

NCOM

Manual de Usuario

Materia: Matemática Superior

Grupo: Grupo Mixto_3

Año: 2019

Nombre y Apellido	Legajo	Curso
Bevilacqua, Diego	159.351-1	K3011
Salmerón, Lucila	159.488-6	K3011
Cirillo, Franco	163.618-2	K3011
Paz, Maximiliano	159.150-2	K3051
Rodríguez, Melisa	160.184-2	K3051

Índice

Manual de Usuario

Framework y lenguaje de programación

Desarrollo de las estructuras de datos y transformaciones

Menú inicial

Operaciones Básicas

Formato

Operaciones Avanzadas

Suma de Fasores

Manual de Usuario

Framework y lenguaje de programación

El framework elegido para el desarrollo de la aplicación es .NET y el lenguaje de programación es C#.

Desarrollo de las estructuras de datos y transformaciones

Durante la primera etapa del trabajo práctico **NCOM** se modeló una estructura de datos que permite representar a números complejos en forma binómica y polar, y trabajar con operaciones entre ellos.

A grandes rasgos, los avances del trabajo práctico fueron los siguientes:

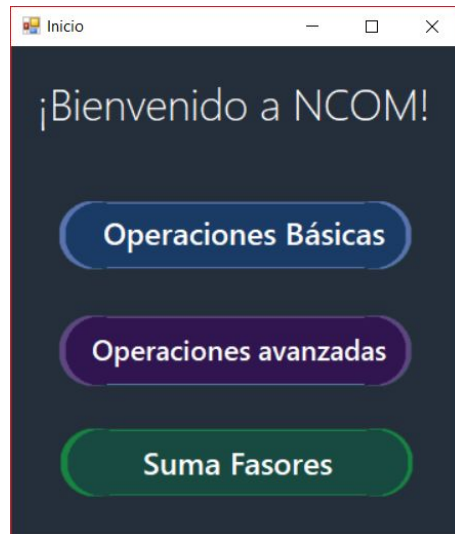
- Están desarrolladas las clases `ComplejoBinomica` y `ComplejoPolar` para poder modelar un número complejo en forma binómica y polar:

```
class ComplejoBinomica
{
    private double ParteReal { get; set; }
    private double ParteImaginaria { get; set; }
```

```
class ComplejoPolar
{
    private double Modulo { get; set; }
    private double Argumento { get; set; }
```

- Existen métodos que nos permiten hacer transformaciones del complejo de forma binómica a polar y viceversa:
 - **ComplejoBinomico::modulo()** → Calcula la raíz cuadrada de la suma entre los cuadrados de su parte real y su parte imaginaria (variables de la clase), por lo que siempre da positivo.
 - **ComplejoPolar::modulo()** → Obtiene el valor de su variable de clase 'módulo'.
 - **ComplejoBinomico::argumento()** → Calcula el arcotangente del cociente entre su parte imaginaria y su parte real. Luego, con la función auxiliar `Cuadrante()` corrige los giros.
 - **ComplejoPolar::argumento()** → Obtiene el valor de su variable de clase 'argumento'.
 - **ComplejoBinomico::pasarAPolar()** → Devuelve una nueva instancia de número `ComplejoPolar` calculando sus variables con las funciones 'módulo()' y 'argumento()'.
 - **ComplejoPolar::pasarABinomica()** → Calcula primero la parte real y la imaginaria con funciones auxiliares en las que aplica la fórmula trigonométrica del módulo por el seno o coseno del argumento (dependiendo de qué parte estamos obteniendo) y luego devuelve una nueva instancia de `ComplejoBinómico` pasándole por parámetros al constructor los datos obtenidos.

Menú inicial



En el menú inicial se podrá clicar en el botón deseado según la operación deseada.

Operaciones Básicas

Operaciones básicas

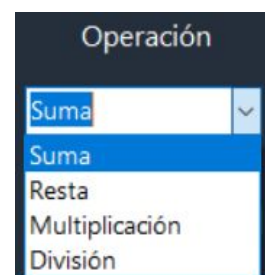
Introduzca los números con el siguiente formato:
En forma binómica: (a,b) - 'a' parte real, 'b' parte imaginaria
En forma polar: [a;b] - 'a' módulo, 'b' argumento

Número complejo	Operación	Número complejo
<input type="text" value="(-1,25 , 5)"/>	<input type="text" value="Suma"/>	<input type="text" value="[5 ; 3,14]"/>
<input type="button" value="Calcular"/>		
<input type="button" value="Atrás"/>		

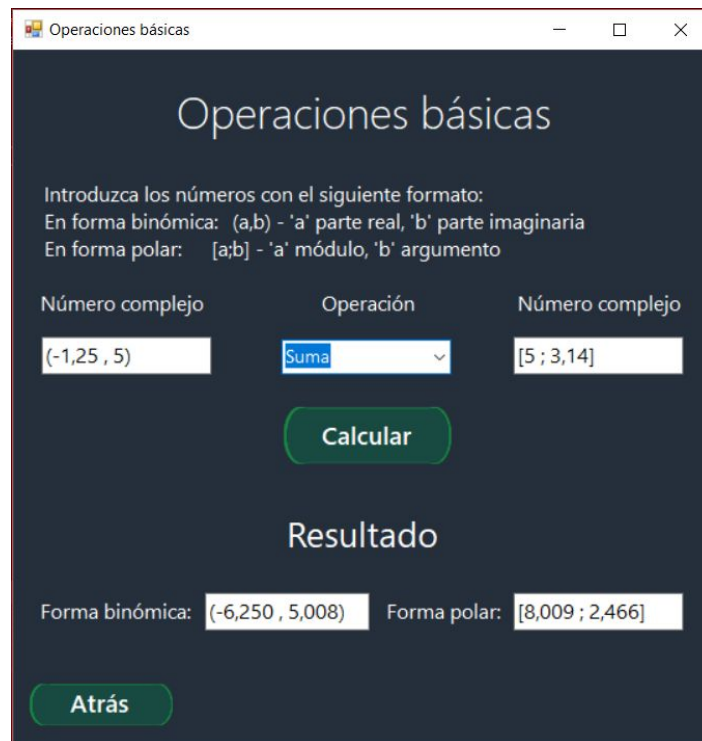
En dos campos debajo de “Número complejo” se deben introducir los operandos; uno en cada campo. Se pueden introducir en [cualquiera de sus formas](#), como se ve en la imagen.

La operación se elige desde el dropdown “Operación”, y puede ser **Suma, Resta, Multiplicación y División**.

Para obtener el resultado de la operación, se debe clicar “Calcular”.



El resultado aparece tanto en la forma binómica como en la forma polar. Para cada número aparecen los primeros 3 dígitos luego de la coma.



Se pueden modificar los operandos y la operación y volver a calcular. Se puede volver al menú inicial en cualquier momento seleccionando “Atrás”.

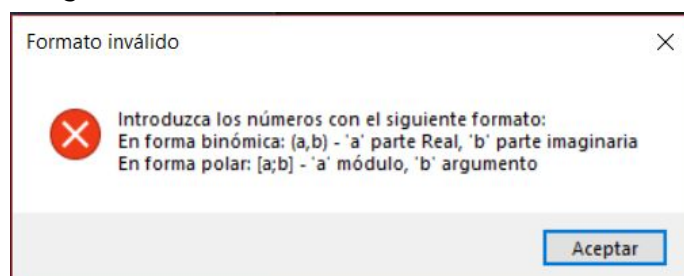
Formato

El formato de introducción y muestra de resultados de números complejos es el siguiente:

En forma binómica: **(a,b)**, siendo **a** la parte real y **b** la parte imaginaria.

En forma polar: **[a;b]**, siendo **a** el módulo y **b** el argumento.

El sistema permite la introducción de cualquier número Real, ya sea positivo, negativo o con decimales. En el caso de que se introduzca con otro formato, aparece el siguiente mensaje de error y se debe corregir los números introducidos.



Aclaración: El delimitador decimal utilizado es “,” (coma), no “.” (punto).

Operaciones Avanzadas

En esta parte se pueden calcular las potencias, raíces y raíces primitivas de cualquier número complejo. El usuario puede ingresarlo de forma binómica o polar (siempre que respete [el formato de introducción](#)) y debe elegir qué operación quiere hacer y el orden de la misma. Así, se evalúa que los datos proporcionados sean válidos y se procede a calcular el resultado.

Para ello, se crearon 3 clases estáticas: '**Potenciación**', '**Radicación**' y '**RaicesPrimitivas**'. Estas clases siempre operan de la misma manera al recibir los parámetros que necesitan y el programa las llama para calcular los resultados. Éstos se disponen en un textbox y con botones de 'Anterior' y 'Siguiente' pueden ver los demás resultados cuando son múltiples, como en el caso de las raíces y las primitivas. Estos resultados se muestran tanto en forma binómica como polar.

“Calcular”:

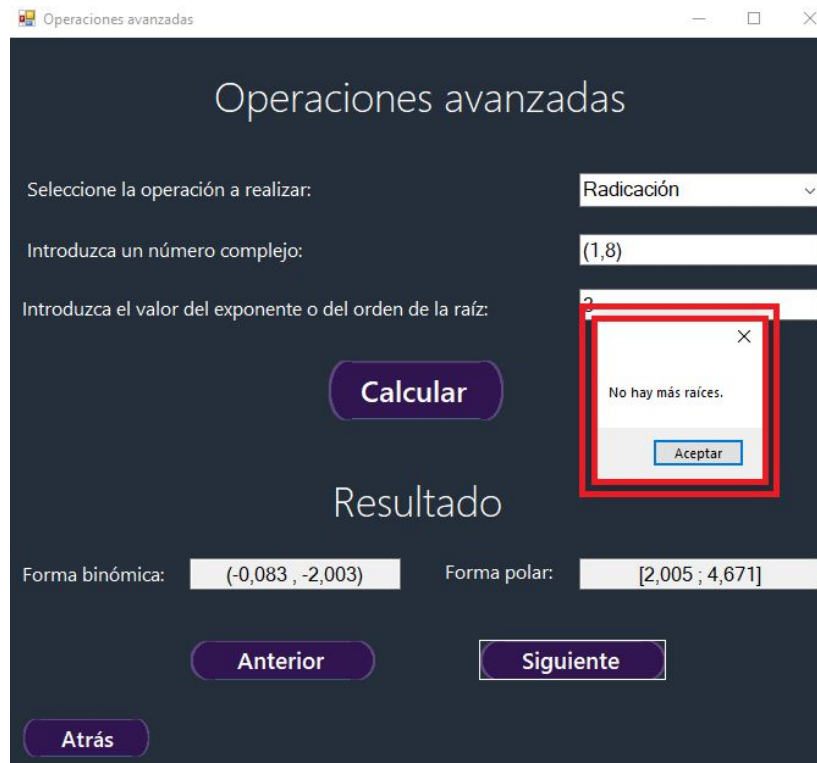
The interface has a dark blue background. At the top, the title 'Operaciones avanzadas' is centered in white. Below it, there are three input fields: 'Seleccione la operación a realizar:' with a dropdown menu showing 'Radicación', 'Introduzca un número complejo:' with the text '(1,8)', and 'Introduzca el valor del exponente o del orden de la raíz:' with the text '3'. A large purple button labeled 'Calcular' is centered below these inputs. Underneath the button, the word 'Resultado' is centered. Below 'Resultado', there are two text boxes: 'Forma binómica:' containing '(1,777 , 0,930)' and 'Forma polar:' containing '[2,005 ; 0,482]'. At the bottom, there are three purple buttons: 'Anterior', 'Siguiente', and 'Atrás'.

“Siguiente”:

This screenshot shows the 'Resultado' section of the interface. It features two text boxes: 'Forma binómica:' with the value '(-1,693 , 1,074)' and 'Forma polar:' with the value '[2,005 ; 2,577]'. Below these are two purple buttons: 'Anterior' and 'Siguiente'.

“Siguiente”:

This screenshot shows the 'Resultado' section of the interface. It features two text boxes: 'Forma binómica:' with the value '(-0,083 , -2,003)' and 'Forma polar:' with the value '[2,005 ; 4,671]'. Below these are two purple buttons: 'Anterior' and 'Siguiente'.



Si se cliquea “Siguiente” y no hay más raíces, aparece un cartel indicándolo.

Suma de Fasores

Para poder resolver este punto se utilizan las clases **Onda**, **TipoDeOnda** y **SumaDeFasores**.

La clase **‘Onda’** representa a las funciones trigonométricas de las cuales se obtendrán sus fasores . A su vez, esta clase tiene una variable **‘Tipo’** que sólo puede tener como valor **‘SENO’** o **‘COSENO’**, según el tipo de función que sea. Esto sirve para determinar si es necesario desplazar la fase inicial para ajustar las funciones que serán sumadas para que coincidan en su tipo.

Finalmente, la clase **‘SumaDeFasores’** guarda toda la lógica de la suma de los fasores asociados de ambas funciones. Para ello, se instancia esta clase con las ondas creadas a partir de los datos extraídos del formulario y se le pide realizar la suma de las mismas. Internamente, esta clase convierte las ondas en fasores binómicos y realiza la suma. Después los convierte después a su forma polar y retorna una instancia de onda llamada **‘ondaResultado’** con la amplitud y la fase inicial obtenidas de la forma polar, la frecuencia dada por el problema, y el tipo de onda que resulta del desplazamiento para hacerlas coincidir en la forma seno o coseno.

Suma de fasores

Suma de fasores

Primera onda

Amplitud: 5 Frecuencia: 2 Fase inicial: -1,047198 Tipo de onda: Coseno

Segunda onda

Amplitud: 8 Frecuencia: 2 Fase inicial: 0,523599 Tipo de onda: Coseno

Calcular

Resultados

Amplitud: 9,43397827767224 Frecuencia: 2 Fase inicial: 4,67738847593861 Tipo de onda: Seno

Amplitud: 9,43397827767224 Frecuencia: 2 Fase inicial: 6,24818480273351 Tipo de onda: Coseno

Atrás

Suma de fasores

Suma de fasores

Primera onda

Amplitud: 5 Frecuencia: 2 Fase inicial: -1,047198 Tipo de onda: Seno

Segunda onda

Amplitud: -8 Frecuencia: 2 Fase inicial: 0,523599 Tipo de onda: Coseno

Calcular

Resultados

Amplitud: 12,9999999999993 Frecuencia: 2 Fase inicial: 5,23598772145965 Tipo de onda: Seno

Amplitud: 12,9999999999993 Frecuencia: 2 Fase inicial: 6,80678404825455 Tipo de onda: Coseno

Atrás

Suma de fasores

Suma de fasores

Primera onda

Amplitud: 5 Frecuencia: 2 Fase inicial: -1,047198 Tipo de onda: Seno

Segunda onda

Amplitud: -8 Frecuencia: 3 Fase inicial: 0,523599 Tipo de onda: Coseno

Calcular

Amplitud: 12,999999999999999 Frecuencia: 3 Fase inicial: -1,047198 Tipo de onda: Seno

Amplitud: 12,999999999999999 Frecuencia: 3 Fase inicial: 0,523599 Tipo de onda: Coseno

Atrás

Error

Las funciones no pueden sumarse utilizando fasores dado que sus frecuencias son diferentes.

Aceptar

Recordamos que se debe utilizar una coma (",") como delimitador decimal, no un punto("."), como se aclaró en el Formato.

En el caso de que se introduzca un valor con "." como decimal, se interrumpe la ejecución del programa y avisa con éste cartel:

Error

El caracter que indica los decimales es ','
Por favor no usar '.'

Aceptar