

## Reporte Sprint #0

### Instrucciones

#### Objetivos

- Tomar decisiones sobre el proyecto de desarrollo de software SOS.
- Aprender pruebas unitarias y programación de GUI en el lenguaje de tu elección.

#### Entregables y políticas de calificación

Lean el documento “descripción del Proyecto 3S2” cuidadosamente y toma las decisiones para el desarrollo del software.

Usen el siguiente template para completar tu reporte.

#### 1. Decisiones claves para el proyecto SOS of the SOS (2 puntos)

Lenguaje de programación orientado a objetos	Java
Librería GUI (recomendable)	Swing, awt
IDE (Integrated Development Environment)	IntelliJ IDEA
Framework xUnit (JUnit for Java por ejemplo)	JUnit
Guía de estilo de programación (debe ser leído con cuidado)	<a href="https://google.github.io/styleguide/javaguide.html">https://google.github.io/styleguide/javaguide.html</a>
Sitio de alojamiento del proyecto	<a href="https://github.com/CarlosVilchez123/Proyecto-CC-3S2">https://github.com/CarlosVilchez123/Proyecto-CC-3S2</a>
Otras decisiones si procede	

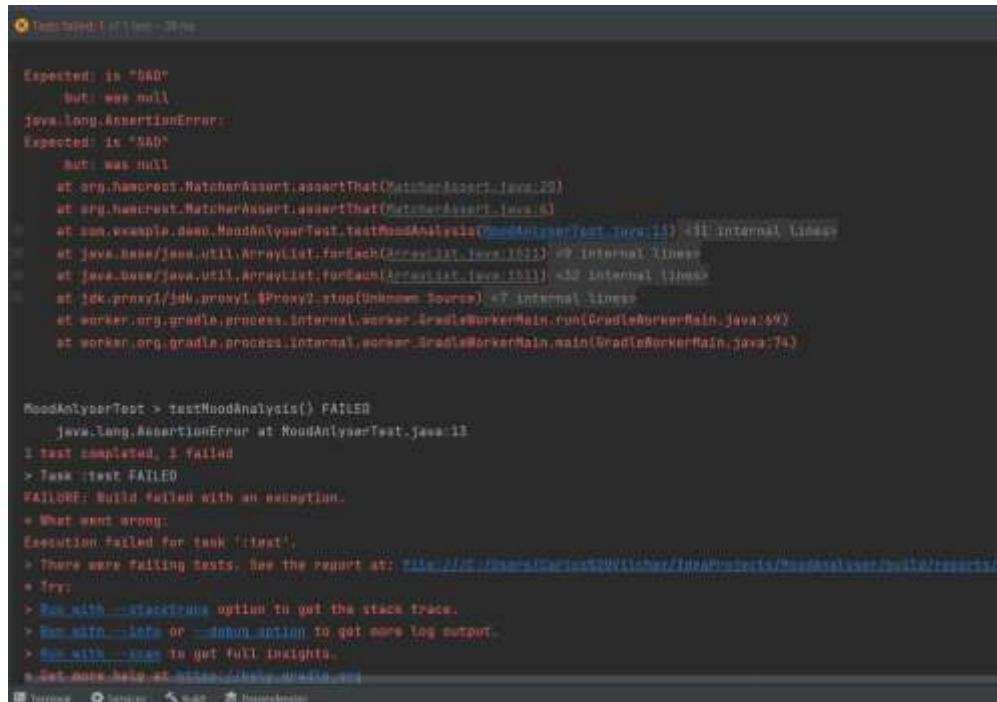
Ejemplos de guía de estilo de programación:

- Guía de estilo de Java Google: <https://google.github.io/styleguide/javaguide.html>
- Guía de estilo de C++ Google: <https://google.github.io/styleguide/cppguide.html>
- Guía de estilo Python Google: <https://google.github.io/styleguide/pyguide.html>

#### 2. Pruebas unitarias (8 puntos)

Encuentren un tutorial sobre el framework de pruebas unitarias que has elegido y escriban al menos dos pruebas xUnit de un programa que hayas escrito o encontrado en otro lugar. Adjunta aquí (1) la captura de pantalla de la ejecución de tu programa.

- Vilchez Torre Carlos Lorenzo



```

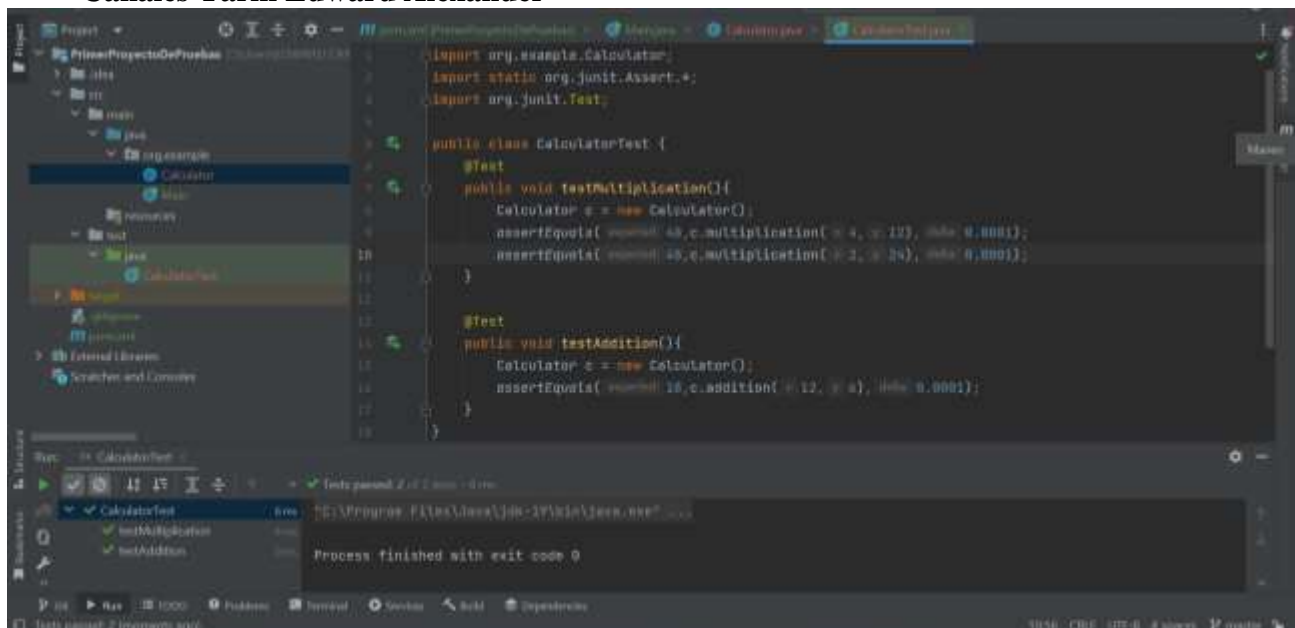
Expected: is "DAD"
but: was null
java.lang.AssertionError:
Expected: is "DAD"
but: was null
    at org.hamcrest.MatcherAssert.assertThat(MatcherAssert.java:20)
    at org.hamcrest.MatcherAssert.assertThat(MatcherAssert.java:6)
    at com.example.demo.MoodAnalyzerTest.testMoodAnalysis(MoodAnalyzerTest.java:11) ~[internal lines]
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) ~[internal lines]
    at java.base/java.util.ArrayList.forEach(ArrayList.java:1511) ~[internal lines]
    at jdk.proxy1/jdk.proxy1.$Proxy1.stop(Unknown Source) ~[internal lines]
    at worker.org.gradle.process.internal.worker.GradleWorkerMain.run(GradleWorkerMain.java:69)
    at worker.org.gradle.process.internal.worker.GradleWorkerMain.main(GradleWorkerMain.java:74)

MoodAnalyzerTest > testMoodAnalysis() FAILED
    java.lang.AssertionError at MoodAnalyzerTest.java:11
1 test completed, 1 failed
> Task :test FAILED
FAILURE: Build failed with an exception.
> What went wrong:
Execution failed for task ':test'.
> There were failing tests. See the report at: file:///C:/Users/Carlo/Desktop/Vilchez/ideaProjects/MoodAnalyzer/build/reports/tests/test/index.html
> Try:
> Run with --stacktrace option to get the stack trace.
> Run with --info or --debug option to get more log output.
> Run with --scan to get full insights.
> Get more help at https://help.gradle.org

```

Captura del Test (Test no pasa)

- Canales Yarin Edward Alexander



```

import org.example.Calculator;
import static org.junit.Assert.*;
import org.junit.Test;

public class CalculatorTest {

    @Test
    public void testMultiplication() {
        Calculator c = new Calculator();
        assertEquals("wrong", 48, c.multiplication( 4, 12), 0.0001);
        assertEquals("wrong", 48, c.multiplication( 2, 24), 0.0001);
    }

    @Test
    public void testAddition() {
        Calculator c = new Calculator();
        assertEquals("wrong", 16, c.addition( 4, 12), 0.0001);
    }
}

```

```

Run: CalculatorTest
Tests passed: 2 of 2 tests - 0 ms
Process finished with exit code 0

```

Captura del Test (Test pasa)

### 3. Programación GUI (10 puntos)

Escriban un programa GUI en el lenguaje que hayas elegido para tu proyecto SOS. La GUI de tu programa debe incluir texto, líneas, una casilla de verificación y botones de opción. Si bien se recomienda considerar la GUI para el tablero de juego SOS, no es obligatorio. En esta tarea, cualquier programa GUI de tu propio trabajo es aceptable.

Adjunten aquí (1) la captura de pantalla de la ejecución de tu programa y (2) el código fuente de tu programa.

```
package org.example;
import javax.swing.*;

public class GUI extends JFrame{
    1 usage
    public GUI(){
        initComponents();
    }
    1 usage

    private void initComponents()
    {
        JButton Salir= new JButton();
        JButton Entrar= new JButton();
        JLabel Nombre = new JLabel();
        JLabel Apellido= new JLabel();
        JTextField TxtName= new JTextField();
        JTextField TxtApellido= new JTextField();
        setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        Salir.setText("Salir");
        Entrar.setText("Entrar");
        Nombre.setText("Nombre");
        Apellido.setText("Apellido");
        TxtName.setText("");
        TxtApellido.setText("");
    }
}
```

```

GroupLayout layout = new GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(18, 18, 18)
            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.TRAILING)
                .addComponent(Salir)
                .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED, 30, Short.MAX_VALUE)
                .addComponent(Entrar))
            .addGroup(layout.createSequentialGroup()
                .addGap(27, 27, 27)
                .addGroup(layout.createParallelGroup(GroupLayout.Alignment.LEADING)
                    .addComponent(Nombre)
                    .addComponent(Apellido))
                .addGap(27, 27, 27)
                .addGroup(layout.createParallelGroup(GroupLayout.Alignment.LEADING)
                    .addComponent(TxtNombre)
                    .addComponent(TxtApellido))))
        .addGap(17, 17, 17))
);

```

```

layout.setVerticalGroup(
    layout.createParallelGroup(GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(29, 29, 29)
            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                .addComponent(Nombre)
                .addComponent(TxtNombre, GroupLayout.PREFERRED_SIZE, GroupLayout.DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE))
            .addGap(22, 22, 22)
            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                .addComponent(Apellido)
                .addComponent(TxtApellido, GroupLayout.PREFERRED_SIZE, GroupLayout.DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(LayoutStyle.ComponentPlacement.RELATED, 65, Short.MAX_VALUE)
            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                .addComponent(Salir)
                .addComponent(Entrar))
            .addGap(17, 17, 17))
);

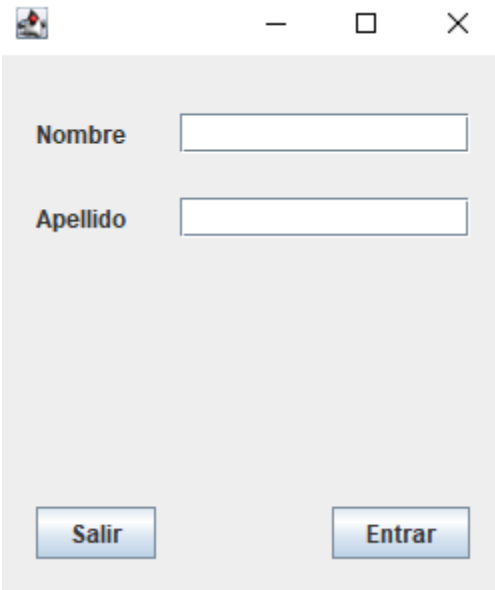
```

```

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new GUI().setVisible(true);
        }
    });
}

```

Codigo



A simple graphical user interface (GUI) window. The window has a title bar at the top with a small icon on the left and three control buttons (minimize, maximize, and close) on the right. The main area of the window is light gray. It contains two labels, "Nombre" and "Apellido", each followed by a white rectangular input field. At the bottom of the window, there are two buttons: "Salir" on the left and "Entrar" on the right. Both buttons have a blue gradient and a black border.

Nombre

Apellido

Salir Entrar

GUI