

**UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN.**  
**FACULTAD DE CIENCIAS FÍSICO MATEMÁTICAS.**

**ANÁLISIS NUMÉRICO.**

**PROYECTO 2: MÉTODO BIRGE-VIETA.**

**PROFESORA: MARÍA DEL CARMEN MARTÍNEZ CEJUDO**

ALUMNO	MATRICULA
FRANCISCO JAVIER MARTÍNEZ PALOMAR.	1937837
KEVIN ALEXIS NÁJERA ÁBREGO.	1798194
ELÍ ISRAEL DELGADO ESCÁRCEGA.	1821213

**SAN NICOLÁS DE LOS GARZA, NUEVO LEÓN.**

# Índice

Índice.....	1
Introducción.....	2
Método escogido.....	3
¿Qué es el método Birge Vieta?.....	3
¿Por qué escogimos el método Birge Vieta? .....	3
Código. ....	4
Diagrama de flujo. ....	9
Manual de usuario y funcionamiento.....	11
Abrir la aplicación.....	11
Funcionamiento.....	12
Uso del programa y aplicaciones. ....	17
Conclusión.....	18
Referencias. ....	19

# Introducción.

El presente escrito es una descripción de un programa realizado para el curso de Análisis Numérico. Dicho programa utiliza el método Birge-Vieta para obtener la raíz o raíces de un polinomio dado.

Este método es sencillo de analizar, y utiliza iteraciones mientras no se cumpla una condición, por lo que considero especial para un lenguaje de programación. A continuación, se mostrarán más detalles sobre el programa, como implementamos este método y o como funciona tanto técnicamente como para un usuario. Además de algún ejemplo de corrida con un ejercicio dado.

# Método escogido.

## ¿Qué es el método Birge Vieta?

Dado un polinomio de la forma,

$$P_n(x) = x^n + a_1x^{n-1} + a_2x^{n-2} + \dots + a_{n-1}x + a_n = \sum_{i=1}^n a_i x^{n-i+1},$$

puede ser factorizado en la forma:

$$P(x) = (x - p_1)(x - p_2) \dots (x - p_n) = \prod_{i=1}^n (x - p_i),$$

donde  $p_i$  es una raíz del polinomio porque  $P(p_i) = 0$

El método Birge-Vieta aplica Newton-Raphson para encontrar una raíz del polinomio. Dado un punto  $x_k$ , evalúa  $P(x_k)$  y  $P'(x_k)$  mediante división sintética. Cuando encuentra una raíz  $p_i$ , elimina el factor  $(x - p_i)$  mediante división sintética y continúa trabajando sobre el polinomio resultante. El proceso se repite hasta encontrar todas las raíces del polinomio.

Es un método para la obtención de raíces de un polinomio, únicamente se aplica para la obtención de raíces reales del mismo.

## ¿Por qué escogimos el método Birge Vieta?

De todos los métodos que vimos hasta el momento nos pareció el más interesante debido a que se utiliza un método muy conocido por nosotros como lo es el método de división sintética, y pudimos notar el cambio en el polinomio inicial con cada iteración hasta encontrar una determinada raíz.

# Código.

Nuestro código fue desarrollado en lenguaje C y compilado como una aplicación ejecutable, es decir, con extensión .exe

A continuación, se muestra el código utilizado en la aplicación para resolver polinomios de grado n utilizando el método de Birge-Vieta, además de explicar en forma general su funcionamiento.

Int main es la función principal, desde esta se mandan llamar diversas funciones que realizan procedimientos en nuestro programa. Las dos principales son la de menú() y metodBirge().

```
int main() {
    system("color f0");
    int option, n, flag;
    welcome();
    float polinomio[MAX];
    do {
        option = menu();
        switch(option) {
            case 1:
                n = leerVector(polinomio);
                flag = metodoBirge(polinomio, n);
                if(flag==1) printf("\n\n Metodo Bige - Vieta aplicado
correctamente.\n\n");
                if(flag==0) printf("\n\n *No es posible continual con el
metodo Bige - Vieta, intente obtener raices con otro metodo.");
                if(flag== -1) printf("\n\n *No es posible aplicar el metodo
Bige - Vieta.\n\n");
                pause();
                break;
            case 2:
                break;
        }
    } while(option != 2);
    bye();
    return 0;
}
```

Int menu() es la función que se encarga de desplegar nuestro menú principal. Unicamente tiene dos opciones, la de ingresar un polinomio a resolver mediante Birge Vieta o la de salir del programa.

```
int menu () {
```

```

char str[MAX];
int option;

do{
    system("cls");
    printf("\n\n\t MENU: \n");
    printf("\n 1. Metodo Bige - Vieta.", 130);
    printf("\n 2. Salir.\n");
    do {
        printf("\n Seleccione una opcion: ", 162);
        in(str);
        option = atoi(str);
    }while(option != 1 && option != 2 || !isInteger(str));
    clear();
}while(option < 0);
return option;
}

```

Int metodoBirge es la función donde está la lógica principal del programa. Es donde se realizan todos los cálculos e impresiones importantes.

```

int metodoBirge(float polinomio[MAX],int n) {
    float x, polinomio1[n+1], mult1[n+1], polinomio2[n+1], mult2[n+1],
    polinomio3[n+1], solucion[n];
    int i, j, iteracion;
    int raicesRest;
    raicesRest = n;
    clear();
    //copiar polinomio
    for(i = 0 ; i < n + 1; i++){
        polinomio2[i] = polinomio[i];
        //printf("%f", polinomio2[i]);
    }
}

```

En esta sección nos encontramos con un for que se repite n veces, una vez por cada raíz buscada del polinomio.

```

    printf("\n Procedimiento de Bige - Vieta:\n\n");
    for(i = 0 ; i < n; i++){
        printf("***** RAIZ   %d\n", i+1);
        for(j = 0 ; j < raicesRest + 1; j++){
            polinomio1[j] = polinomio2[j];

```

```

        //printf("%f", polinomio1[j]);
    }
    //Valor de x
    if(polinomio1[raicesRest - 1] != 0){
        x = (-1.0)*polinomio1[raicesRest]/polinomio1[raicesRest - 1];
    } else return -1;
    //Inicializar variables
    for(j = 0; j < raicesRest + 1; j++) {
        mult1[j] = 0;
        polinomio2[j] = 0;
        mult2[j] = 0;
        polinomio3[j] = 0;
    }

```

Esta sección contiene un bloque do-while que se repite mientras la división sintética sea diferente de cero, es decir, mientras esta condición se cumpla, el programa seguirá obteniendo nuevos valores de x, hasta aproximarse a la raíz exacta.

```

iteracion = 0;
do {
    printf(" X%d = %.4f\t", iteracion, x);
    for(j = 0; j < raicesRest + 1; j++) {
        if(j == 0) {
            polinomio2[j] = polinomio1[j];
        } else {
            mult1[j] = polinomio2[j-1] * x;
            polinomio2[j] = polinomio1[j] + mult1[j];
        }
    }
    for(j = 0; j < raicesRest; j++) {
        if(j == 0 && j < raicesRest - 1) {
            polinomio3[j] = polinomio2[j];
        } else {
            mult2[j] = polinomio3[j - 1] * x;
            polinomio3[j] = polinomio2[j] + mult2[j];
        }
    }
    for(j = 0; j < raicesRest + 1; j++) {
        printf(" %.2f", polinomio1[j]);
    }
    printf("\n\t\t");
    for(j = 0; j < raicesRest + 1; j++) {
        printf(" %.2f", mult1[j]);
    }
    printf("\n\t\t");

```

```

        for(j = 0; j < raicesRest + 1; j++) {
            printf("_____");
        }
        printf("\n\t\t");
        for(j = 0; j < raicesRest + 1; j++) {
            printf(" %.2f", polinomio2[j]);
        }
        printf("\n\t\t");
        for(j = 0; j < raicesRest; j++) {
            printf(" %.2f", mult2[j]);
        }
        printf("\n\t\t");
        for(j = 0; j < raicesRest + 1; j++) {
            printf("_____");
        }
        printf("\n\t\t");
        for(j = 0; j < raicesRest; j++) {
            printf(" %.2f", polinomio3[j]);
        }
        printf("\n\n");
        x = x - polinomio2[raicesRest]/polinomio3[raicesRest - 1];
        iteracion++;
        if(iteracion == 100) return 0;
    }while(polinomio2[raicesRest] != 0 || polinomio2[raicesRest] != -0);
    solucion[i] = x;
    raicesRest--;
}
printf("*****\n\n");

```

Una vez obtenidas las soluciones, estas se deben imprimir para poderlas mostrar a usuario final. Para eso son los siguiente printf, que muestran las soluciones.

```

//solucion[n - 1] = ((-1) * polinomio2[1]) / polinomio[0];
printf("\n\n      Ecuacion: \n\n");
imprimirVector(polinomio, n);
printf("\n\n      SOLUCIONES: \n\n");
for(i = 0; i < n; i++) {
    printf("\n X%d = %.6f", i+1, solucion[i]);
}
}

```

Int leerVector es una función que se encarga de leer un vector donde se guardarán los coeficientes del polinomio ingresado.

```

int leerVector(float vector[MAX]) {

```



```

int n, option;
char str[MAX];

do {
    clear();
    printf("\n\n\tMETODO BIGE - VIETA\n");
    do {
        printf(" \n Ingrese el grado de su ecuacion: ");
        in(str);
        n = atoi(str);
    } while(n < 2 || !isNumber(str));

    clear();
    int i;

    printf("\n\n\t Ingrese los coeficientes de cada termino");

    for(i = 0 ; i <= n ; i++){
        do{
            printf("\n x^%i: ",n-i);
            in(str);
            vector[i] = atof(str);
        }while(!isFloat(str));
    }

    clear();

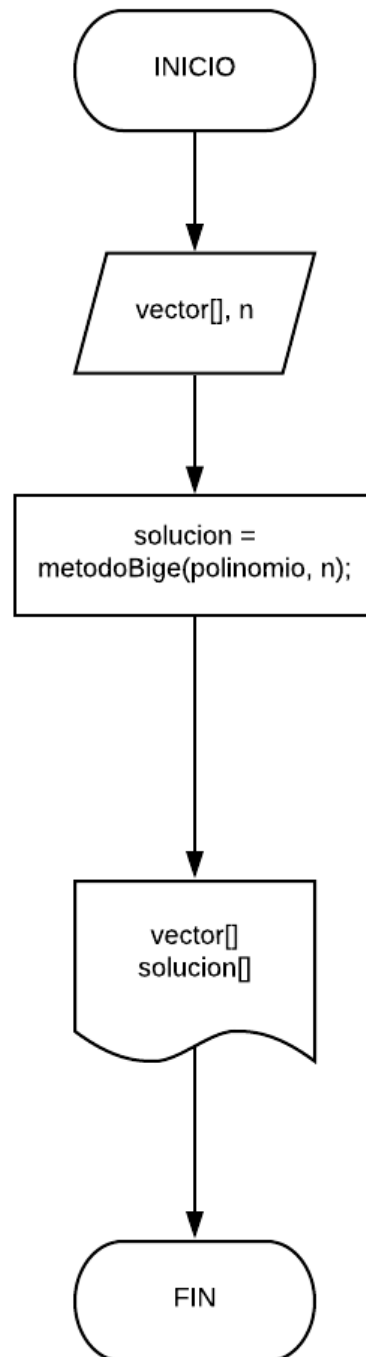
    printf("\n\n %cSu ecuacion es la siguiente?\n\n",168);
    /*Imprimo la matriz que forman los coeficientes de la ecuacion*/
    imprimirVector(vector, n);
    do{
        printf("\n [1.Si 2.No]: ");
        in(str);
        option = atoi(str);
    }while(option!=1 && option!=2 || !isInteger(str));

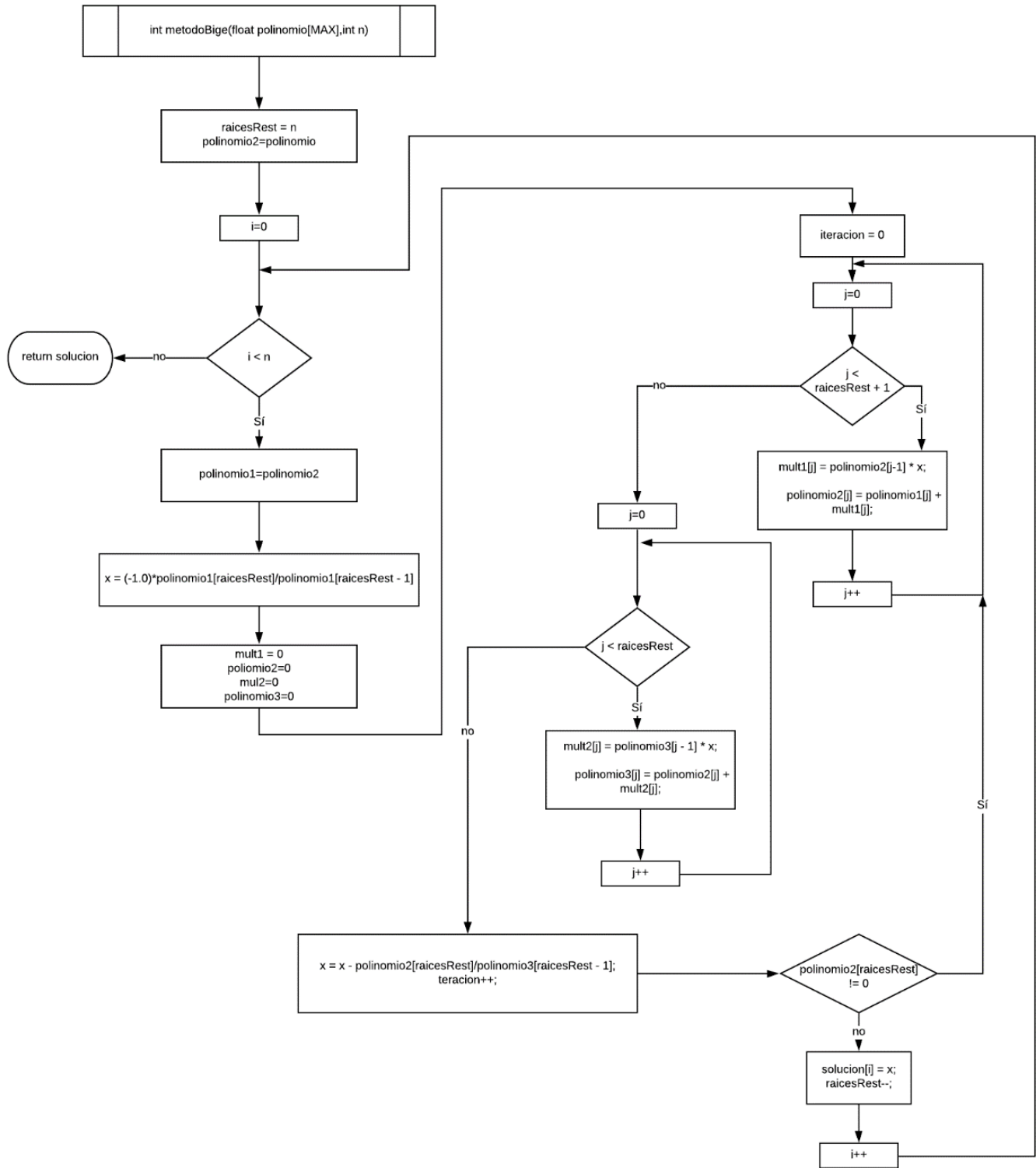
    }while(option != 1);
return n;
}

```

# Diagrama de flujo.

A continuación, se presenta el diagrama de flujo para el Birge Vieta.





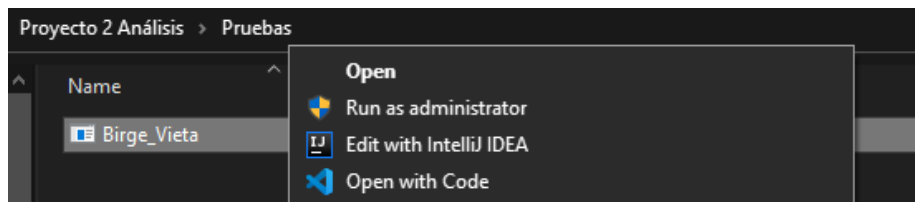
# Manual de usuario y funcionamiento.

El programa Birge\_Vieta.exe tiene como finalidad obtener la raíz o raíces de un polinomio de grado  $n$ , a través del uso del Método de Birge-Vieta. Además de mostrar la solución, el programa es capaz de ir desplegando el procedimiento, e indicar si el polinomio puede ser resuelto o no por nuestro método.

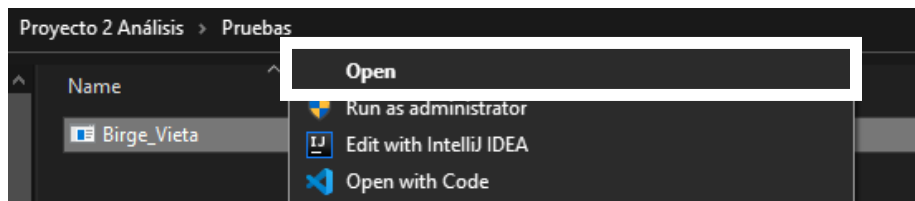
## Abrir la aplicación.

Para poder abrir la aplicación se realizan los siguientes pasos.

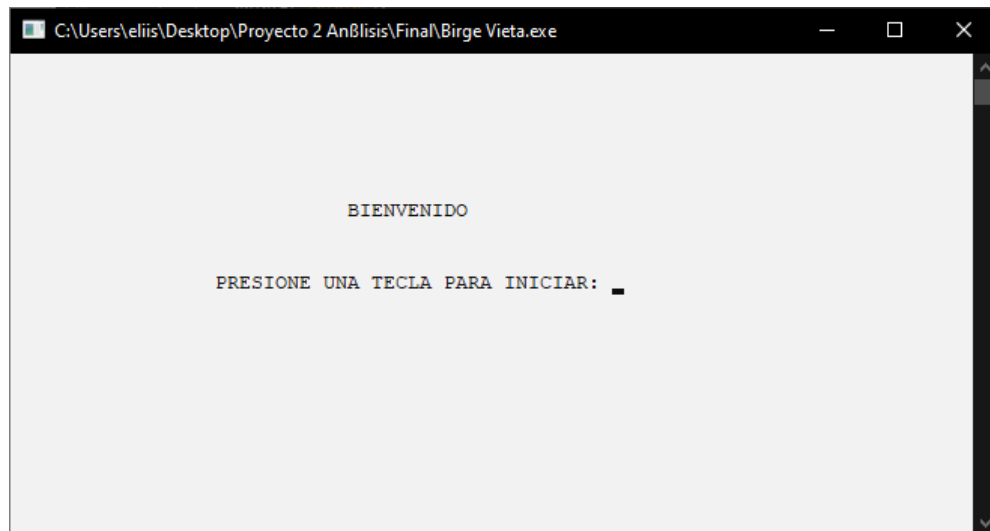
1. Clic derecho para seleccionar y clic izquierdo para desplegar menú en nuestra aplicación Birge\_Vieta.exe:



2. Clic derecho en abrir u open (dependiendo del idioma del equipo):

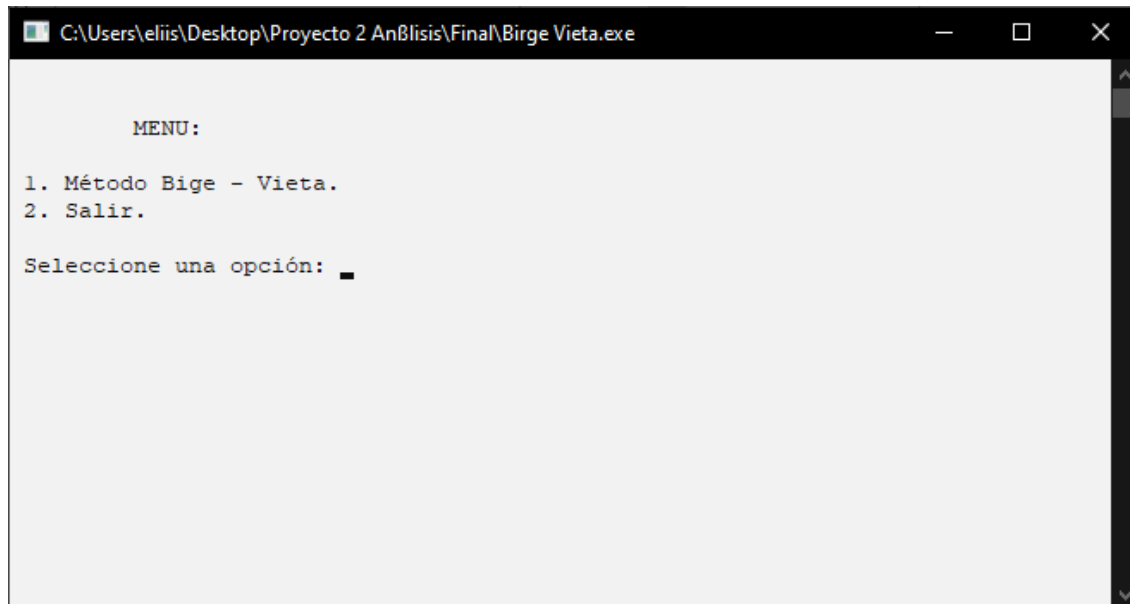


3. La aplicación se desplegará como en la siguiente ventana:

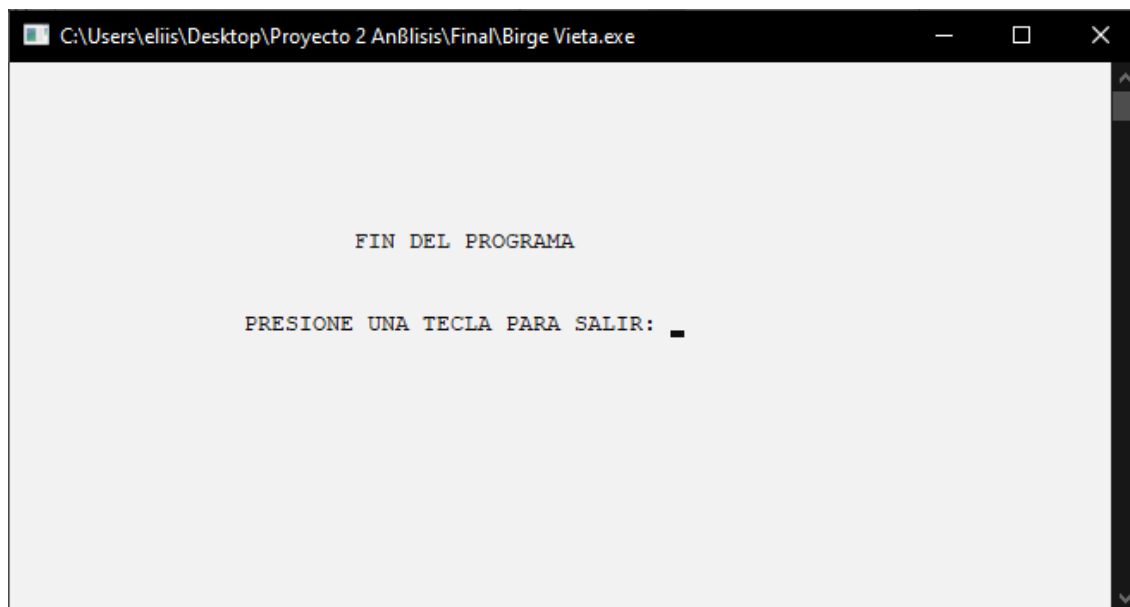


## Funcionamiento.

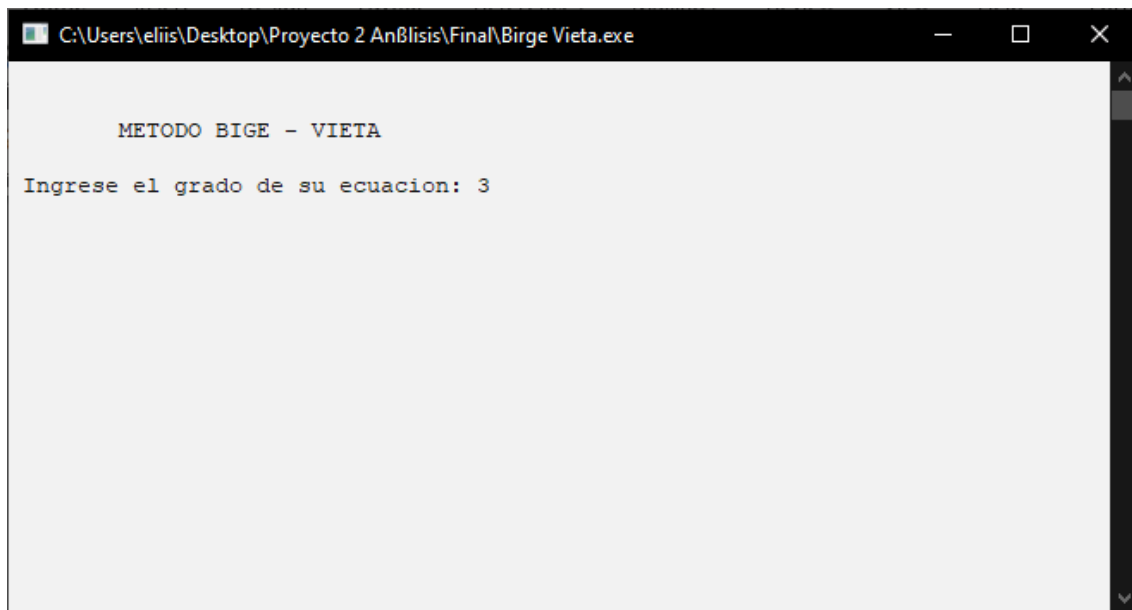
Después de abrir la aplicación, se desplegará una ventana de bienvenida. Para iniciarla debemos presionar cualquier tecla. Una vez hecho esto, nos mostrará un menú de inicio:



Debemos escoger una opción ingresando un número (1 o 2), y dar enter: Si escogemos la opción 2, nos desplegará la ventana de Fin del programa:



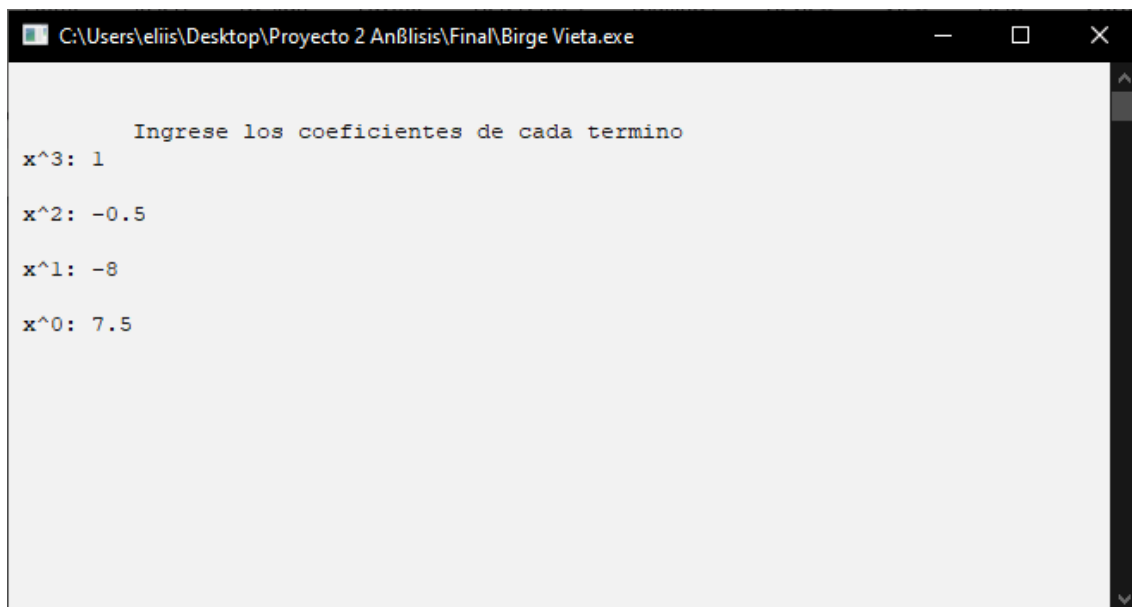
Si escogemos la opción 1, podremos ingresar el polinomio a resolver. Nos preguntará el grado del polinomio que queremos ingresar, este número debe ser mayor o igual a 2:



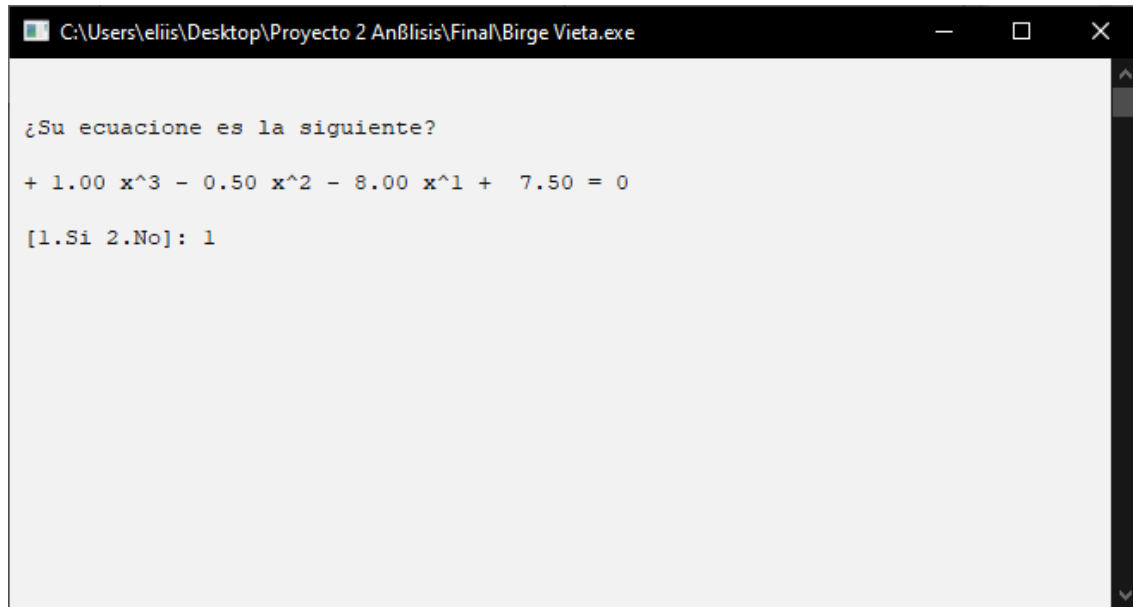
Una vez escogido el grado, ingresaremos el polinomio coeficiente a coeficiente. Debemos ingresar el coeficiente y dar enter uno a uno. Para el siguiente polinomio

$$x^3 - 0.5x^2 - 8x + 7.5 = 0$$

esta sería la entrada correspondiente:



Nos desplegará una venta de confirmación donde deberemos seleccionar una opción (1 o 2). En caso de seleccionar no, nos volverá a pedir el grado del polinomio, y el polinomio.



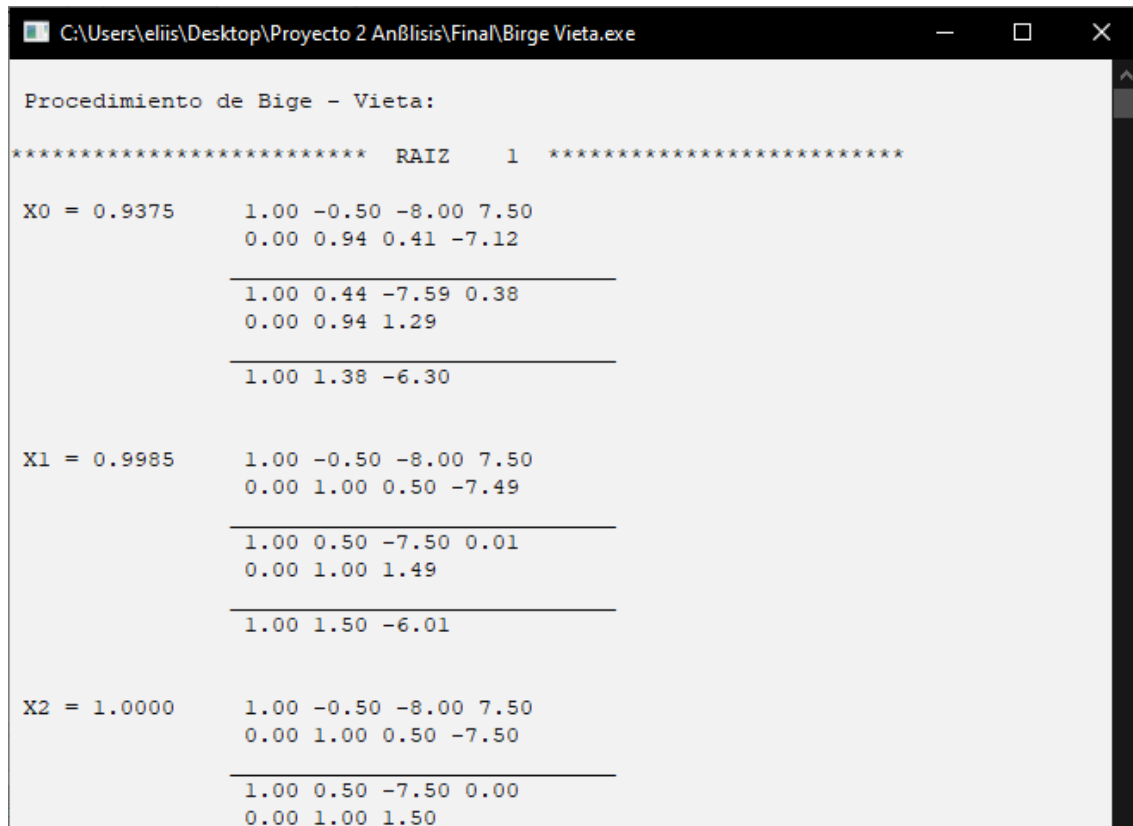
```
C:\Users\eliis\Desktop\Proyecto 2 Análisis\Final\Birge Vieta.exe

¿Su ecuacione es la siguiente?

+ 1.00 x^3 - 0.50 x^2 - 8.00 x^1 + 7.50 = 0

[1.Si 2.No]: 1
```

Si seleccionamos que sí, nos mostrará el resultado y procedimiento para la resolución:



```
C:\Users\eliis\Desktop\Proyecto 2 Análisis\Final\Birge Vieta.exe

Procedimiento de Bige - Vieta:

***** RAIZ      1 *****

X0 = 0.9375      1.00 -0.50 -8.00 7.50
                  0.00 0.94 0.41 -7.12
                  -----
                  1.00 0.44 -7.59 0.38
                  0.00 0.94 1.29
                  -----
                  1.00 1.38 -6.30

X1 = 0.9985      1.00 -0.50 -8.00 7.50
                  0.00 1.00 0.50 -7.49
                  -----
                  1.00 0.50 -7.50 0.01
                  0.00 1.00 1.49
                  -----
                  1.00 1.50 -6.01

X2 = 1.0000      1.00 -0.50 -8.00 7.50
                  0.00 1.00 0.50 -7.50
                  -----
                  1.00 0.50 -7.50 0.00
                  0.00 1.00 1.50
```

```

C:\Users\eliis\Desktop\Proyecto 2 Analisis\Final\Birge Vieta.exe
***** RAIZ 2 *****

X0 = 15.0000    1.00 0.50 -7.50
                0.00 15.00 232.50
                -----
                1.00 15.50 225.00
                0.00 15.00
                -----
                1.00 30.50

X1 = 7.6230     1.00 0.50 -7.50
                0.00 7.62 61.92
                -----
                1.00 8.12 54.42
                0.00 7.62
                -----
                1.00 15.75

X2 = 4.1668     1.00 0.50 -7.50
                0.00 4.17 19.45
                -----
                1.00 4.67 11.95
                0.00 4.17
                -----
                1.00 8.83

C:\Users\eliis\Desktop\Proyecto 2 Analisis\Final\Birge Vieta.exe
***** RAIZ 3 *****

X0 = -3.0000    1.00 3.00
                0.00 -3.00
                -----
                1.00 0.00
                -0.00
                -----
                1.00

*****

Ecuacion:

+ 1.00 x^3 - 0.50 x^2 - 8.00 x^1 + 7.50 = 0

SOLUCIONES:

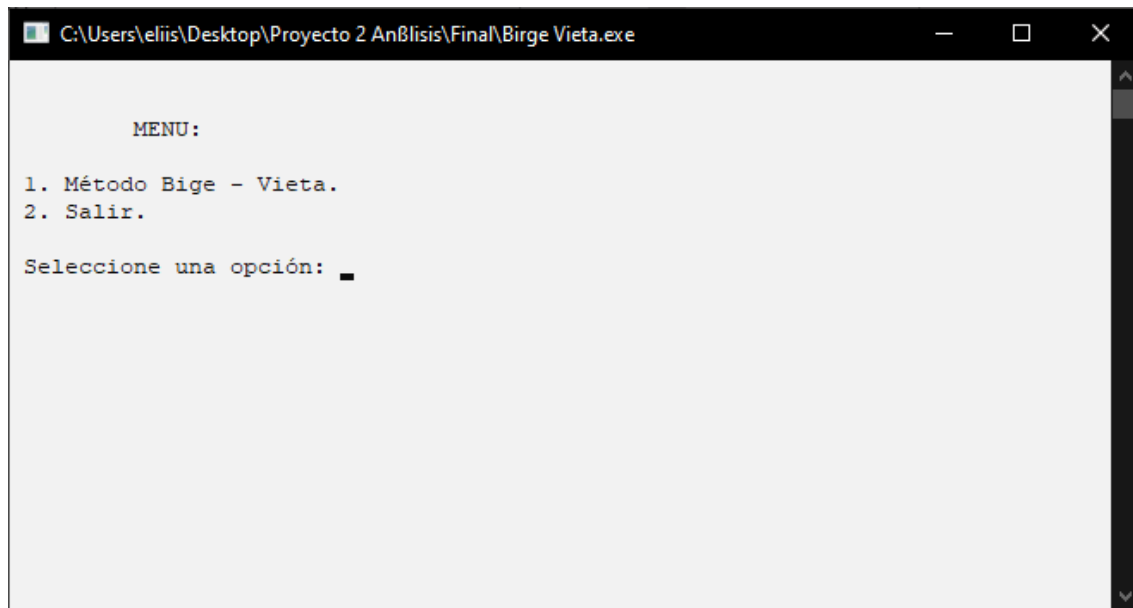
X1 = 1.000000
X2 = 2.500000
X3 = -3.000000

Metodo Birge - Vieta aplicado correctamente.

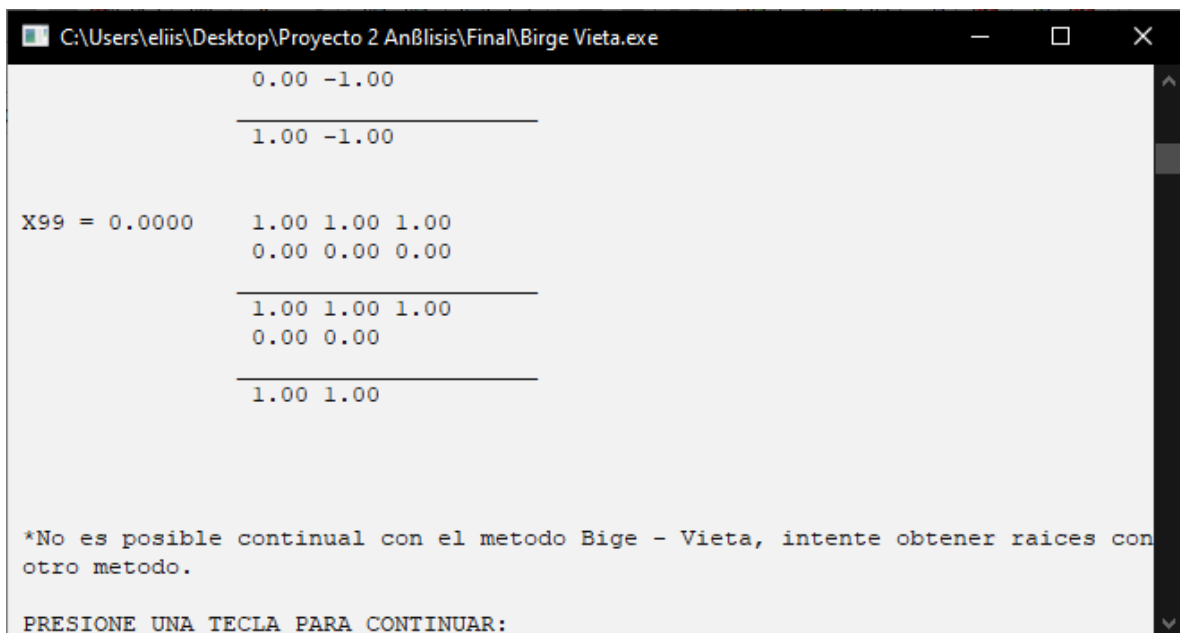
```



Una vez vista la respuesta, si damos clic a cualquier tecla, nos desplegará el menú de inicio nuevamente:



En caso de haber ingresado un polinomio con solución imaginaria, o coeficiente en x igual a 0, el programa nos indicará que no es posible dar solución. Por ejemplo, para el polinomio  $x^2 + x + 1 = 0$  obtenemos:



# Uso del programa y aplicaciones.

Los polinomios pueden ser utilizados en la planificación financiera. Por ejemplo, una ecuación polinómica se puede utilizar para calcular la cantidad de interés que se devengará de una cantidad de depósito inicial en una inversión o cuenta de ahorros a una tasa de interés dada.

Los polinomios son objetos muy utilizados en matemáticas y en ciencia. En la práctica, son utilizados en cálculo y análisis matemático para aproximar cualquier función derivable.

Los polinomios generalmente se aplican a la hora de calcular áreas y volúmenes, lo que muchas personas desconocen es que la aplicación de los polinomios va más allá de lo que pensamos, ya que se usan mucho en la medicina, la ciencia, la química, entre otros.

Ejemplo:

En un rectángulo el largo mide  $(x + 7)$  y el ancho  $(x + 2)$ . Si el área del rectángulo es 36, halla el valor de  $x$ .

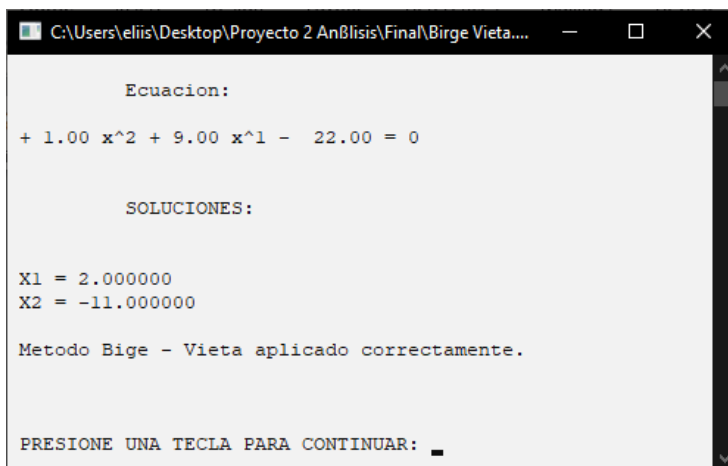
Planteamiento:

$$(x + 7)(x + 2) = 36$$

Desarrollando:

$$x^2 + 9x - 22 = 0$$

Ingresándolo a la aplicación:



Obtenemos las raíces:

$$x_1 = 2$$

$$x_2 = -11$$

Al ser un área, sus dimensiones deben ser positivas. Por lo tanto, el resultado es  $x = 2$ .

# Conclusión

La elaboración de este proyecto nos ayudó a reforzar el manejo del método, debido a que lo analizamos paso a paso para poder desarrollar el código de este. Desafortunadamente durante el proceso y las pruebas que realizábamos nos topamos con el inconveniente de que este no sirve para la obtención de raíces imaginarias, generando un error de tipo cíclico durante la resolución de un problema determinado. Como los cálculos por computadora se realizan mucho más rápido no demora mucho tiempo detectar este tipo de situaciones, por lo que la solución no fue muy complicada, otro caso hubiera sido si hiciéramos un problema de este tipo a mano.

En general, programar este tipo de métodos ayuda a ahorrar tiempo a la hora de resolver polinomios de grado  $n$ , dado que evita la necesidad de realizar iteraciones con lápiz y papel hasta encontrar la raíz o raíces.

# Referencias.

Prezi. ¿Dónde se aplican los polinomios? Recuperado de:  
<https://prezi.com/bw6tac1othdz/donde-se-aplican-los-polinomios/>

Itesm. Recuperado de: <https://www.mty.itesm.mx/dtie/deptos/cb/cb00854-1/Apuntes/nolineales-foils.pdf>