

```

1 package ejercicios;
2
3 import javafx.application.Application;
4 import javafx.scene.Scene;
5 import javafx.scene.control.*;
6 import javafx.scene.layout.FlowPane;
7 import javafx.stage.Stage;
8
9 public class App extends Application {
10     @Override
11     public void start(Stage escenario) {
12         TextField numero1 = new TextField();
13         Label labelOperacion = new Label(" + ");
14         TextField numero2 = new TextField();
15         Button botonCalcular = new Button(" = ");
16         TextField resultado = new TextField();
17         FlowPane gestorDeLayout = new FlowPane();
18         gestorDeLayout.getChildren().add(numero1);
19         gestorDeLayout.getChildren().add(labelOperacion);
20         gestorDeLayout.getChildren().add(numero2);
21         gestorDeLayout.getChildren().add(botonCalcular);
22         gestorDeLayout.getChildren().add(resultado);
23         Scene escenaPrincipal = new Scene(gestorDeLayout, 600, 50);
24         escenario.setScene(escenaPrincipal);
25         escenario.setTitle("Sumador");
26         escenario.setResizable(false);
27         escenario.show();
28         botonCalcular.setOnAction(event -> {
29             float n1 = Float.parseFloat(numero1.getText());
30             float n2 = Float.parseFloat(numero2.getText());
31             float resu = n1 + n2;
32             resultado.setText(Float.toString(resu));
33         });
34     }
35     public static void main(String[] args) {
36         launch();
37     }
38 }

```

- explíquelo y describa su funcionamiento al correrlo.
- ¿Cómo dividiría, línea a línea, las responsabilidades entre modelo, vista y controlador?
- Indique qué ventaja(s) y desventaja(s) ofrecería MVC frente a esta versión del programa.

- Se tiene un programa al cual se le ingresan dos números en los textfields de la suma, y devuelve la suma de estos dos en el tercero, al presionar el botón igual (=), con los decimales incluidos.
- Modelo es la línea 31 ya que ahí se produce la suma de los números. Vista es desde la línea 12 a la 27 ya que es donde están todos los componentes de la ventana a mostrar. También la línea 32 donde se muestra el resultado. Controlador es desde la línea 28 a la 31, donde se captan los inputs ingresados por el usuario, y al presionar el botón de igual se produce la devolución del cálculo.
- Como ventaja, podríamos destacar una mejor modularización del programa. Como desventaja, habría que elaborar funciones para que se comuniquen entre sí M, V y C

1) Dado el siguiente programa:

```

1 class Suma {
2     public void calcularSucesor() {
3         Main.pantalla.print("Ingrese n: ");
4         int numero = Main.teclado.nextInt();
5         Main.pantalla.println("El sucesor de " + numero + " es: " + (numero + 1));
6     }
7 }
8
9 public class Main {
10     public static final java.util.Scanner teclado = new java.util.Scanner(System.in);
11     public static final java.io.PrintStream pantalla = new java.io.PrintStream(System.out);
12     private static Suma suma;
13
14     public static void main(String[] args) {
15         suma = new Suma();
16         suma.calcularSucesor();
17         new Producto().duplicar();
18     }
19 }
20
21 class Producto {
22     public void duplicar() {
23         Main.pantalla.print("Ingrese n: ");
24         int numero = Main.teclado.nextInt();
25         Main.pantalla.println("El doble de " + numero + " es: " + (numero * 2));
26     }
27 }

```

a) describa su estructura mediante un diagrama de clases;
b) describa su funcionamiento (en palabras o mediante un diagrama de secuencia) y su salida;
c) indique si viola algún principio de diseño y, de ser así, indique cuál(es), explique por qué y proponga una solución.

b) En este programa:

- . Se pide al usuario que ingrese un número.
- . Paso siguiente, se imprime por pantalla el sucesor de este número (por ejemplo, si se ingresa 5, se imprime el sucesor de 5 es 6)
- . De la misma manera, se imprime por pantalla el doble de ese número (Por ejemplo, el doble de 5 es 10)

c) Se viola KISS y DRY, ya que no solo se repite código entre la suma y el producto, sino que además en el caso de querer agregar otro cálculo, por ejemplo, habría que agregar otra clase. Para esto se podría definir una clase operación que contenga los cálculos a realizar.

2) Dado el siguiente programa:

```

1 import java.util.*;
2 import java.time.LocalDateTime;
3 public class Main {
4     public static void main(String[] args) {
5         Timer timer = new Timer();
6         TimerTask task = new TimerTask() {
7             @Override
8             public void run() {
9                 LocalDateTime ahora = LocalDateTime.now(); // Obtiene del S.O. la hora actual
10                 int cuando = ahora.getSecond();
11                 if (cuando != 0) System.out.print(".");
12                 else timer.cancel();
13             }
14         };
15         timer.scheduleAtFixedRate(task, 0, 1000);
16     }
17 }

```

a) explique su funcionamiento y escriba su salida por la terminal al correrlo;
b) indique la duración de su ejecución y justifique su respuesta.

a) En este código, se obtiene la hora actual del sistema operativo. De ahí obtengo los segundos solamente de esa hora obtenida. Si segundos es distinto de cero,

entonces el programa imprimirá una secuencia de puntos, caso contrario detendrá su ejecución.

- b) La duración del tiempo de la ejecución dependerá exclusivamente de los segundos obtenidos en el SO, siendo un mínimo de 0 en el caso de obtener la hora con los segundos empezamos en cero, y 59 segundos como máximo, ya que es el rango máximo de segundos entre el 0 y el 60. Esto porque en el segundo 1, pasan 0 segundos en el timer, y en el segundo 2 ahí si recién pasa un segundo, y así hasta el 59, y que el segundo final se convierta en 0 por el cambio de ciclo horario.