2022

validación y verificación

# Ejercicio


| | |
|---|---|
| Andy Yamada | 6481 |
| Jhon Hilasaca | 6608 |
| Wellington Cabezas | 6683 |
| Jhonnathan Castillo | 6588 |




**Problemática:** Desarrollo de un programa obtenga las raíces de polinomio de grado 2

**Requisitos:**

*RF1.* *Ingresar por teclado el primer término de la ecuación.*
*RF2.* *Ingresar por teclado el segundo término de la ecuación.*
*RF3.* *Ingresar por teclado el tercer término de la ecuación.*
*RF4.* *Validar el ingreso de cada término, donde no se admite letras o caracteres especiales.*
*RF5.* *Validar el ingreso del primer término, que no sea "0"*
*RF6.* *Cada término ingresado debe ser un número real.*
*RF7.* *El Sistema debe calcular las raíces imaginarias, en caso de que existan.*
*RF8.* *El Sistema debe indicar que se calcularon las raíces imaginarias, en el caso de hacerlo*
*RF9.* *El Sistema debe calcular las raíces reales, en caso de que existan.*
*RF10.* *El Sistema debe indicar que se calcularon las raíces reales, en el caso de hacerlo.*

**Código:**

## Clase base

```
package com.mycompany.raicescuadraticas;


import java.util.Scanner;

import java.text.DecimalFormat;


public class base {

        double a;

        double b;

        double c;

        double raiz1 =0;

        double  raiz2=0;

        Scanner lectura = new Scanner (System.in);

/*

 * comprovador de numeros
```

```
 * */
public boolean comprobar_num(String a) {
        try {
                Double.parseDouble(a);
                return true;}
        catch(NumberFormatException nfe){
                System.out.println("Dato erroneo ingresar un numero permitido");
                return false;}
}
public void constructor(){
        String cadena;
        do{
        do {
                System.out.println("Ingresar valor 1 (el valor debe ser diferente de 0)");
                cadena=lectura.next();
        }while(!comprobar_num(cadena));}
        while(Double.parseDouble(cadena)==0);
         a = Double.parseDouble(cadena);
                do {
                        System.out.println("Ingresar valor 2");
                        cadena=lectura.next();
                }while(!comprobar_num(cadena));
         b = Double.parseDouble(cadena);
                do {
                        System.out.println("Ingresar valor 3");
                        cadena=lectura.next();
```

```
            }while(!comprobar_num(cadena));
            c = Double.parseDouble(cadena);
            }



public double discriminante(double a1,double b1,double c1) {
            double r;
            r= ((Math.pow(b1,2))-(4*a1*c1));
            return r;
}

public void raices(double a1,double b1,double c1) {
            double disc= discriminante(a1, b1, c1);
            if(disc<0){
                        raiz1=(b1+Math.sqrt(disc*-1))/(2*a);
                        raiz2=(b1-Math.sqrt(disc*-1))/(2*a);
            }else{
                        raiz1=(b1+Math.sqrt(disc))/(2*a);
                        raiz2=(b1-Math.sqrt(disc))/(2*a);
            }

}

public void resultado(double a1,double b1,double c1) {
```

```
        double disc= discriminante(a1, b1, c1);
        DecimalFormat decimal=new DecimalFormat ("#.####");
        if(disc<0){
                System.out.println("x1="+decimal.format(raiz1)+" i
        x2="+decimal.format(raiz2)+" i");
        }else{
                System.out.println("x1="+decimal.format(raiz1)+"
        x2="+decimal.format(raiz2));
        }
}
}
```

## Clase main

```
package com.mycompany.raicescuadraticas;

public class RaicesCuadraticas {

    public static void main(String[] args) {

                        base arte=new base();
                        arte.constructor();
                        arte.raices(arte.a, arte.b, arte.c);
                        arte.resultado(arte.a, arte.b, arte.c);
```

```
        }
}
```

**Pruebas Unitarias:**

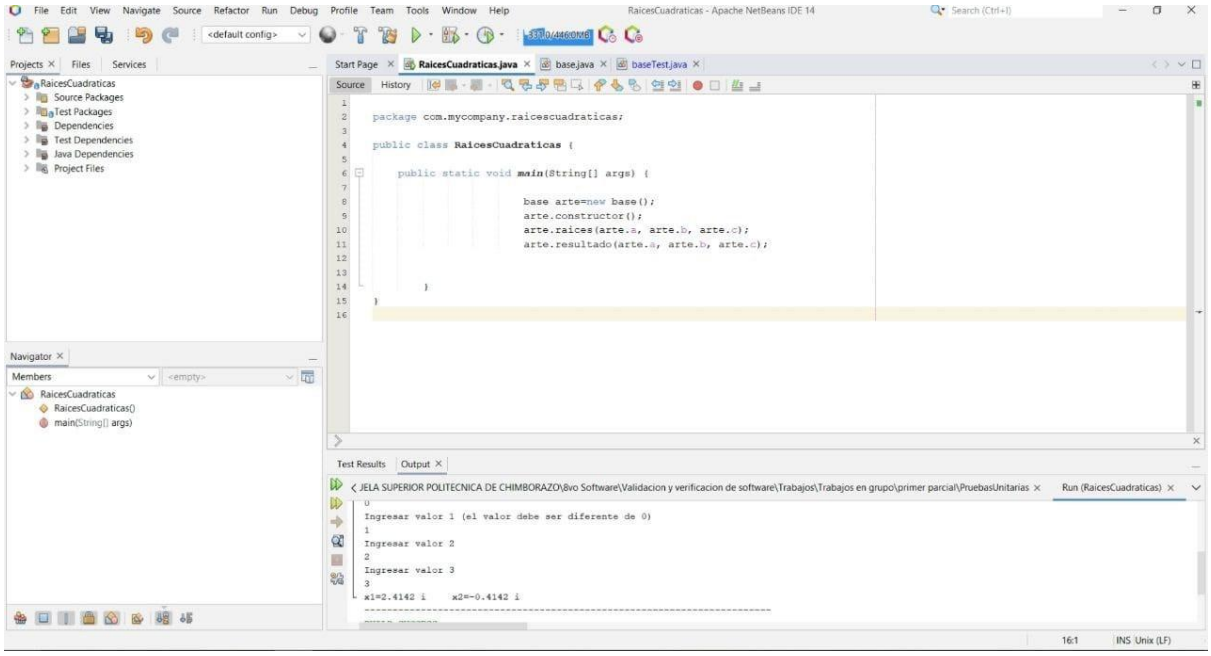
***Pruebas Unitarias comprobando RF1***

*No.casos de prueba:* 1

No.1 Ingreso del valor del primer termino

*Pruebas exitosas:* 1

*Observacion:* Al momento de la ejecucion se comprueba que el Sistema acepta los valores ingresados



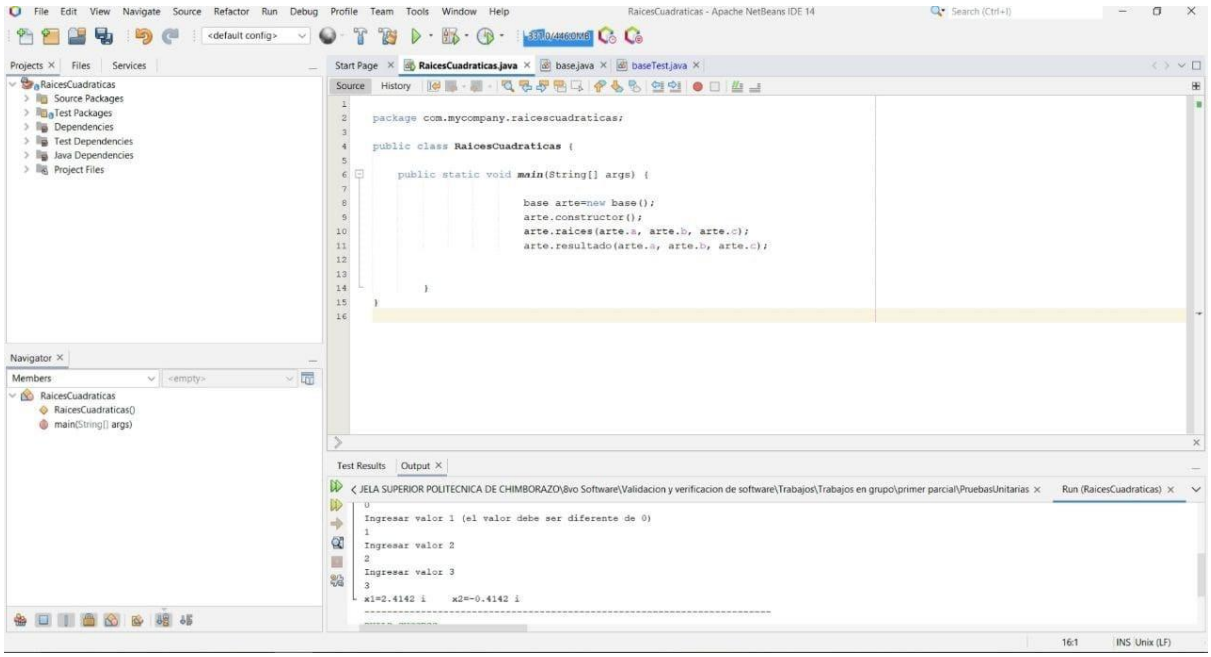

***Pruebas Unitarias comprobando RF2***

*No.casos de prueba:* 1

No.1 Ingreso del valor del segundo termino

*Pruebas exitosas:* 1

*Observacion:* Al momento de la ejecucion se comprueba que el Sistema acepta los valores ingresados



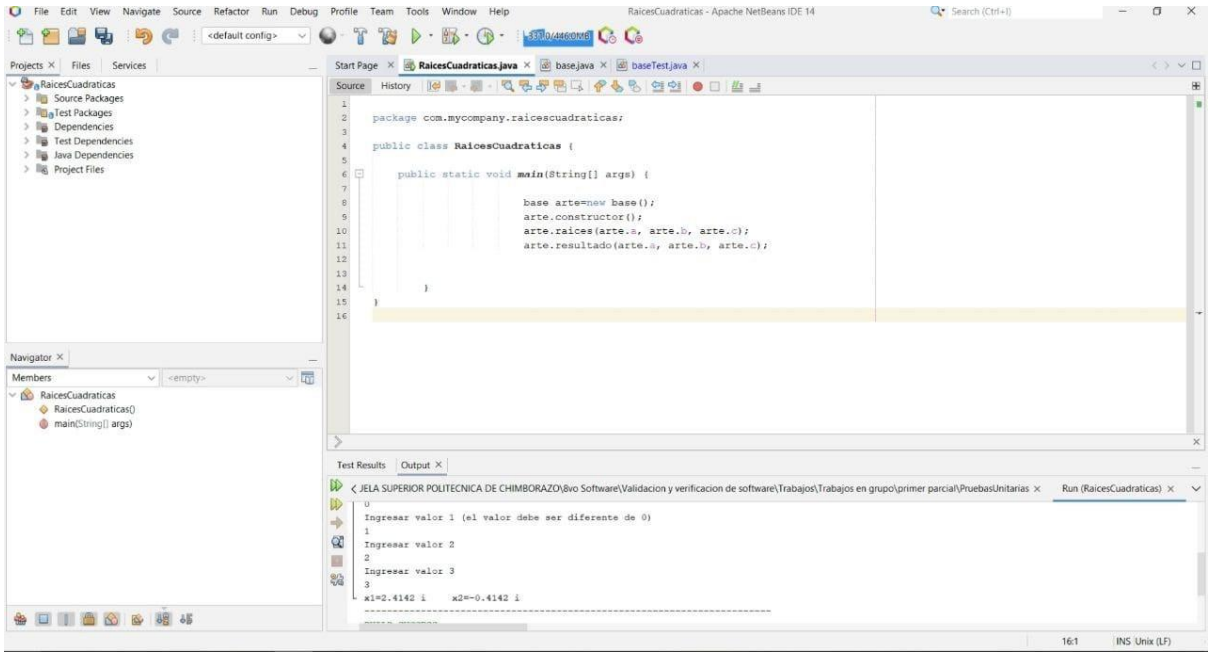

***Pruebas Unitarias comprobando RF3***

*No.casos de prueba:* 1

No.1 Ingreso del valor del tercer termino

*Pruebas exitosas:* 1

*Observacion:* Al momento de la ejecucion se comprueba que el Sistema acepta los valores ingresados



***Validacion si un valor Ingresado es número***

*No.casos de prueba:* 3

No.1 Ingreso de letras

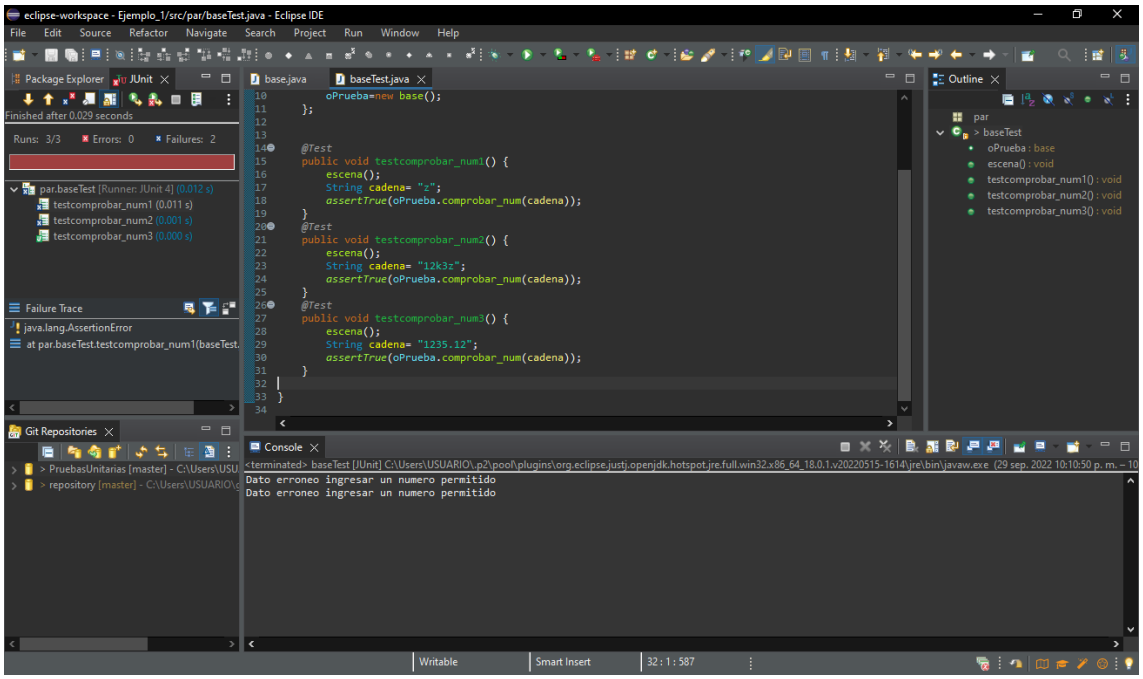
No2.Ingreso de letras y números

No3.Ingreso de unicamente numeros

*Pruebas exitosas:* 3

*Observacion:* Al momento de la ejecucion nos dio 2 errores (como se esperaba) porque los 2 primeros casos contienen letras y dichos valores no son posibles.

El ultimo caso es positivo por que solo contien numeros:



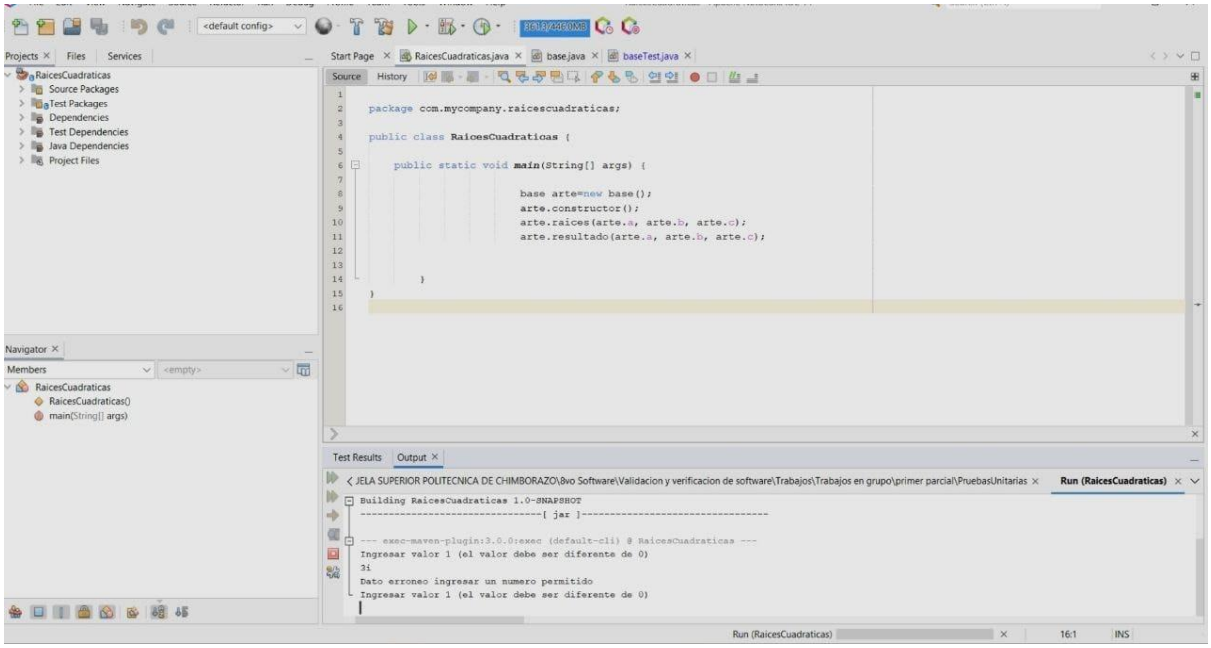

***Pruebas Unitarias comprobando RF6***

*No.casos de prueba:* 1

No.1 Ingreso del valor 0

*Pruebas exitosas:* 1

*Observacion:* Al momento de la ejecucion se nota que el sistema no se admite el valor de imaginario al momento de ingresar los valores



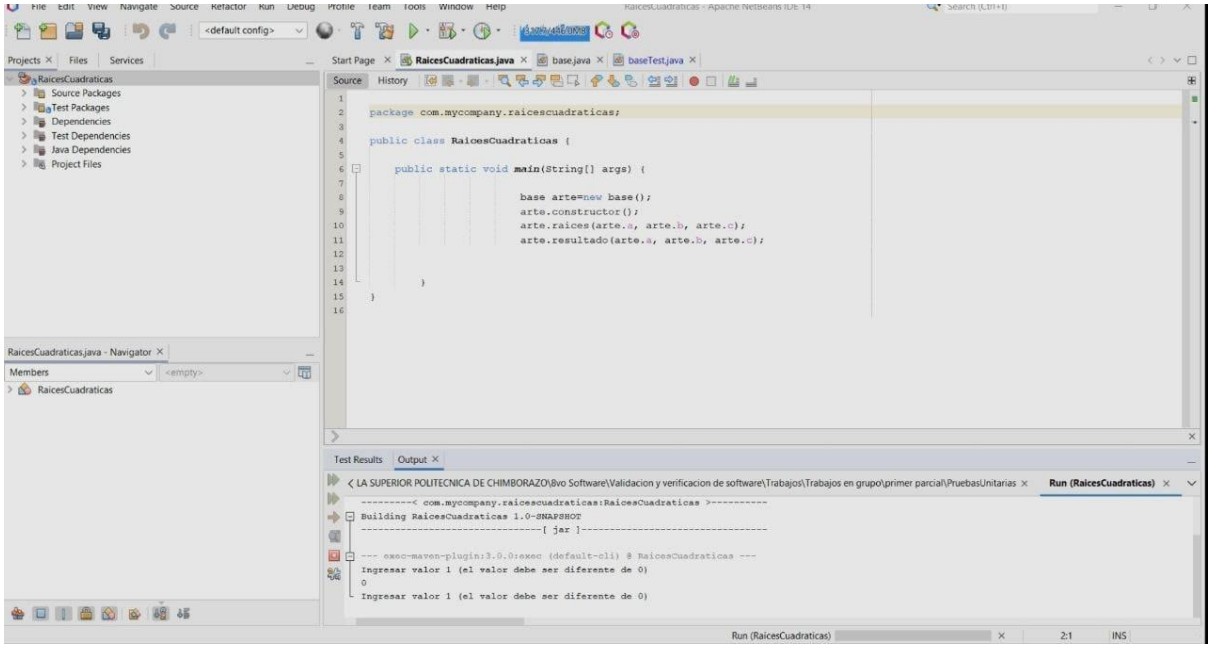

***Pruebas Unitarias comprobando RF5 (RF1.Validar el ingreso del primer término, que no sea "0")***

*No.casos de prueba:* 1

No.1 Ingreso del valor 0

*Pruebas exitosas:* 1

*Observacion:* Al momento de la ejecucion se nota que el sistema no se admit el valor de 0 en el primer valor y vuelve a solitar el primer valor




***Pruebas Unitarias comprobando el valor del discriminante***

*No.casos de prueba:* 3

No.1 Ingreso de valores para obtener un valor negativo

No2. Ingreso de valores para obtener el elemento 0

No3. Ingreso de valores para obtener un valor positivo

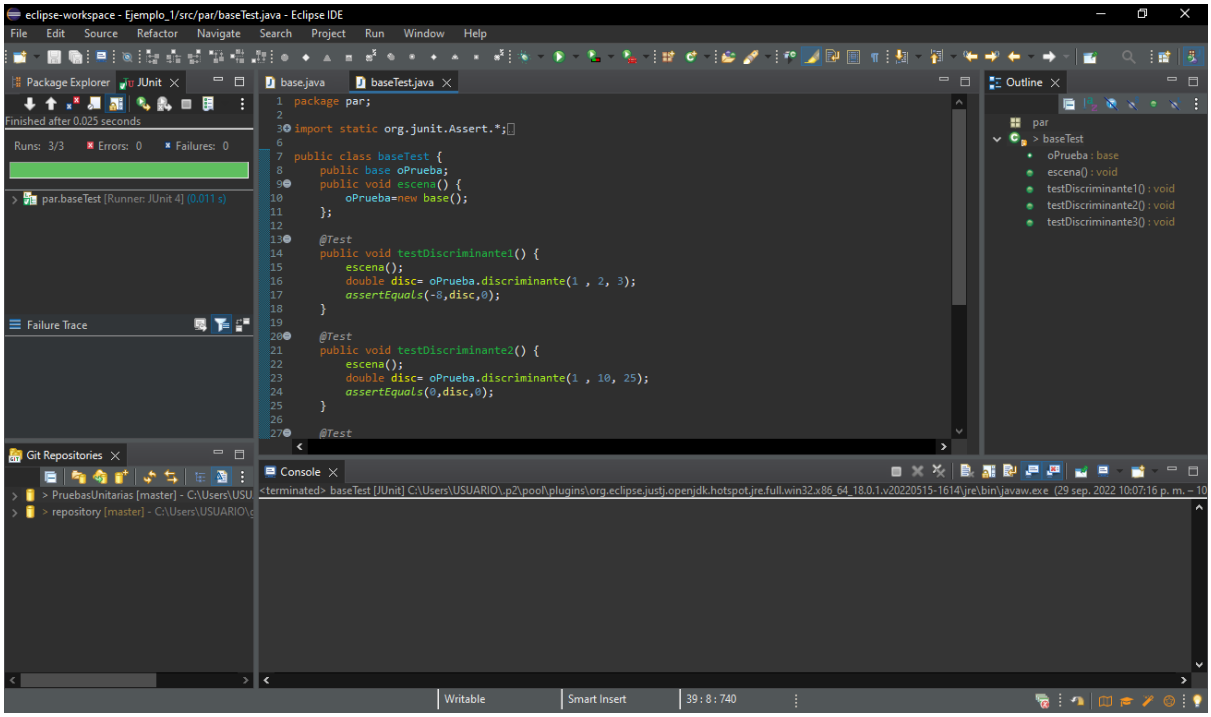
*Pruebas exitosas:* 3

*Observacion:* Al momento de la ejecucion no se ejecuto los errores porque se obserba:

cuando el discriminante debe ser negativo Resultado: Correcto

cuando el discriminante es 0 Resultado: Correcto

Cuando el discriminante debe ser positivo Resultado: Correcto




***Pruebas Unitarias comprobando el valor de la raiz***

*No.casos de prueba:* 2

No.1 Optener raices imaginarias

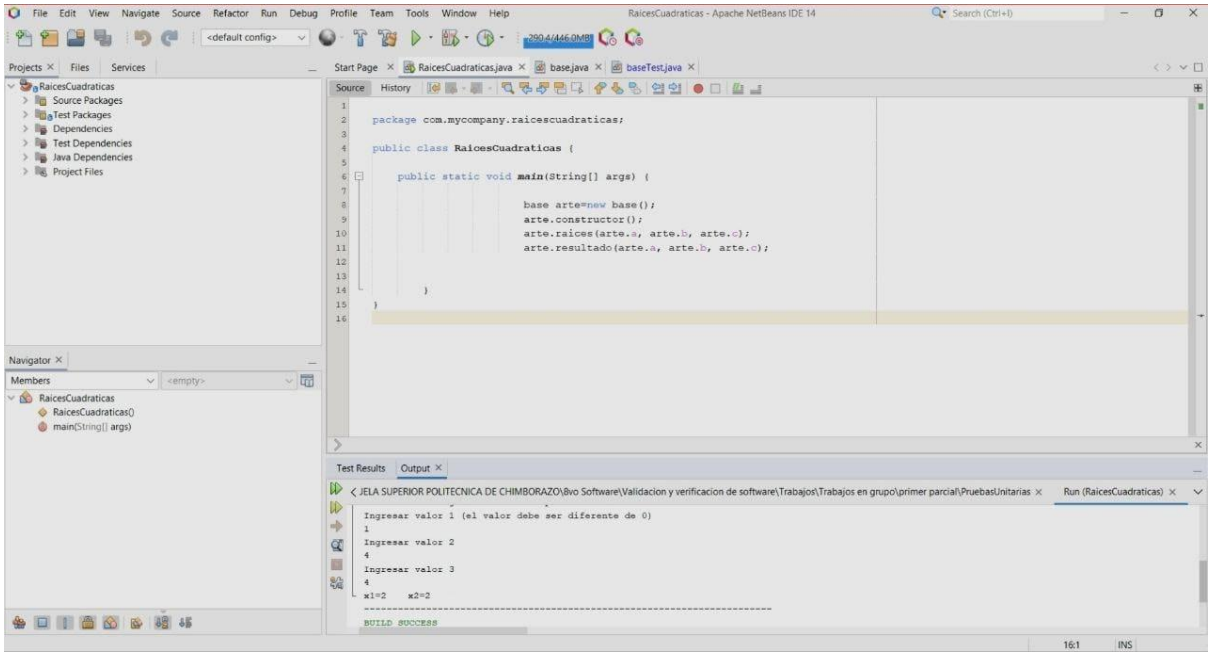
No2. Optener raices reales

*Pruebas exitosas:* 2

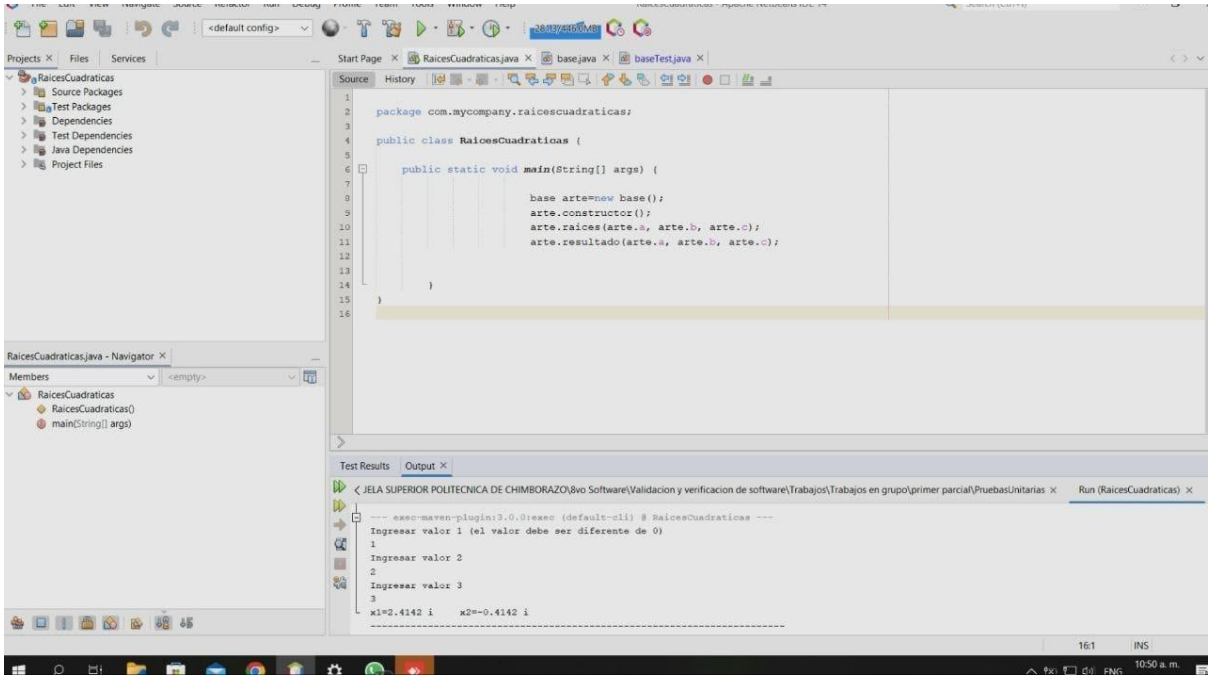
*Observacion:* Al momento de la ejecucion no se ejecuto los errores porque se obserba:

cuando las raices deben ser imaginaries Resultado: Correcto

cuando las raices deben ser reales Resultado: Correcto




*(Raices reales)*




*(Raices Imaginarias)*