

Unidad Profesional Interdisciplinaria de Ingeniería y

Ciencias Sociales y Administrativas

**Alumnos**

Gonzaléz Martinez Daniel

González Zarate Miguel Ángel

Sanchez Ramirez Brian Alexis

**Fecha:** 20/03/2024

**Grupo:** 5NV71

**Materia:** Ingeniería de Pruebas

**Profesor:** Cruz Martínez Ramón

**Plan De Pruebas - Calculadora (Versión Final)**



# Índice

[**Índice 2**](#_pjbzlha6r1df)

[**Análisis 3**](#_6kazoufjfrys)

[Objetivo 3](#_c5g9toh5l36q)

[Misión 3](#_77lv769vx1ls)

[Visión 3](#_14d6ijio2hdu)

[Alcance 3](#_lebitpurgmy2)

[**Características del Software 4**](#_p2x0gq43eh7a)

[**Estrategia de Pruebas 5**](#_35sop94ht8kr)

[Diagrama de flujo 5](#_qnt0st5i9v6)

[Diagrama de Casos de Uso 7](#_kxeaym521nka)

[**Recursos Necesarios 7**](#_x8kf1ct318u7)

[Requisitos Funcionales 8](#_7apz6r9hoorj)

[Requisitos no Funcionales 8](#_1vfgh4h972rs)

[Cronograma 9](#_38surs3m690y)

[Criterios de Entrada y Salida 9](#_r2yrk3qm1dga)

[Suma 10](#_vf1l8k7pw1sc)

[Resta 10](#_ycp4x9ggtg9v)

[Multiplicación 10](#_wfvk5eubvjol)

[División 11](#_hgnxab2wic1s)

[Módulo 11](#_rw5mfm66nz6m)

[**Informe de Pruebas V1.0 y V2.0 12**](#_32dlnybnwez)

[**Informe de Pruebas V3.0 16**](#_kzvu1tm8z5vf)

[**Informe de Pruebas Versión Final V4.0 20**](#_4o5zm339uvv4)

[**Ejercicio de Tester 25**](#_dwnotwansbpg)

[**Conclusión 29**](#_5qqtp53sbh8)

[**Referencias Bibliográficas 29**](#_ks3yhetie6h9)

# 

# **Análisis**

## Objetivo

Se busca realizar una calculadora la cual tenga las funciones básicas (suma, resta, multiplicación, división) además del apartado módulo y limpieza de pantalla, cumpliendo las pruebas de funcionalidad.

## Misión

Nuestro programa realizará las funciones establecidas de manera correcta y cualquier usuario que la utilice no tendrá dificultades al utilizarla.

## Visión

Se desea lograr un completo análisis para llevar un control del programa para prevenir problemáticas o bien en un plazo hacerle mejoras para un mejor trabajo.

## Alcance

Este programa comenzará con las necesidades básicas que debe realizar una calculadora, se plantea hacerle mejoras en caso de ser necesarias en un plazo mediano de tiempo.

# **Características del Software**

El lenguaje del programa será C# el cual es un lenguaje de programación orientado a objetos, creado por Microsoft para el desarrollo de aplicaciones en la plataforma .NET.

**Algunas ventajas al utilizar este lenguaje son:**

* Facilidad de aprendizaje.
* Seguridad: proporciona características que ayudan a proteger de vulnerabilidades.
* Permite realizar código rápido y reducción de errores y bugs.
* Es un lenguaje multiplataforma.
* Se cuenta con una comunidad la cual te puede ayudar en caso de dudas.
* Es un lenguaje de muy alta demanda, además de ser rentable provocando así un proyecto exitoso.

# 

# 

# 

# 

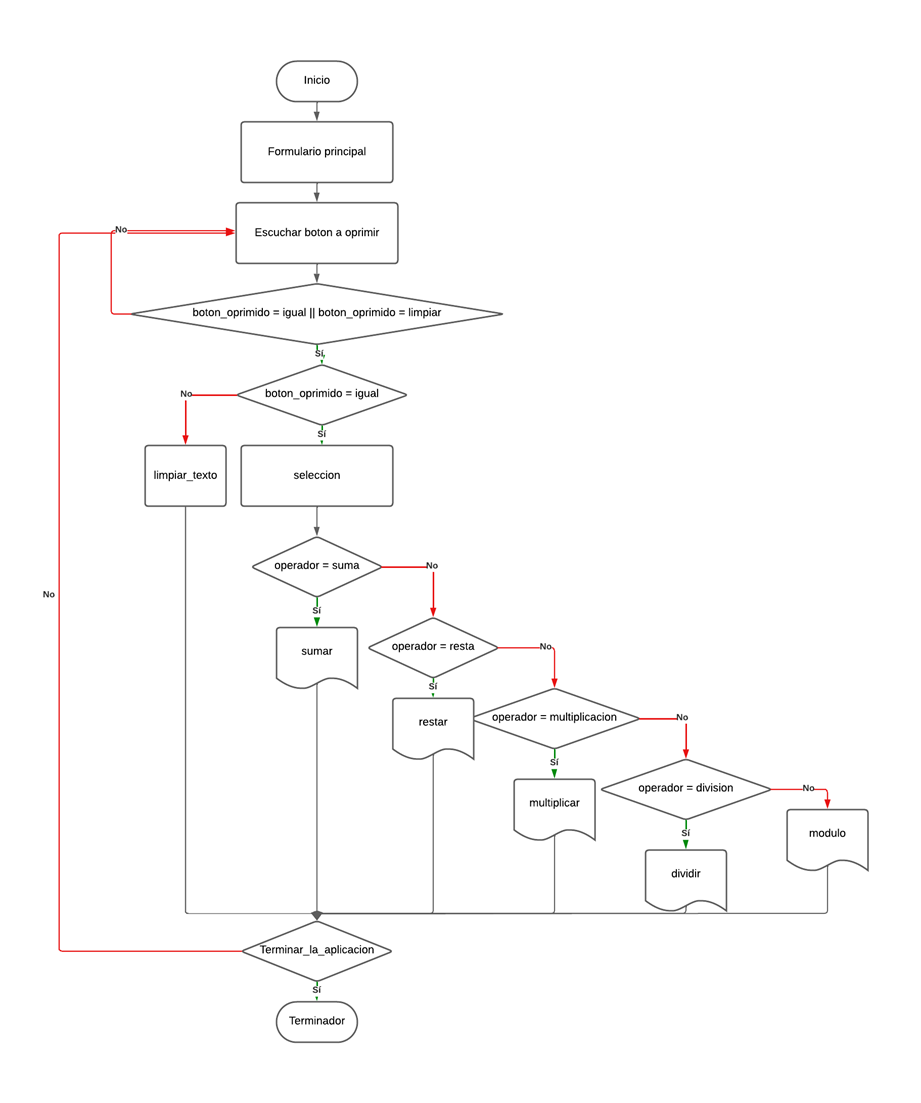
# 

# **Estrategia de Pruebas**

Después de un análisis, se optó por el uso de los siguientes apoyos para una mejor administración:

## Diagrama de flujo

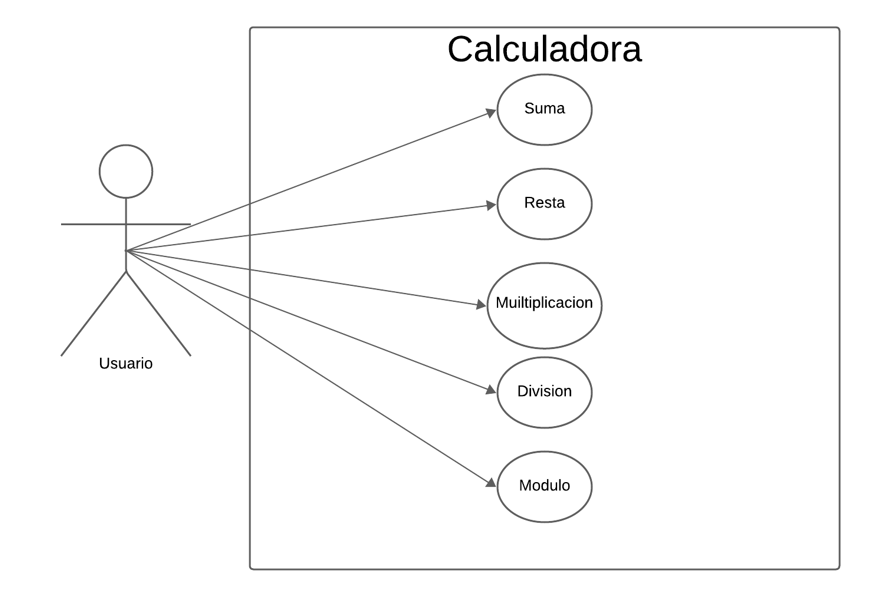
Es una representación gráfica y secuencial de un proceso o flujo de trabajo con todas las tareas y actividades principales necesarias para lograr un objetivo común. Es de gran importancia destacar los procesos que deseamos realizar para ver si nuestro algoritmo tiene coherencia además que al hora de programar facilita mucho tener el diagrama de flujo de guía (Diagrama 1.1).



*Diagrama 1.1*

## Diagrama de Casos de Uso

Define una secuencia de acciones que da lugar a un resultado de valor observable. Por lo tanto, el uso de este diagrama nos permite visualizar aquellas acciones que pueden hacer los involucrados en el programa de la calculadora para un mejor comprendimiento de lo que se realiza al ejecutarse (Diagrama 1.2).



*Diagrama 1.2*

# **Recursos Necesarios**

El equipo se planteó primero las necesidades y requerimientos que deseamos que cumpla nuestro programa, teniendo en cuenta podemos continuar con la elaboración del diagrama de flujo el cual nos permitirá tener una mejor observación del algoritmo que se está elaborando además de visualizar el planteamiento que se está llevando.

Teniendo el diagrama de flujo realizado y comprendido por los integrantes, el diagrama de casos de uso nos permitirá asemejar aquellas acciones que se desean realizar con el programa y bien ver posibles dificultades que se pueden mejorar o bien, para una mejor comprensión del usuario destinado a utilizar el programa.

## Requisitos Funcionales

1. **Suma:** Realizar una relación de adicción entre 2 números.
2. **Resta:** Realizar una relación de sustracción entre 2 números.
3. **Multiplicación:** Realizar una relación de adiciones sucesivas entre dos números.
4. **División:** Realizar una relación de sustracción de manera sucesiva entre 2 números.
5. **Módulo:** Encontrar los sobrantes al realizar una sustracción de manera sucesiva entre dos números.
6. Evitar que el usuario, pueda realizar una división entre 0.
7. Evitar que el usuario realice o coloque doble punto decimal.

## Requisitos no Funcionales

1. Mantener la limitación del usuario al solo permitir que pueda realizar la entrada de datos con los botones de la calculadora, además de evitar que este no pueda agregar números o caracteres directamente al combo box.
2. Acumular los resultados de las operaciones, además de permitir que el valor de un número pueda ser negativo o positivo (sin importar la relación).
3. Un botón de limpieza para el cambio de valores.
4. Código legible y entendible para su constante mantenimiento

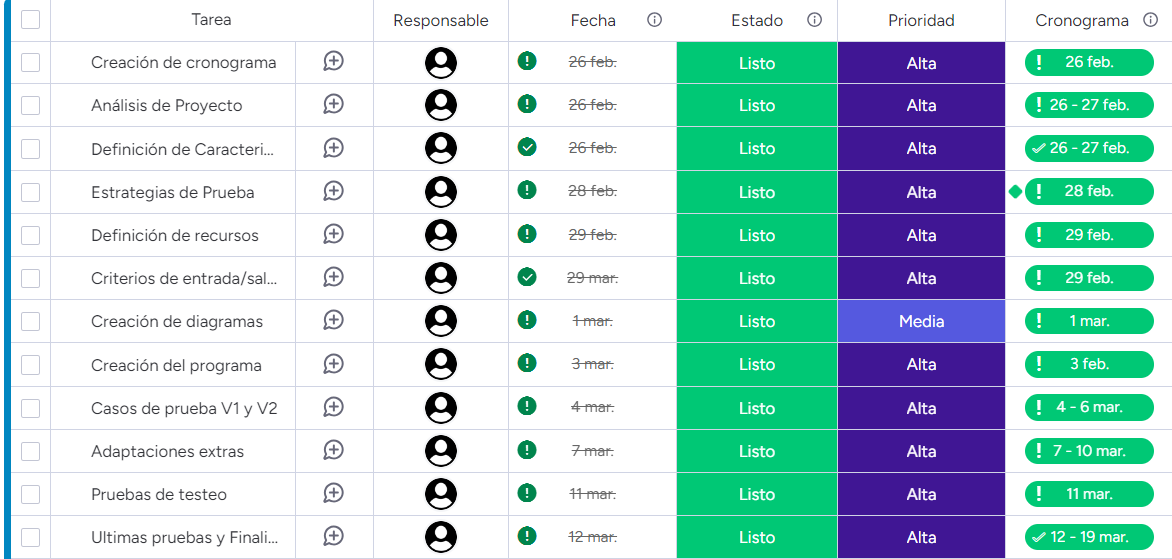
## 

## 

## 

## 

## Cronograma

****

## Criterios de Entrada y Salida

Se tendrán que plantear las variables de entrada las cuales fueron definidas de la siguiente manera:

| **private** string Num1 = string.**Empty** **private** string Num2 = string.**Empty** **private** string Simbolo = string.**Empty** **private** string operador = string.**Empty** **private** string simbolonum2 = string.**Empty**  **private** int i = 1 **private** int j = 1 **private** int k = 1 |
| --- |

Para posteriormente continuar con el apartado de las operaciones básicas de nuestra calculadora las cuales son suma, resta, multiplicación y división, con la ayuda de un if dependiendo del botón que haya presionado el usuario se mostrará la operación predeterminada:

### Suma

Se compara el operador con el símbolo “+” para continuar con la operación suma.

| **if** (operador == "+") {  txtResultado.Text = objet1.Suma(Num1,Num2);  Num1 = txtResultado.Text;  Num2 = string.Empty;  simbolo = string.Empty;  simbolonum2 = string.Empty;  operador = string.Empty; } |
| --- |

### Resta

Se compara el operador con el símbolo “-” para continuar con la operación resta.

| **if** (operador == "-") {  txtResultado.Text = objet1.Resta(Num1, Num2);  Num1 = txtResultado.Text;  Num2 = string.Empty;  simbolo = string.Empty;  simbolonum2 = string.Empty;  operador = string.Empty; } |
| --- |

### Multiplicación

Se compara el operador con el símbolo “ \* ” para continuar con la operación multiplicación.

| **if** (operador == "\*") {  txtResultado.Text = objet1.Multiplicacion(Num1, Num2);  Num1 = txtResultado.Text;  Num2 = string.Empty;  simbolo = string.Empty;  simbolonum2 = string.Empty;  operador = string.Empty; } |
| --- |

### División

Se compara el operador con el símbolo “/ ” para continuar con la operación división.

| **if** (operador == "/") {  txtResultado.Text = objet1.Division(Num1, Num2);  Num1 = txtResultado.Text;  Num2 = string.Empty;  simbolo = string.Empty;  simbolonum2 = string.Empty;  operador = string.Empty; } |
| --- |

### Módulo

De igual manera, para el uso del módulo se estaría utilizando el símbolo “ % “ dando como resultado.

| **if** (operador == "%") {  txtResultado.Text = objet1.Modulo(Num1, Num2);  Num1 = txtResultado.Text;  Num2 = string.Empty;  simbolo = string.Empty;  simbolonum2 = string.Empty;  operador = string.Empty; } |
| --- |

# **Informe de Pruebas V1.0 y V2.0**

**Proyecto:** Calculadora gráfica

**Fecha del Informe:** 04/03/2024

**Equipo de Pruebas:**

* Gonzaléz Martinez Daniel
* González Zarate Miguel Ángel
* Sanchez Ramirez Brian Alexis

**Versión del software:** v1.0 y v2.0

Matriz de Casos de Prueba

| **Tipo de prueba** | **Descripción** | **Pasos** | **Resultado esperado** | **Estatus** |
| --- | --- | --- | --- | --- |
| Funcionalidad | Ingresar la expresión  1111111111111111111111111 + 2222222222222222222222222 | Ingresar un número mayor a 20 dígitos sumado a otro de 20 dígitos | Obtener como resultado  3333333333333333333333333 | **Pass** |
| Funcionalidad | Ingresar dos signos de suma | Ingresar la suma 2++2 | Efectuar la suma correctamente | **Fail** |
| Funcionalidad | Ingresar dos signos de menos | Ingresar la suma 2 - - 2 | Efectuar la multiplicación de signos correctamente | **Fail** |
| Funcionalidad | Ingresar dos signos de multiplicación | Ingresar la siguiente expresión 2 \*\* 2 | Provocar un error en la sintaxis | **Fail** |
| Funcionalidad | Ingresar dos signos de división | Ingresar la siguiente expresión 2 // 2 | Provocar un error en la sintaxis | **Fail** |
| Funcionalidad | Ingresar una operación que requiera jerarquía de operaciones | Ingresar la expresión 1 + 2 \* 3 | Obtener 7 como resultado | **Fail** |
| Funcionalidad | Probar que la calculadora sea multi argumento | Ingresar la expresión 1 + 2 + 3 + 4 + 5 - 6 | Obtener 9 como resultado | **Fail** |
| Funcionalidad | Probar que el resultado se guarde correctamente | Ingresar la expresión 1 + 2 posteriormente restar 4 al resultado | Obtener -1 como resultado | **Pass** |
| Funcionalidad | Probar que el programa no lance error por doble punto en un número | Ingresar la suma de 1..0 + 2..0 | Provocar error en la sintaxis | **Crash** |
| Funcionalidad | Probar que no acepte un numero con puntos adicionales | Ingresar …….1 | Provocar error de sintaxis o borrado de pantalla | **Pass** |
| Funcionalidad | Probar la notación científica | Ingresar 100000000000000000000 + 1 | 1E+20 | **Pass** |
| Funcionalidad | Probar la división entre 0 | Ingresar 1 / 0 | Error de sintaxis | **Pass** |
| Funcionalidad | Probar la división entre 0 y después restarle al error una unidad | Ingresar 1 / 0 y despues -1 | Error de sintaxis | **Fail** |
| Funcionalidad | Probar la división de 0 entre 0 | Ingresar 0 / 0 | Error de sintaxis | **Fail** |
| Funcionalidad | Probar falta de primer argumento en la suma | Ingresar + 2 | Error de sintaxis | **Fail** |
| Funcionalidad | Probar falta de primer argumento en la resta | Ingresar - 2 | Obtener -2 | **Pass** |
| Funcionalidad | Probar falta de primer argumento en la división | Ingresar / 2 | Error de sintaxis | **Fail** |
| Funcionalidad | Probar falta de primer argumento en la multiplicación | Ingresar \* 2 | Error de sintaxis | **Fail** |
| Funcionalidad | Probar falta de segundo argumento en la suma | Ingresar 2 + | Error de sintaxis | **Fail** |
| Funcionalidad | Probar falta de segundo argumento en la resta | Ingresar 2 - | Error de sintaxis | **Fail** |
| Funcionalidad | Probar falta de segundo argumento en la división | Ingresar 2 / | Error de sintaxis | **Fail** |
| Funcionalidad | Probar falta de segundo argumento en la multiplicación | Ingresar 2 \* | Error de sintaxis | **Fail** |
| Usabilidad | Probar la interfaz del usuario | Alargar los bordes de la calculadora | Interfaz de usuario más grande | **Fail** |

Actividades Realizadas

**Pruebas funcionales:**

* + Se crearon y ejecutaron casos de prueba para cubrir todas las funciones especificadas en los requisitos.
  + Se identificaron y documentaron 17 defectos, que se han comunicado al equipo de desarrollo.

**Pruebas de usabilidad:**

* + Se realizaron evaluaciones de usabilidad mediante la participación de usuarios beta.
  + Se recopilaron retroalimentaciones y se implementaron mejoras de usabilidad según sea necesario.

**Pruebas de rendimiento:**

* + Se realizaron pruebas de carga para evaluar el rendimiento del sistema bajo condiciones de uso intensivo.

Cobertura de Pruebas

**1. Pruebas Unitarias:** 100%

**2. Pruebas de Integración:** 100%

**3. Pruebas de Sistema:** 100%

Resultados y Conclusiones

* **Defectos Encontrados:** 17
* **Defectos Resueltos:** 0
* **Defectos Pendientes:** 17
* **Cobertura de Pruebas:** 100%

# **Informe de Pruebas V3.0**

**Proyecto:** Calculadora gráfica

**Fecha del Informe:** 05/03/2024

**Equipo de Pruebas:**

* Gonzaléz Martinez Daniel
* González Zarate Miguel Ángel
* Sanchez Ramirez Brian Alexis

**Versión del software:** v3.0

Matriz de Casos de Prueba

| **Tipo de prueba** | **Descripción** | **Pasos** | **Resultado esperado** | **Estatus** |
| --- | --- | --- | --- | --- |
| Funcionalidad | Ingresar la expresión  1111111111111111111111111 + 2222222222222222222222222 | Ingresar un número mayor a 20 dígitos sumado a otro de 20 dígitos | Obtener como resultado  3333333333333333333333333 | **Pass** |
| Funcionalidad | Ingresar dos signos de suma | Ingresar la suma 2++2 | Efectuar la suma correctamente | **Pass** |
| Funcionalidad | Ingresar dos signos de menos | Ingresar la suma 2 - - 2 | Efectuar la multiplicación de signos correctamente | **Pass** |
| Funcionalidad | Ingresar dos signos de multiplicación | Ingresar la siguiente expresión 2 \*\* 2 | Provocar un error en la sintaxis | **Pass** |
| Funcionalidad | Ingresar dos signos de división | Ingresar la siguiente expresión 2 // 2 | Provocar un error en la sintaxis | **Pass** |
| Funcionalidad | Ingresar una operación que requiera jerarquía de operaciones | Ingresar la expresión 1 + 2 \* 3 | Obtener 7 como resultado | **Pass** |
| Funcionalidad | Probar que la calculadora sea multi argumento | Ingresar la expresión 1 + 2 + 3 + 4 + 5 - 6 | Obtener 9 como resultado | **Pass** |
| Funcionalidad | Probar que el resultado se guarde correctamente | Ingresar la expresión 1 + 2 posteriormente restar 4 al resultado | Obtener -1 como resultado | **Pass** |
| Funcionalidad | Probar que el programa no lance error por doble punto en un número | Ingresar la suma de 1..0 + 2..0 | Provocar error en la sintaxis | **Pass** |
| Funcionalidad | Probar que no acepte un numero con puntos adicionales | Ingresar …….1 | Provocar error de sintaxis o borrado de pantalla | **Pass** |
| Funcionalidad | Probar la notación científica | Ingresar 100000000000000000000 + 1 | 1E+20 | **Pass** |
| Funcionalidad | Probar la división entre 0 | Ingresar 1 / 0 | Error de sintaxis | **Pass** |
| Funcionalidad | Probar la división entre 0 y después restarle al error una unidad | Ingresar 1 / 0 y despues -1 | Error de sintaxis | **Pass** |
| Funcionalidad | Probar la división de 0 entre 0 | Ingresar 0 / 0 | Error de sintaxis | **Pass** |
| Funcionalidad | Probar falta de primer argumento en la suma | Ingresar + 2 | Obtener 2 | **Pass** |
| Funcionalidad | Probar falta de primer argumento en la resta | Ingresar - 2 | Obtener -2 | **Pass** |
| Funcionalidad | Probar falta de primer argumento en la división | Ingresar / 2 | Obtener 0 | **Pass** |
| Funcionalidad | Probar falta de primer argumento en la multiplicación | Ingresar \* 2 | Obtener 0 | **Pass** |
| Funcionalidad | Probar falta de segundo argumento en la suma | Ingresar 2 + | Obtener 2 | **Pass** |
| Funcionalidad | Probar falta de segundo argumento en la resta | Ingresar 2 - | Obtener 2 | **Pass** |
| Funcionalidad | Probar falta de segundo argumento en la división | Ingresar 2 / | Error de sintaxis | **Pass** |
| Funcionalidad | Probar falta de segundo argumento en la multiplicación | Ingresar 2 \* | Obtener 0 | **Pass** |
| Usabilidad | Probar la interfaz del usuario | Alargar los bordes de la calculadora | Interfaz de usuario más grande | **Pass** |
| Funcionalidad | Probar el botón M+ | Escribir 15 | Tener 15 guardado | **Fail** |
| Funcionalidad | Probar el botón M- | Escribir 15, apretar el botón M+ y después escribir 2 y apretar el botón M- | Resultado 13 | **Fail** |
| Funcionalidad | Probar el botón MR | Escribir 15 y presionar el botón MR | Resultado 15 | **Fail** |

Actividades Realizadas

* **Pruebas funcionales:**
  + Se crearon y ejecutaron casos de prueba para cubrir todas las funciones especificadas en los requisitos.
  + Se identificaron y documentaron 10 defectos, que se han comunicado al equipo de desarrollo.
* **Pruebas de usabilidad:**
  + Se realizaron evaluaciones de usabilidad mediante la participación de usuarios beta.
  + Se recopilaron retroalimentaciones y se implementaron mejoras de usabilidad según sea necesario.
* **Pruebas de rendimiento:**
  + Se realizaron pruebas de carga para evaluar el rendimiento del sistema bajo condiciones de uso intensivo.

Cobertura de Pruebas

1. **Pruebas Unitarias:** 100%

2. **Pruebas de Integración:** 100%

3. **Pruebas de Sistema:** 100%

Resultados y Conclusiones

* **Defectos Encontrados:** 14
* **Defectos Resueltos:** 7
* **Defectos Pendientes:** 14
* **Cobertura de Pruebas:** 100%

# **Informe de Pruebas Versión Final V4.0**

**Proyecto:** Calculadora gráfica

**Fecha del Informe:** 19/03/2024

**Equipo de Pruebas:**

* Gonzaléz Martinez Daniel
* González Zarate Miguel Ángel
* Sanchez Ramirez Brian Alexis

**Versión del software:** v4.0

Matriz de Casos de Prueba

| **Tipo de prueba** | **Descripción** | **Pasos** | **Resultado esperado** | **Estatus** |
| --- | --- | --- | --- | --- |
| Funcionalidad | Ingresar la expresión  1111111111111111111111111 + 2222222222222222222222222 | Ingresar un número mayor a 20 dígitos sumado a otro de 20 dígitos | Obtener como resultado  3333333333333333333333333 | **Pass** |
| Funcionalidad | Ingresar dos signos de suma | Ingresar la suma 2++2 | Efectuar la suma correctamente | **Pass** |
| Funcionalidad | Ingresar dos signos de menos | Ingresar la suma 2 - - 2 | Efectuar la multiplicación de signos correctamente | **Pass** |
| Funcionalidad | Ingresar dos signos de multiplicación | Ingresar la siguiente expresión 2 \*\* 2 | Provocar un error en la sintaxis | **Pass** |
| Funcionalidad | Ingresar dos signos de división | Ingresar la siguiente expresión 2 // 2 | Provocar un error en la sintaxis | **Pass** |
| Funcionalidad | Ingresar una operación que requiera jerarquía de operaciones | Ingresar la expresión 1 + 2 \* 3 | Obtener 7 como resultado | **Pass** |
| Funcionalidad | Probar que la calculadora sea multi argumento | Ingresar la expresión 1 + 2 + 3 + 4 + 5 - 6 | Obtener 9 como resultado | **Pass** |
| Funcionalidad | Probar que el resultado se guarde correctamente | Ingresar la expresión 1 + 2 posteriormente restar 4 al resultado | Obtener -1 como resultado | **Pass** |
| Funcionalidad | Probar que el programa no lance error por doble punto en un número | Ingresar la suma de 1..0 + 2..0 | Provocar error en la sintaxis | **Pass** |
| Funcionalidad | Probar que no acepte un numero con puntos adicionales | Ingresar …….1 | Provocar error de sintaxis o borrado de pantalla | **Pass** |
| Funcionalidad | Probar la notación científica | Ingresar 100000000000000000000 + 1 | 1E+20 | **Pass** |
| Funcionalidad | Probar la división entre 0 | Ingresar 1 / 0 | Error de sintaxis | **Pass** |
| Funcionalidad | Probar la división entre 0 y después restarle al error una unidad | Ingresar 1 / 0 y despues -1 | Error de sintaxis | **Pass** |
| Funcionalidad | Probar la división de 0 entre 0 | Ingresar 0 / 0 | Error de sintaxis | **Pass** |
| Funcionalidad | Probar falta de primer argumento en la suma | Ingresar + 2 | Obtener 2 | **Pass** |
| Funcionalidad | Probar falta de primer argumento en la resta | Ingresar - 2 | Obtener -2 | **Pass** |
| Funcionalidad | Probar falta de primer argumento en la división | Ingresar / 2 | Obtener 0 | **Pass** |
| Funcionalidad | Probar falta de primer argumento en la multiplicación | Ingresar \* 2 | Obtener 0 | **Pass** |
| Funcionalidad | Probar falta de segundo argumento en la suma | Ingresar 2 + | Obtener 2 | **Pass** |
| Funcionalidad | Probar falta de segundo argumento en la resta | Ingresar 2 - | Obtener 2 | **Pass** |
| Funcionalidad | Probar falta de segundo argumento en la división | Ingresar 2 / | Error de sintaxis | **Pass** |
| Funcionalidad | Probar falta de segundo argumento en la multiplicación | Ingresar 2 \* | Obtener 0 | **Pass** |
| Usabilidad | Probar la interfaz del usuario | Alargar los bordes de la calculadora | Interfaz de usuario más grande | **Pass** |
| Funcionalidad | Probar el botón M+ | Escribir 15 | Tener 15 guardado | **Pass** |
| Funcionalidad | Probar el botón M- | Escribir 15, apretar el botón M+ y después escribir 2 y apretar el botón M- | Resultado 13 | **Pass** |
| Funcionalidad | Probar el botón MR | Escribir 15 y presionar el botón MR | Resultado 15 | **Pass** |

## 

**NOTA:** Al no manejar el uso de paréntesis se solicita que el usuario digite las operaciones de manera individual ejemplo

Operación: 1 + 2 + 3 + 4 + 5 - 6, al realizar operaciones individuales si nos arroja el resultado correcto el cual es 9.

Planificaciones de Prueba

Para planificar las pruebas anteriormente mencionadas, consultamos articulos y posts para observar cómo otras personas hicieron pruebas para determinado software o incluso pruebas para una calculadora como la que realizamos, obtuvimos mucha experiencia en realizar pruebas y ser más atentos a los detalles pequeños visitando páginas en todo internet, y a como realizar la tabla y con que elementos y como elaborar las descripciones y el status.

Riesgos

Probablemente estos errores de funcionalidad no representen ningún riesgo ya que no se le permite al usuario ingresar caracteres directamente del teclado, sin embargo, si es un riesgo desplegar la aplicación en el estado actual por que podría incomodar al usuario tal o cuales fallas en las que se llevaría una mala experiencia de nuestro software.

Objetivos Empresariales

Dentro de los objetivos empresariales no tenemos uno mas en especifico para este software de calculadora, más que corregir todos los errores principalmente funcionales que encontramos para poder terminar exitosamente una calculadora y agregarla a nuestro portafolio de proyectos.

Actividades Realizadas

* **Pruebas funcionales:**
  + Se crearon y ejecutaron casos de prueba para cubrir todas las funciones especificadas en los requisitos.
  + Se identificaron y documentaron 0 defectos, que se han comunicado al equipo de desarrollo.
* **Pruebas de usabilidad:**
  + Se realizaron evaluaciones de usabilidad mediante la participación de usuarios beta.
  + Se recopilaron retroalimentaciones y se implementaron mejoras de usabilidad según sea necesario.
* **Pruebas de rendimiento:**
  + Se realizaron pruebas de carga para evaluar el rendimiento del sistema bajo condiciones de uso intensivo.

Cobertura de Pruebas

**1. Pruebas Unitarias:** 100%

**2. Pruebas de Integración:** 100%

**3. Pruebas de Sistema:** 100%

Resultados y Conclusiones

* **Defectos Encontrados:** 0
* **Defectos Resueltos:** 7
* **Defectos Pendientes:** 0
* **Cobertura de Pruebas:** 100%

# **Ejercicio de Tester**

Objetivos y Alcance

Nuestro objetivo es encontrar la menor cantidad de fallas en la calculadora de

nuestros compañeros, evaluar los errores que tuvimos en el pasado en su

calculadora para posteriormente autorizar o no el despliegue a producción.

Estrategia de Pruebas

Nuestra estrategia de pruebas se basa en ingresar fallas que nuestra calculadora

tuvo en la calculadora de los compañeros, comprobando que no tengan los mismos

errores que nosotros y si los llegan a tener documentarlos.

Ambiente de Pruebas

Nuestro ambiente de pruebas es un SO Windows 11 de 64 bits con un procesador

Ryzen 3, 8gb de ram y un disco de estado sólido.

Matriz de Casos de Uso

| **Tipo de prueba** | **Descripción** | **Pasos** | **Resultado esperado** | **Resultado obtenido** | **Estatus** |
| --- | --- | --- | --- | --- | --- |
| Funcionalidad | Comprobar la división 0 entre 0 | Ingresar 0 / 0 | Error de sintaxis | NaN | **Fail** |
| Funcionalidad | Comprobar la división entre 0 | Ingresar 1 / 0 | Error de sintaxis | Símbolo de infinito | **Fail** |
| Funcionalidad | Ingresar un punto decimal | Ingresar . | Error de sintaxis | La cadena de entrada no tiene el formato correcto | **Fail** |
| Funcionalidad y usabilidad | Comprobar el manejo de números muy grandes | Ingresar 65 veces el número 1 | Error de sintaxis | Se borra el contenido de las letras en la interfaz de usuario | **Fail** |
| Funcionalidad | Comprobar si es multi argumento | Ingresar 1 + 2 + 3 | 6 | No permite varios argumentos | **Fail** |
| Funcionalidad | Comprobar la operación de suma | 1 + 2 | 3 | 3 | **Pass** |
| Funcionalidad | Comprobar la operación de resta | 1 - 2 | -1 | -1 | **Pass** |
| Funcionalidad | Comprobar la operación de multiplicación | 2 \* 2 | 4 | 4 | **Pass** |
| Funcionalidad | Comprobar la operación de división | 1 / 1 | 1 | 1 | **Pass** |
| Funcionalidad | Suma de punto flotante | Ingresar 1.0 + 1.1 | 2.1 | 2.1 | **Pass** |
| Funcionalidad | Resta de punto flotante | Ingresar 1.0 - 1.1 | -0.1 | -0.1 | **Pass** |
| Funcionalidad | Multiplicación de punto flotante | Ingresar 1.0 \* 1.1 | 1.1 | 1.1 | **Pass** |
| Funcionalidad | División de punto flotante | Ingresar 1.0 / 1.1 | 0.9 | 0.9 | **Pass** |

Procedimiento de Pruebas

**Actividades Realizadas**

1. **Pruebas funcionales:**
   1. Se crearon y ejecutaron casos de prueba para cubrir todas las funciones especificadas en los requisitos.
   2. Se identificaron y documentaron 5 de defectos, que se han comunicado al equipo de desarrollo.
2. **Pruebas de usabilidad:**
   1. Se realizaron evaluaciones de usabilidad mediante la participación de usuarios beta
   2. Se recopilaron retroalimentaciones y se implementaron mejoras de usabilidad según sea necesario.
3. **Pruebas de rendimiento**
   1. Se realizaron pruebas de carga para evaluar el rendimiento del sistema bajo condiciones de uso intensivo. Los resultados indicaron que el sistema maneja adecuadamente 1 usuario concurrentemente

Cobertura de Pruebas

1. **Pruebas unitarias:** 0

2. **Pruebas de Integración:** 10

3. **Pruebas de Sistema:** 10

Resultados y Conclusiones del Testing

**Defectos Encontrados:** 5

**Defectos Resueltos:** No aplica

**Defectos Pendientes:** 5

**Cobertura de Pruebas:** 70%

Como conclusión llegó el equipo a que no se va a desplegar esta aplicación ya que

carece de funcionalidades básicas que debería tener una calculadora, además de

manejar lenguaje técnico informático para indicar algún error imprevisto que el

usuario final no pueda entender por su falta de conocimientos en computación.

Consideramos que para ser la versión 3 de la calculadora esta muy madura en su

desarrollo y con facilidad podrá llegar a modificar algunos aspectos para poder

desplegarla en producción.

Informe de Cobertura de Código

El código se ve limpio con bastantes buenas prácticas lo que indica que los

desarrolladores se tomaron en serio el papel de desarrollar una calculadora de

calidad.

Roles y Responsabilidades

* **Gonzalez Zarate Miguel Ángel** – Creación de documentación de calculadora v3 y Tester
* **Gonzalez Martínez Daniel** – Tester
* **Sánchez Ramírez Brian Alexis** - Tester

Desarrolladores de la Aplicación y Responsabilidades

* **Santillán Pérez Jorge Alejandro** – Creación de la calculadora
* **Sánchez Garcia Alan Alejandro** – Desarrollador de diseño
* **Huerta Enríquez David** – Desarrollador de requerimiento

# **Conclusión**

Posteriormente de haber concluido este proyecto que al comienzo consideramos que era sencillo por ser la creación de una calculadora provocó que aplicaremos nuestro punto analitico de manera mas detallado para asi lograr ver posibles errores, fallas o defectos que pueden ser muy simples o bien un tanto dificiles de ver, ademas de la importancia de la documentación y la importancia de hacer testeos de un tercero para conocer otros puntos de vista y bien lograr visualizar temas que se abarcan durante el parcial en este proyecto de manera concisa.

# **Referencias Bibliográficas**

MYTASKPANEL (2020) Introducción a C#: Definición y utilidades. Recuperado desde: <https://www.mytaskpanel.com/introduccion-a-csharp/>

IBM (2021) Definición de casos de uso. Recuperado desde: <https://www.ibm.com/docs/es/engineering-lifecycle-management-suite/lifecycle-management/6.0.3?topic=requirements-defining-use-cases>

Asana (2024) ¿Qué es un diagrama de flujo y cómo hacerlo?. Recuperado desde: <https://asana.com/es/resources/what-is-a-flowchart>

Tes. (2024, 15 enero). Cómo escribir casos de prueba para software: ejemplos y tutorial. Parasoft. https://es.parasoft.com/blog/how-to-write-test-cases-for-software-examples-tutorial/

Estefafdez. (2020, 4 noviembre). 067- Pruebas Unitarias (IV): Ejemplo de la calculadora. Una QA En Apuros. https://unaqaenapuros.wordpress.com/2020/12/23/067-pruebas-unitarias-iv-ejemplo-de-la-calculadora/