Método de la ingeniería

1. **Identificación del problema:**

* La empresa Oracle requiere el desarrollo de un software que encuentre la raíz o raíces de un polinomio de máximo grado 10 con dos algoritmos diferentes.
* No hay información de desarrollo previo ni soluciones planteadas con antelación.
* La solución del problema debe garantizar las raíces reales e imaginarias, si existen, con el uso de cualquiera de los dos algoritmos.
* La solución del problema deben ser los dos algoritmos más eficientes encontrados de los encontrados
* La solución debe ser capaz de generar un polinomio aleatorio de máximo de grado 10 y coeficientes de máximo valor permitido.

1. **Recopilación de información:**

Polinomio

Un polinomio es una expresión matemática constituida por un conjunto finito de variables (no determinadas o desconocidas) y constantes (números fijos llamados coeficientes), utilizando únicamente las operaciones aritméticas de suma, resta y multiplicación, así como también exponentes enteros positivos.

Raíces de un polinomio

Las raíces de un **polinomio**son números tales que hacen que un polinomio valga cero.

La cantidad de raíces que tiene un polinomio lo determinamos por el teorema fundamental del algebra, el cual dice, un polinomio en una variable de coeficientes complejos y no constante tiene tantas raíces como su grado. Entonces para un polinomio de grado n posee puntualmente n soluciones complejas.

Grado de un polinomio

Es el grado máximo de los exponentes de las variables de los monomios que lo componen,

Mejor dicho, la variable constituida de un literal y su exponente de mayor exponente entre

el polinomio.

Raíz real

Son las intersecciones con el eje x del plano donde en aquellos valores en la función vale cero

Raíz compleja

Son aquellas raíces donde la solución no se encuentra dentro del limite de los reales, por ejemplo,

un polinomio x^2+1 que tiene raíz compleja.

1. **Búsqueda de soluciones**

Alternativa 1. Método de Newton Raphson

* Requiere de un punto para empezar a aproximar, entre mas cerca este a la raíz el

punto escogido, en menos pasos puede encontrar una aproximación mas precisa a la raíz del polinomio, de ahí se sacan rectas tangentes para hallar el punto de corte de cada una, entre mas se repita este proceso se acercará cada vez más a la raíz del polinomio que estamos buscando

Alternativa 2. Método de división sintética

* Método en el que se usa la conocía división sintética, donde se sacan los coeficientes de el monomio con mayor grado (p)y la constante c(q) y se les sacan sus correspondientes divisores p y q, luego se sacan combinaciones de la forma p/q y de ahí se procede a probar cada una de estas combinaciones con los coeficientes del polinomio.

Alternativa 3. Método de Müller

* Toma tres puntos, dos de entrada y uno que se puede calcular por bisección o regula-falsi, y aproxima una función a la que le estamos buscando las raíces para así aproximar su solución f(x)=0

1. Transición de las Ideas a los Diseños Preliminares

Alternativa 1. Método de Newton Raphson

* El método implica iterar desde un primer valor aproximado hasta llegar a un valor tan cercano como un valor de error mínimo indicado sea alcanzado.
* Es un método abierto en el que no está garantizada su convergencia global.

Alternativa 2. Método de división sintética

* Requiere probar las combinaciones entre el coeficiente del monomio de mayor grado y la constante del polinomio a resolver para hallar raíces reales.
* Para hallar raíces completas requiere el redimensionamiento de los coeficientes hasta llegar a un polinomio más simple.
* Requiere sacar los coeficientes del polinomio y aplicar el proceso completo de división sintética

Alternativa 3. Método de Müller

* El método tiene un error que viene determinado por |Xn – Xn-1/Xn|
* El método permite hallar raíces complejas
* Al calcular las aproximaciones, hay que elegir el signo del denominador que coincida con el valor del coeficiente que hemos hallado.

1. Evaluación y Selección de la Mejor Solución

Criterios:

* Criterio A. Precisión de la solución. Entrega una solución:
  + [2] exacta
  + [1] aproximada
* Criterio B. Dificultad de implementación. Que tan difícil es implementarlo en java:
  + [2] difícil
  + [1] fácil
* Criterio C. completitud. Se prefiere una solución que encuentre todas las soluciones. Cuantas soluciones entrega:
  + [3] Todas
  + [2] mas de una si las hay, pero no todas
  + [1] solo pocos casos una o ninguna
* Criterio D. Eficiencia. Se prefiere a un algoritmo eficiente en base a su complejidad temporal, tipo de complejidad:
  + [4] Constante
  + [3] Mayor a constante
  + [2] Logarítmica
  + [1] Lineal

**Evaluación**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Alternativas | Criterio A. | Criterio B. | Criterio C. | Criterio D. |
| Alternativa 1. Método de Newton Raphson | 1. aproximada |  | 3. todas |  |
| Alternativa 2. Método de división sintética |  |  | 3. todas |  |
| Alternativa 3. Método de Müller | 1. aproximada |  | 3. todas |  |

Selección

De acuerdo con la evaluación anterior se decide usar las alternativas 1 y 2 por ……………..(Cosas que se infieren de la tabla cuando este completa)

**6. Preparación de informes y especificaciones**

**7. Implementación del diseño**