:

- o Al menos dos capturas por kata que muestren los puntos de interrupción en acción.
- o Una captura debe mostrar el estado de las variables en el panel de depuración.

Criterios de Evaluación

- Soluciones correctas y funcionales para cada kata.
- Uso adecuado de puntos de interrupción y herramientas de depuración en Eclipse.
- Claridad y legibilidad del código entregado.
- Calidad de las capturas de pantalla (deben ser legibles y mostrar la información solicitada).

Fecha de Entrega: 15 de diciembre 2024

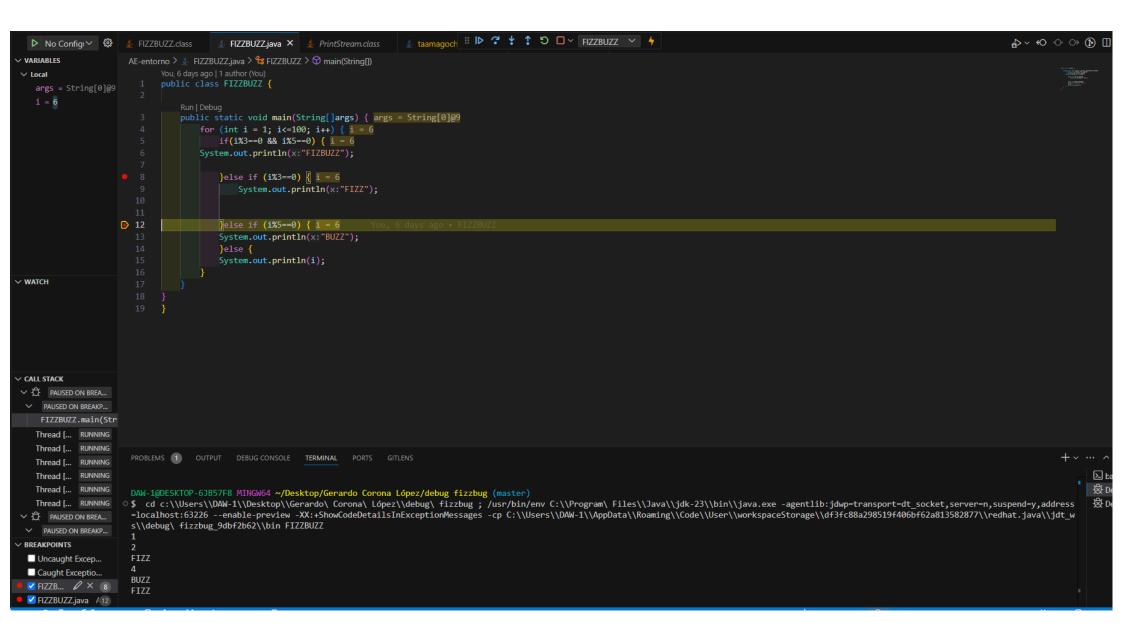
Formato de Entrega: Crea un repositorio (el repo debe ser público y tener un README explicando el contenido del proyecto) y sube los archivos de código como las capturas de pantalla y envía el enlace a la plataforma de la asignatura en Classroom.

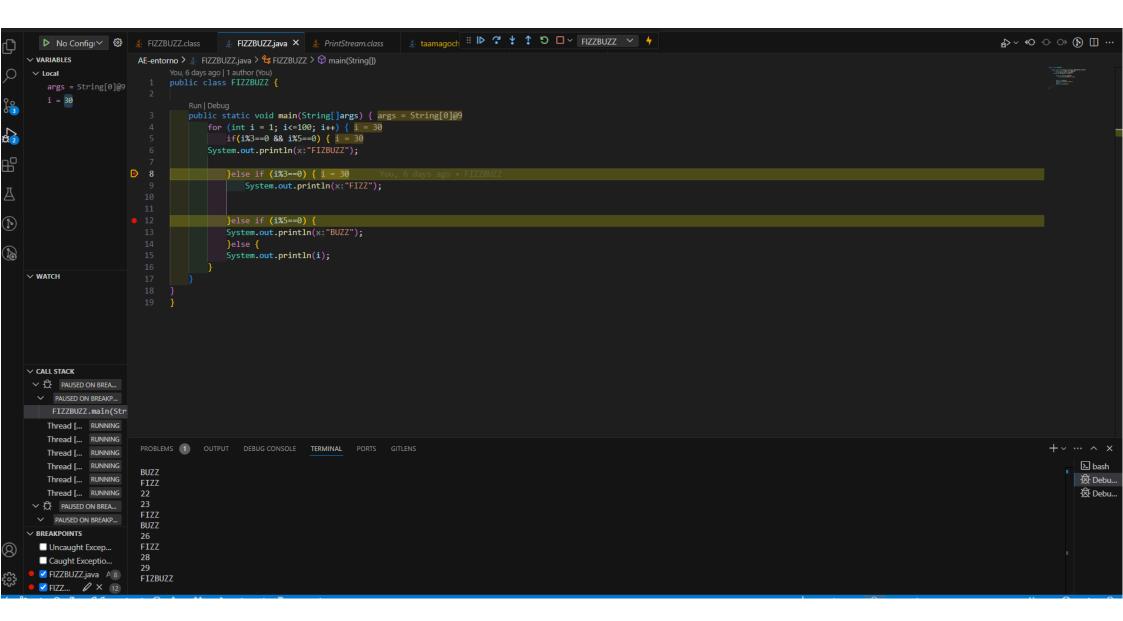
Kata 1

FizzBuzz

```
public FIZZBUZZ() { // Constructor de la clase FIZZBUZZ. Aunque no se usa en este programa, se define de
public static void main(String[] var0) { // Método principal donde comienza la ejecución del programa.
   for(int var1 = 1; var1 <= 100; ++var1) { // Bucle `for` que recorre los números del 1 al 100.
      if (var1 % 3 == 0 && var1 % 5 == 0) { // Comprueba si `var1` es divisible entre 3 Y entre 5 al mismo
         System.out.println("FIZBUZZ"); // Si la condición es verdadera, imprime "FIZBUZZ".
      } else if (var1 % 3 == 0) { // Si no es divisible entre ambos, verifica si es divisible solo entre
         System.out.println("FIZZ"); // Si es divisible entre 3, imprime "FIZZ".
      } else if (var1 \% 5 == 0)  { // Si no es divisible entre 3, verifica si es divisible solo entre 5.
         System.out.println("BUZZ"); // Si es divisible entre 5, imprime "BUZZ".
         System.out.println(var1); // Imprime el número actual (`var1`).
```

```
You, 6 days ago | 1 author (You)
     public class FIZZBUZZ {
          Run | Debug
          public static void main(String[]args) {
              for (int i = 1; i <= 100; i++) {
                  if(i%3==0 && i%5==0) {
              System.out.println(x:"FIZBUZZ");
                  }else if (i%3==0) {
                      System.out.println(x:"FIZZ");
10
11
12
                   }else if (i%5==0) {
                  System.out.println(x:"BUZZ");
13
14
                  }else {
15
                  System.out.println(i);
16
17
18
19
```





Kata 2

Tamagotchi v1.0

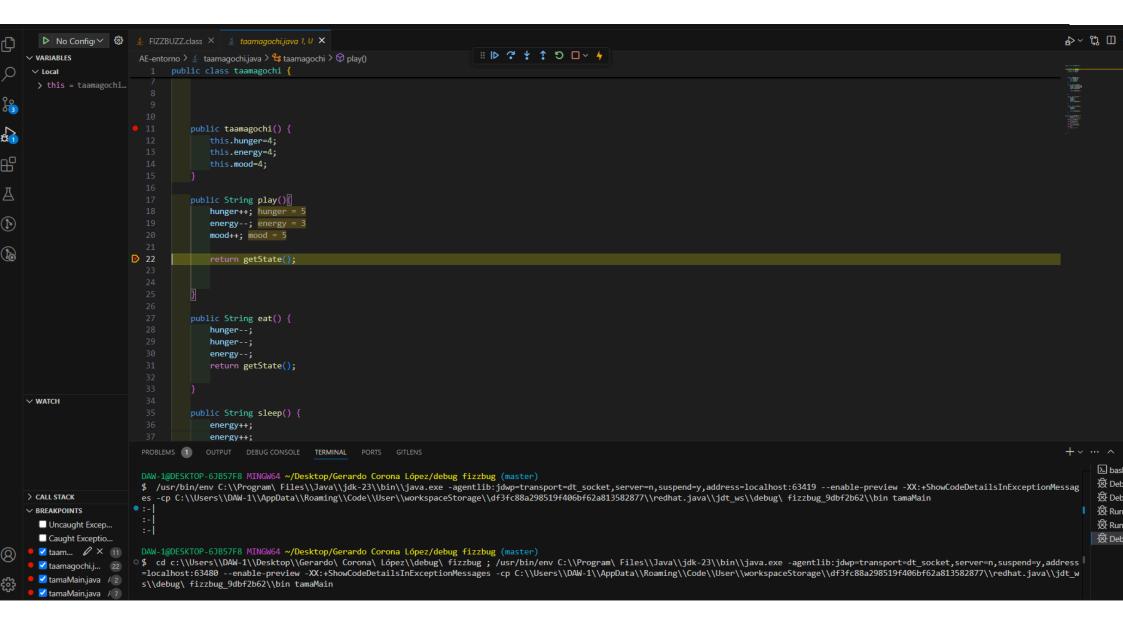
```
public class taamagochi { // Declara una clase pública llamada "taamagochi". Representa una mascota virtual.
   private int hunger; // Variable privada que mide el nivel de hambre (a mayor valor, más hambre tiene).
   private int energy; // Variable privada que mide el nivel de energía (a menor valor, más cansado está).
   public taamagochi() { // Constructor de la clase "taamagochi". Se ejecuta al crear una nueva instancia.
       this.hunger = 4; // Inicializa el nivel de hambre en 4.
       this.energy = 4; // Inicializa el nivel de energía en 4.
       this.mood = 4; // Inicializa el nivel de humor en 4.
   public String play() { // Método que simula que la mascota juega.
```

```
hunger++;
   energy--;
   mood++;
   return qetState(); // Devuelve el estado actual de la mascota después de jugar.
public String eat() { // Método que simula que la mascota come.
   hunger--;
   hunger--;
   energy--;
   return qetState(); // Devuelve el estado actual de la mascota después de comer.
public String sleep() { // Método que simula que la mascota duerme.
   energy++;
   energy++;
```

```
return qetState(); // Devuelve el estado actual de la mascota después de dormir.
public String getState() { // Método que determina el estado actual de la mascota basado en sus atributos.
   return "o o"; // Devuelve este estado para indicar disgusto o tristeza.
   } else if (energy < 2) { // Si la energía es menor a 2, la mascota está cansada.
```

```
public class tamaMain{
   public static void main(String[]args){
      taamagochi tama1 = new taamagochi();

      System.out.println(tama1.play());
      System.out.println(tama1.eat());
      System.out.println(tama1.sleep());
}
```



```
▶ No Configr > ∰

♣ FIZZBUZZ.class × 
♣ FIZZBUZZ.iava M

                                                             🍨 taamagochi.java 1, U 🗙 🎍 tamaMain.java M
                                                                                                           Debug.pdf U
                                                                                               # | □ | □ | ↑ | ↑ | □ | • |

∨ VARIABLES

                       AE-entorno > ∮ taamagochi.java > ⇔ taamagochi > ۞ play()
                              public class taamagochi { // Declara una clase pública llamada "taamagochi". Representa una mascota virtual.

∨ Local

  > this = taamagochi...
                                  private int hunger; // Variable privada que mide el nivel de hambre (a mayor valor, más hambre tiene).
                                  private int energy; // Variable privada que mide el nivel de energía (a menor valor, más cansado está).
                                  private int mood; // Variable privada que mide el nivel de ánimo o humor (a mayor valor, más feliz está).
                                  public taamagochi() { // Constructor de la clase "taamagochi". Se ejecuta al crear una nueva instancia.
                                      this.hunger = 4; // Inicializa el nivel de hambre en 4.
                                      this.energy = 4; // Inicializa el nivel de energía en 4.
                                      this.mood = 4; // Inicializa el nivel de humor en 4.
                                  public String play() { // Método que simula que la mascota juega.
                                                      // Aumenta el nivel de hambre al jugar. hunger = 5
                     15
                                                      // Aumenta el nivel de humor al jugar.
                                      mood++;
                                      return getState(); // Devuelve el estado actual de la mascota después de jugar.
                                  public String eat() { // Método que simula que la mascota come.
                                                      // Disminuye el nivel de hambre.
                                      hunger--;
                                      hunger--:
                                                      // Disminuye nuevamente el nivel de hambre (la comida satisface más).
                                      energy--;
                                      return getState(); // Devuelve el estado actual de la mascota después de comer.
                                  public String sleep() { // Método que simula que la mascota duerme.
                                                      // Aumenta el nivel de energía al dormir.

∨ WATCH

                                      return getState(); // Devuelve el estado actual de la mascota después de dormir.
                        PROBLEMS (1) OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS
                                                                                                                                                                                                                                 ΣĿ
                      $ cd c:\\Users\\DAW-1\\Desktop\\Gerardo\ Corona\ López\\debug\ fizzbug ; /usr/bin/env C:\\Program\ Files\\Java\\jdk-23\\bin\\java.exe -agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address
                                                                                                                                                                                                                                 级口
                        =localhost:63807 --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\\Users\\DAW-1\\AppData\\Roaming\\Code\\User\\workspaceStorage\\df3fc88a298519f406bf62a813582877\\redhat.java\\jdt_w |
                                                                                                                                                                                                                                 级(
                        s\\debug\ fizzbug 9dbf2b62\\bin tamaMain
                                                                                                                                                                                                                                 爱
数
数
> CALL STACK
∨ BREAKPOINTS
                                                                                                                                                                                                                                 ₩.
  Uncaught Excep...
                        DAW-1@DESKTOP-6JB57F8 MINGW64 ~/Desktop/Gerardo Corona López/debug fizzbug (master)
  Caught Exceptio...
                       🗦 cd c:\\Users\\DAW-1\\Desktop\\Gerardo\ Corona\ López\\debug\ fizzbug ; /usr/bin/env C:\\Program\ Files\\Java\\jdk-23\\bin\\java.exe -agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address

✓ taamagochi.j... 15

                        =localhost:63829 --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -cp C:\\Users\\DAW-1\\AppData\\Roaming\\Code\\User\\workspaceStorage\\df3fc88a298519f406bf62a813582877\\redhat.java\\jdt_w
                        s\\debug\ fizzbug_9dbf2b62\\bin tamaMain
```