

Development of Calculator for Finding Complex Roots of n^{th} Degree Polynomials

Yaniar Rahmah

Informatika, Fakultas MIPA

Universitas Sebelas Maret

Jl. Ir. Sutami No. 36A Surakarta

yaniarrahmah@student.uns.ac.id

Sarngadi Palgunadi

Informatika, Fakultas MIPA

Universitas Sebelas Maret

Jl. Ir. Sutami No. 36A Surakarta

palgunadi@staff.uns.ac.id

Esti Suryani

Informatika, Fakultas MIPA

Universitas Sebelas Maret

Jl. Ir. Sutami No. 36A Surakarta

suryapalapa@yahoo.com

ABSTRACT

Software as a learning medium has become a trend in society. The development of science and technology encourage modernization in every field of education, including mathematics. One of the mathematical models often used is polynomial equation. Many mathematicians have done research on solving polynomials by using numerical algorithms, but only focus on mathematical analysis. In this research, design and construction of a calculator as a medium of learning has been done. The calculator is completed by many algorithm such as the quadratic formula, Cardano, Viète's, Bairstow, revise Bairstow, Muller, and combination algorithms. The calculator was designed by UML approach and implemented on Java Swing GUI. The test results using black box method showed that 100% of functional calculator can work well. Based on the average calculation error, cubic polynomial is solved properly by using Cardano algorithm which the average error is 1.43568059121049E-13%. Revise Bairstow algorithm showed good performance to find the complex roots of 4th degree polynomial with an average error is 1.34421873307271E-09% and average of iteration is 4. The best combination algorithm on solving n^{th} degree polynomials is Muller-Cardano which has an average error 4.27781615793958E-13% and Revise Bairstow-Cardano with average number of iteration is 7 iterations.

Keywords

Finding roots polynomial, calculator, quadratic formula, Cardano, Viète's, Bairstow, Revised Bairstow, Muller

1. PENDAHULUAN

Perangkat lunak sebagai media pembelajaran telah menjadi sebuah tren di kalangan masyarakat. Kata “media” berasal dari bahasa Latin dan merupakan bentuk jamak dari kata “medium” yang secara harfiah berarti “perantara” yaitu perantara sumber pesan dengan penerima pesan [1]. Media pembelajaran merupakan teknologi pembawa pesan yang dapat dimanfaatkan untuk keperluan pembelajaran. Media pembelajaran dapat berupa sarana komunikasi seperti buku, film, video, slide, maupun perangkat lunak [2]. Perangkat lunak (*software*) merupakan istilah yang digunakan untuk menggambarkan instruksi-instruksi yang memberitahu perangkat keras untuk melakukan suatu tugas sesuai dengan perintah [3]. Berdasarkan pengertian media dan perangkat lunak maka dapat diketahui perangkat lunak sebagai media pembelajaran merupakan penerapan *software* pada perangkat keras komputer yang bertujuan menyampaikan pesan pembelajaran. Perkembangan ilmu pengetahuan dan teknologi mendorong modernisasi di setiap bidang pendidikan, termasuk bidang matematika. Model matematika kerap kali digunakan dalam berbagai bidang seperti bidang fisika, kimia, ekonomi, teknik sipil,

teknik mesin, elektro, dan lainnya [4]. Salah satu model matematika yang sering digunakan adalah polinomial. Persamaan polinomial memiliki bentuk $x^n + a_1x^{n-1} + a_2x^{n-2} + a_3x^{n-3} + \dots + a_n = 0$ dimana a_1, a_2, \dots, a_n merupakan koefisien real [5]. Sebuah polinomial dengan derajat n akan memiliki akar sebanyak n . Akar polinomial dapat berupa bilangan real atau kompleks [6].

Penelitian yang dilakukan Iges Windra menyatakan bahwa metode Bairstow memiliki laju konvergensi yang cukup tinggi dan menghasilkan galat yang rendah dalam mencari akar polinomial [7]. Perhitungan metode Bairstow tersebut dikembangkan dan dikenal sebagai revisi algoritma Bairstow [5]. Metode Muller sukses diaplikasikan pada komputer dengan menawarkan banyak keunggulan [8]. Namun, penelitian lain menyebutkan berdasarkan jumlah langkah kerja, Bairstow lebih baik kinerjanya dibandingkan metode Muller [9]. Dalam penelitian lainnya, penyelesaian persamaan polinomial berderajat tiga dapat dilakukan dengan menggunakan solusi Cardan [10] dan substitusi Viète's [11].

Berdasarkan penelitian-penelitian di atas, diketahui bahwa keempat penelitian mengenai pencarian akar kompleks polinomial dilakukan oleh matematikawan-matematikawan sehingga penelitian hanya fokus pada penyelesaian polinomial secara matematika saja, sedangkan media pembelajaran mengenai polinomial masih sangat minim. Oleh karena itu, peneliti tertarik untuk membuat sebuah perangkat lunak sebagai media pembelajaran polinomial. Perangkat lunak berupa sebuah kalkulator yang didalamnya telah diimplementasikan metode-metode numerik khusus pencarian akar kompleks polinomial seperti rumus kuadrat, algoritma Cardano, Viète's, Bairstow, revisi Bairstow, Muller, serta kombinasi algoritma. Sampai saat ini belum tersedia media pembelajaran kalkulator pencarian akar kompleks polinomial yang dilengkapi dengan beberapa pilihan algoritma.

Analisa dan perancangan program kalkulator dilakukan dengan menggunakan pendekatan UML dan diimplementasikan dengan menggunakan bahasa Java Swing GUI. Kalkulator yang dibuat dapat menghitung akar kompleks pada polinomial derajat dua, derajat tiga, derajat empat, dan derajat n . Galat (*error*) yang dihasilkan dari perhitungan pada masing-masing algoritma dalam pencarian akar kompleks polinomial akan dibandingkan untuk mengetahui algoritma terbaik. Kriteria algoritma terbaik adalah algoritma yang menghasilkan galat terendah dengan jumlah langkah kerja (iterasi) terkecil.

2. TINJAUAN PUSTAKA

2.1 Polinomial

Persamaan polinomial memiliki bentuk standar seperti berikut :

$$a_0x^n + a_1x^{n-1} + a_2x^{n-2} + \dots + a_{n-1}x + a_n = 0 \quad (2.1)$$

commit to user

dimana a disebut sebagai koefisien polinomial yang merupakan bilangan riil dan n merupakan derajat polinomial yang berupa bilangan bulat positif. Suku-suku pangkat dari x disusun dalam derajat menurun [12]. Polinomial disebut sebagai polinomial *monoid*. Polinomial monoid merupakan polinomial yang koefisien variabel x berderajat tertinggi sama dengan 1. Bentuk polinomial monoid adalah sebagai berikut:

$$x^n + a_1x^{n-1} + a_2x^{n-2} + \dots + a_n = 0 \quad (2.2)$$

Contoh-contoh polinomial berdasarkan derajat polinomial adalah sebagai berikut :

- a. Polinomial kuadrat (*quadratic polynomial*) yaitu polinomial dengan nilai pangkat tertinggi dari suku x adalah 2 [13].

$$f(x) = ax^2 + bx + c = 0 \quad (2.3)$$

- b. Polinomial kubik (*cubic polynomial*) merupakan polinomial yang nilai pangkat tertinggi dari suku x adalah tiga ($n=3$).

$$f(x) = ax^3 + bx^2 + cx + d = 0 \quad (2.4)$$

dengan a, b, c, d merupakan bilangan real dan $a \neq 0$ [15].

- c. Polinomial kuartik (*quartic polynomial*) adalah polinomial berderajat empat dimana nilai pangkat tertinggi x adalah empat ($n=4$).

$$f(x) = ax^4 + bx^3 + cx^2 + dx + e = 0 \quad (2.5)$$

- d. Polinomial berderajat n merupakan polinomial dengan nilai pangkat tertinggi dari variabel x adalah n .

$$f(x) = x^n + a_1x^{n-1} + a_2x^{n-2} + \dots + a_n = 0 \quad (2.6)$$

dimana a_1, a_2, \dots, a_n merupakan bilangan real [5].

Persamaan polinomial $f(x) = 0$ memiliki akar polinomial r jika dan hanya jika $f(r) = 0$. Akar-akar sebuah persamaan polinomial dapat berupa akar-akar kompleks, akar-akar irasional dan rasional. Apabila persamaan polinomial $f(x) = 0$ memiliki koefisien real dan jika bilangan kompleks $a + bi$ adalah akar dari $f(x) = 0$, maka konjugasi (sekawan) kompleks $a - bi$ juga merupakan akar persamaan polinomial tersebut [12].

2.1.1 Rumus Kuadrat

Polinomial kuadrat (Persamaan 2.3) dan dapat dituliskan dalam bentuk lain seperti Persamaan 2.6.

$$x^2 + px + q = 0 \quad (2.6)$$

dimana nilai $p = \frac{b}{a}$ dan nilai $q = \frac{c}{a}$ [13]. Maka diskriminan :

$$D = \left(\frac{p}{2}\right)^2 - q \quad (2.7)$$

Penyelesaian polinomial kuadrat Persamaan 2.3 dapat dilakukan dengan rumus kuadrat berikut :

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad (2.8)$$

Penyelesaian bentuk standar polinomial kuadrat Persamaan 2.6 :

$$x_{1,2} = -\frac{p}{2} \pm \sqrt{\left(\frac{p}{2}\right)^2 - q} = -\frac{p}{2} \pm \sqrt{D} \quad (2.9)$$

2.1.2 Algoritma Cardano

Apabila polinomial kubik (Persamaan 2.4) dibagi dengan a akan menghasilkan bentuk polinomial monoid seperti berikut:

$$x^3 + rx^2 + sx + t = 0 \quad (2.10)$$

dimana

$$r = \frac{b}{a}; s = \frac{c}{a}; t = \frac{d}{a}$$

Polinomial kubik dapat diselesaikan dengan menggunakan algoritma Cardano yang diterbitkan oleh Girolamo Cardano (1501-1576) dalam tulisannya *Ars Magna* [15]. Berikut merupakan langkah-langkah algoritma Cardano [13] :

1. Menginputkan polinomial dengan bentuk Persamaan 2.10 dan menentukan nilai variabel r, s , dan t .
2. Mensubstitusikan nilai y pada Persamaan 2.10 :

$$y = x + \frac{r}{3} \quad (2.11)$$

Substitusi nilai y akan menghasilkan persamaan tereduksi :

$$y^3 + py + q = 0 \quad (2.12)$$

$$q = \frac{2r^3}{27} - \frac{rs}{3} + t \text{ dan } p = \frac{3s-r^2}{3}$$

3. Menghitung nilai diskriminan persamaan tereduksi.

$$D = \left(\frac{p}{3}\right)^3 + \left(\frac{q}{2}\right)^2 \quad (2.13)$$

4. Menentukan penyelesaian polinomial kubik dan menghitung akar-akar polinomial (x_1, x_2 dan x_3). Penyelesaian ditentukan berdasarkan nilai diskriminan [10] :

- a. Penyelesaian polinomial kubik dengan $D \geq 0$ dilakukan dengan menghitung nilai u dan v :

$$u = \sqrt[3]{-\frac{q}{2} + \sqrt{D}} \quad (2.14)$$

$$v = \sqrt[3]{-\frac{q}{2} - \sqrt{D}} \quad (2.15)$$

Akar polinomial kubik didapat dari rumus berikut:

$$x_1 = -\frac{r}{3} + u + v \quad (2.16)$$

$$x_{2,3} = -\frac{r}{3} - \frac{u+v}{2} \pm \sqrt{3} \frac{u-v}{2} i \quad (2.17)$$

dimana nilai $i = \sqrt{-1}$.

- b. Apabila $D < 0$ akan menghasilkan tiga akar riil yang berbeda (*irreducible case*). Penyelesaian dilakukan dengan menggunakan trigonometri :

$$\cos \varphi = -\frac{q}{2\sqrt{(p/3)^3}} \quad (2.18)$$

sehingga solusi dari persamaan tereduksi Persamaan 2.12 adalah sebagai berikut :

$$x_1 = -\frac{r}{3} + 2 \cdot \sqrt{\frac{|p|}{3}} \cos\left(\frac{\varphi}{3}\right) \quad (2.19)$$

$$x_2 = -\frac{r}{3} - 2 \cdot \sqrt{\frac{|p|}{3}} \cos\left(\frac{\varphi-\pi}{3}\right) \quad (2.20)$$

$$x_3 = -\frac{r}{3} - 2 \cdot \sqrt{\frac{|p|}{3}} \cos\left(\frac{\varphi+\pi}{3}\right) \quad (2.21)$$

2.1.3 Algoritma Viete's

Dalam algoritma Viete's yang diusulkan oleh matematikawan Perancis, Farniscus Vieta (1540-1603), pencarian akar polinomial dilakukan menggunakan nilai [16] :

$$\omega_1 = \frac{-1+i\sqrt{3}}{2} \quad (2.22)$$

$$\omega_2 = \frac{-1-i\sqrt{3}}{2} \quad (2.23)$$

Langkah-langkah algoritma Viete's yaitu :

1. Menginputkan persamaan polinomial kubik monoid (Persamaan 2.10) dan mengubahnya menjadi persamaan tereduksi (Persamaan 2.12)

2. Menghitung

$$c = -\frac{q}{2} + \sqrt{\frac{p^3}{27} + \frac{q^2}{4}} \quad (2.24)$$

$$r = c^{\frac{1}{3}} \quad (2.25)$$

$$d = -\frac{p}{3} \quad (2.26)$$

3. Menentukan akar-akar polinomial (x_1, x_2 dan x_3) :

$$x_1 = r + \frac{d}{r} \quad (2.27)$$

$$x_2 = \omega_1 r + \frac{d}{\omega_1 r} \quad (2.28)$$

$$x_3 = \omega_2 r + \frac{d}{\omega_2 r} \quad (2.29)$$

2.1.4 Algoritma Bairstow

Algoritma Bairstow merupakan metode iteratif yang dapat digunakan untuk mencari akar polinomial berderajat n . Langkah-langkah algoritma Bairstow adalah [5] :

1. Menentukan nilai awal u dan v . Pada kebanyakan kasus, nilai $u_i = v_i = 0$ atau dapat ditentukan dengan :

$$u = \frac{a_{n-1}}{a_{n-2}} \text{ dan } v = \frac{a_n}{a_{n-2}} \quad (2.30)$$

dimana a merupakan koefisien dari polinomial derajat n .

2. Menghitung nilai b_1, b_2, \dots, b_n dengan rumus berikut :

$$b_k = a_k - b_{k-1}u - b_{k-2}v \quad (2.31)$$

dengan nilai $k = 2, 3, \dots, n$.

3. Menghitung nilai c_1, c_2, \dots, c_{n-1} dengan rumus berikut :

$$c_k = b_k - c_{k-1}u - c_{k-2}v \quad (2.32)$$

dengan nilai $k = 2, 3, \dots, n-1$.

4. Nilai Δu dan Δv didapat dengan menghitung :

$$\Delta u = \frac{\begin{vmatrix} b_n & c_{n-2} \\ b_{n-1} & c_{n-3} \end{vmatrix}}{\begin{vmatrix} c_{n-2} & c_{n-3} \\ c_{n-1} & c_{n-2} \end{vmatrix}} \quad (2.33)$$

$$\Delta v = \frac{\begin{vmatrix} c_{n-1} & b_n \\ c_{n-2} & b_{n-1} \end{vmatrix}}{\begin{vmatrix} c_{n-2} & c_{n-3} \\ c_{n-1} & c_{n-2} \end{vmatrix}} \quad (2.34)$$

5. Menghitung nilai u dan v untuk iterasi berikutnya.

$$u_{i+1} = u_i + \Delta u_i \quad (2.35)$$

$$v_{i+1} = v_i + \Delta v_i \quad (2.36)$$

6. Mengulang langkah poin 2 hingga didapat nilai Δu dan Δv mendekati nol dan memenuhi nilai ϵ sebagai berikut:

$$|\Delta u_{i+1}| + |\Delta v_{i+1}| < \epsilon \quad (2.37)$$

7. Apabila poin 6 telah terpenuhi, maka nilai u dan v yang telah didapat disubstitusikan ke dalam persamaan berikut:

$$(x^2 + ux + v)(x^{n-2} + b_1x^{n-3} + b_2x^{n-4} + \dots + b_{n-3}x + b_{n-2}) + \text{remainder} \quad (2.38)$$

Menghitung akar dari persamaan kuadrat yang dihasilkan menggunakan rumus kuadrat seperti yang tertera pada Persamaan 2.8 sehingga akar 1 dan akar 2 dari persamaan polinomial telah didapat.

8. Perhitungan akar berikutnya dilakukan dengan mengulangi langkah 1 untuk persamaan polinomial tereduksi dan sisa yang dihasilkan pada nilai akhir merupakan nilai b .**2.1.5 Revisi Algoritma Bairstow**

Revisi algoritma Bairstow merupakan pengembangan algoritma Bairstow. Persamaan polinomial dapat dibagi dengan faktor kuadrat $x^2 + ux + v$ untuk mendapatkan persamaan polinomial berderajat $n-2$ ditambah sisa [5]. Revisi algoritma Bairstow mengembangkan solusi untuk Δu dan Δv sebagai berikut [5] :

$$\Delta u = \frac{\begin{vmatrix} b_{n-1} & c_{n-3} \\ b_n & c_{n-2} \end{vmatrix}}{\begin{vmatrix} c_{n-2} & c_{n-3} \\ (c_{n-1} - b_{n-1}) & c_{n-2} \end{vmatrix}} \quad (2.39)$$

$$\Delta v = \frac{\begin{vmatrix} c_{n-2} & b_{n-1} \\ (c_{n-1} - b_{n-1}) & c_{n-3} \end{vmatrix}}{\begin{vmatrix} c_{n-2} & c_{n-3} \\ (c_{n-1} - b_{n-1}) & c_{n-2} \end{vmatrix}} \quad (2.40)$$

Langkah-langkah yang dilakukan dalam revisi algoritma Bairstow sama dengan langkah-langkah algoritma Bairstow. Namun, nilai Δu dan Δv yang digunakan dalam revisi algoritma Bairstow menggunakan Persamaan 2.39 dan 2.40.

2.1.6 Algoritma Muller

Algoritma Muller merupakan generalisasi dari metode secant tetapi dalam penerapannya menggunakan tiga poin interpolasi kuadrat [6]. Metode iterasi ini konvergen di dekat suatu akar, tidak memerlukan pengevaluasian turunan fungsinya, dan memperoleh akar-akar nyata maupun kompleks meskipun akar-akar tersebut tidak sederhana [8]. Algoritma Muller dapat diterapkan dengan langkah-langkah berikut ini [17]:

1. Menentukan tiga poin tebakan awal yaitu x_0, x_1, x_2 2. Menghitung nilai h_0, h_1, h_2 dimana :

$$h_0 = x_0 - x_2 \quad (2.41)$$

$$h_1 = x_1 - x_2 \quad (2.42)$$

$$h_2 = h_0 * h_1 * (h_0 - h_1) \quad (2.43)$$

3. Menghitung $f(x_0), f(x_1), f(x_2)$.4. Menghitung nilai koefisien a, b, c :

$$a = \frac{h_1(f(x_0) - f(x_2)) - h_0(f(x_1) - f(x_2))}{h_2} \quad (2.44)$$

$$b = \frac{h_0^2(f(x_1) - f(x_2)) - h_1^2(f(x_0) - f(x_2))}{h_2} \quad (2.45)$$

$$c = f(x_2) \quad (2.46)$$

5. Menghitung nilai diskriminan dari perhitungan di atas

$$D = \sqrt{b^2 - 4ac} \quad (2.47)$$

6. Menentukan nilai akar polinomial untuk menghasilkan pendekatan baru dari $f(x)$:

$$x_3 = x_2 - \frac{2c}{b \pm D} \quad (2.48)$$

Tanda pada penyebut dipilih untuk mendapatkan nilai absolut atau modulus yang terbesar (apabila $b > 0$ pilih "+", apabila $b < 0$, maka pilih "-", dan apabila $b = 0$, maka pilih salah satu tanda. Setelah x_3 ditemukan, poin sebelumnya yaitu x_0 diabaikan sehingga tersisa tiga tebakan akar yaitu x_1, x_2 , dan x_3 yang kemudian digunakan dalam proses selanjutnya.

7. Menghitung $y = c$, apabila $y > \epsilon$, maka dilakukan perulangan langkah 2 hingga 6 hingga $y < \epsilon$.

Apabila $y < \epsilon$, maka iterasi dihentikan. Nilai akar polinomial adalah x_3 . Polinomial direduksi dengan membagi persamaan polinomial dengan akar yang ditemukan. Pencarian akar selanjutnya dilakukan menggunakan persamaan polinomial hasil reduksi kemudian mengulang langkah 1 hingga 7.

2.2 Galat

Output komputasi numerik biasanya menghasilkan nilai hampiran dari nilai penyelesaian persamaan yang sebenarnya [18]. Galat atau kesalahan (*error*) didefinisikan sebagai selisih nilai sejati dengan nilai hampiran. Jika \bar{x} merupakan aproksimasi dari nilai sebenarnya x , maka galat relatif dapat dihitung dengan :

$$e_{rel} = \frac{\text{Absolute error}}{\text{True value}} = \frac{e_{abs}}{x} = \frac{x - \bar{x}}{x}, x \neq 0 \quad (2.49)$$

Penelitian ini mengusulkan perhitungan bilangan kompleks dimana perhitungan galat relatif suatu akar kompleks $z_1 = p_1 + q_1 i$ dengan i satuan *imaginer* dari suatu nilai eksak $z = p + qi$ adalah rasio norm dari selisih eksak dengan hampiran dengan norm dari eksak dan disajikan dalam bentuk persen.

$$\text{galat} = \frac{\|z - z_1\|}{\|z\|} \times 100\% \quad (2.50)$$

commit to user

2.3 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak merupakan suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisa kebutuhan pengguna, menentukan spesifikasi dari kebutuhan pengguna, desain, pengkodean, pengujian hingga pemeliharaan sistem setelah digunakan [19]. Model rekayasa perangkat lunak umumnya mengacu pada *Software Development Life Cycle* (SDLC) yang memiliki tahapan analisis (*analysis*), desain (*design*), implementasi (*implementasi*), pengujian (*testing*), dan perawatan (*maintenance*) [19].

2.3.1 Analisis

Analisis sistem merupakan sebuah teknik pemecahan masalah yang menguraikan sebuah sistem menjadi komponen-komponen dengan tujuan mempelajari seberapa baik komponen-komponen tersebut bekerja dan berinteraksi untuk mencapai tujuan [19].

2.3.2 Desain

Desain perangkat lunak merupakan tugas, tahapan, atau aktivitas yang difokuskan pada spesifikasi rinci dari solusi berbasis komputer. Desain perangkat lunak fokus pada sisi teknis dan implementasi sebuah perangkat lunak [19]. Desain perangkat lunak dapat dilakukan menggunakan *flowchart* maupun pemodelan menggunakan UML.

A. Flowchart

Bagan alir sistem (*system flowchart*) merupakan bagan yang menunjukkan arus pekerjaan secara keseluruhan dari sistem. Bagan ini menjelaskan urutan-urutan dari prosedur-prosedur yang ada dalam sistem [20].

B. UML

Unified Modelling Language merupakan seperangkat aturan dan notasi untuk spesifikasi sistem perangkat lunak yang dikelola dan diciptakan oleh *Object Management Group*. Notasi tersebut berupa elemen grafis untuk memodelkan bagian-bagian sistem [21].

2.3.3 Implementasi

Implementasi merupakan tahapan menerjemahkan hasil desain logis dan fisik ke dalam kode-kode program komputer [19].

2.3.4 Pengujian

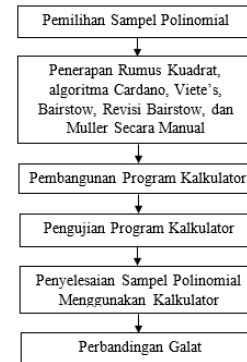
Pengujian perangkat lunak merupakan sebuah proses terhadap aplikasi/program untuk menemukan segala kesalahan dan segala kemungkinan yang akan menimbulkan kesalahan sesuai dengan spesifikasi perangkat lunak yang telah ditentukan sebelum aplikasi tersebut diserahkan kepada pelanggan [22].

2.3.5 Perawatan

Perawatan dapat dilakukan dengan berbagai tipe seperti perawatan *corrective*, perawatan *routine*, dan perawatan *system upgrade* [19].

3. METODE PENELITIAN

Tahap-tahap penelitian dapat dilihat dalam Gambar 3.1.



Gambar 3.1 Alur metodologi

3.1 Pemilihan Sampel Polinomial

Sampel polinomial berupa persamaan polinomial berderajat dua, tiga, empat, dan n yang didapat dari beberapa buku.

3.2 Penerapan Algoritma Secara Manual

Sampel polinomial terpilih diselesaikan secara manual dengan menerapkan algoritma-algoritma penyelesaian yang akan digunakan dalam program kalkulator. Hasil dari contoh perhitungan ini akan digunakan sebagai dasar implemenasi algoritma dalam program.

3.3 Pembangunan Program Kalkulator

3.3.1 Analisis

Analisa kebutuhan perangkat lunak, baik kebutuhan fungsional maupun kebutuhan non fungsional, dilakukan sebelum program kalkulator dibangun.

3.3.2 Perancangan

Perancangan program kalkulator dilakukan menggunakan pendekatan UML dan *flowchart*.

3.3.3 Implementasi

Hasil analisa dan perancangan program kalkulator diimplementasikan menggunakan Java Swing GUI. Algoritma penyelesaian dalam program kalkulator adalah :

- Rumus kuadrat untuk polinomial derajat 2
 - Cardano, Viete's, Bairstow, revisi Bairstow untuk polinomial derajat 3
 - Bairstow, revisi Bairstow, Muller untuk polinomial derajat 4
- Polinomial derajat n diselesaikan dengan kombinasi algoritma seperti yang tertera dalam Tabel 3.1.

Tabel 3.1 Algoritma Penyelesaian Polinomial Derajat n

Polinomial derajat n , dengan n bilangan ganjil	Polinomial derajat n , dengan n bilangan genap
Muller	Muller
Muller – Cardano	Muller – Bairstow
Muller – Bairstow	Muller – Revisi Bairstow
Muller – Revisi Bairstow	Bairstow
Bairstow	Bairstow – Revisi Bairstow
Bairstow – Revisi Bairstow	Bairstow – Muller
Bairstow – Cardano	Revisi Bairstow
Bairstow – Muller	Revisi Bairstow – Bairstow
Revisi Bairstow	Revisi Bairstow – Muller
Revisi Bairstow – Bairstow	
Revisi Bairstow – Cardano	
Revisi Bairstow – Muller	

3.4 Pengujian Program Kalkulator

Pengujian program dilakukan dengan tujuan untuk mengetahui kualitas dan kinerja program yang dibangun. Pada penelitian ini, pengujian program dilakukan dengan menggunakan metode *black box* dimana pengujian dilakukan pada setiap fungsional yang terdapat pada program.

3.5 Penyelesaian Sampel Polinomial

Penyelesaian sampel-sampel polinomial yang telah ditentukan menggunakan program kalkulator dilakukan setelah program kalkulator pencarian akar kompleks polinomial selesai dibangun.

3.6 Perbandingan Galat

Galat yang dihasilkan kemudian dibandingkan satu sama lain. Algoritma dengan galat rendah dapat dikategorikan sebagai algoritma yang baik. Pada algoritma Bairstow, Revisi Bairstow, dan Muller, perbandingan juga dilihat dari banyaknya iterasi yang dilakukan. Ketiga algoritma tersebut dikenakan delta toleransi sebesar $1E-8$. Dengan demikian, maka akan diketahui algoritma yang tercepat dengan galat terendah.

4. HASIL DAN PEMBAHASAN

4.1 Sampel Polinomial

Sampel polinomial yang digunakan dalam penelitian ditampilkan dalam Tabel 4.1.

Tabel 4.1 Sampel polinomial

No	Derajat Polinomial	Sampel Polinomial
1	Polinomial derajat 2	$x^2 + 18x + 81$
		$x^2 - 13x - 30$
		$x^2 + 10x - 14$
		$x^2 + 5x + 11$
		$x^2 - 3x + 5$
2	Polinomial derajat 3	$x^3 - 8x^2 + 19x + 12$
		$x^3 + 3x^2 - 2x + 7$
		$x^3 + 3x^2 + 3x + 2$
		$x^3 + 2x^2 - 5x - 6$
		$x^3 - 4x^2 - 2x + 3$
3	Polinomial derajat 4	$x^4 + 7x^3 - x^2 + 7x - 1$
		$x^4 - 2x^3 + 2x^2 - 7x + 1$
		$x^4 - 16x^3 + 86x^2 - 176x + 105$
		$x^4 - 5x^3 + 13x^2 - 19x + 10$
4	Polinomial derajat n	$x^7 + 3x^6 - 4x^5 - 2x^3 + 2$
		$x^8 - x^7 - 39x^6 + 37x^5 + 446x^4 - 108x^3 - 1928x^2 - 256x + 1920$

4.2 Perhitungan Manual Algoritma

Beberapa sampel pada Tabel 4.1 dihitung secara manual menggunakan algoritma penyelesaian polinomial derajat n yang tersedia pada program kalkulator. Hasil perhitungan dijadikan dasar pembuatan kalkulator. Berikut ini merupakan penyelesaian polinomial $x^3 - 4x^2 - 2x + 3$ menggunakan algoritma Cardano :

1. Bentuk persamaan kubik $x^3 + rx^2 + sx + t = 0$, maka diketahui $r = -4$, $s = -2$, dan $t = 3$.

2. Mensubstitusikan $y = x + \frac{r}{3}$ pada persamaan kubik, sehingga menghasilkan persamaan tereduksi $y^3 + py + q = 0$.

$$p = \frac{3s - r^2}{3} = \frac{3(-2) - (-4)^2}{3} = -7.3$$

$$q = \frac{2r^3 - rs}{27} + t = \frac{2(-4)^3 - (-4)(-2)}{27} + 3 = -4.4$$

3. Menghitung nilai diskriminan

$$D = \left(\frac{p}{3}\right)^3 + \left(\frac{q}{2}\right)^2$$

$$D = \left(\frac{-7.3}{3}\right)^3 + \left(\frac{-4.4}{2}\right)^2$$

$$D = -9.8$$

4. Karena $D < 0$, maka kasus ini tergolong *irreducible case* sehingga pencarian akar kompleks polinomial menerapkan $\cos \varphi = -\frac{q}{2\sqrt{(|p|/3)^3}}$, berikut ini detail perhitungannya :

$$\cos \varphi = -\frac{q}{2\sqrt{(|p|/3)^3}} = -\frac{-4.4}{2\sqrt{(|-7.3|/3)^3}}$$

$$\cos \varphi = -\frac{(-4.4)}{2\sqrt{(|-7.3|/3)^3}}$$

$$\cos \varphi = -\frac{(-4.4)}{2\sqrt{(2.4)^3}}$$

$$\cos \varphi = 0.576611$$

Sehingga,

$$\varphi = 0.95622171$$

Dengan demikian, akar-akar polinomial adalah :

- $x_1 = -\frac{r}{3} + 2 \cdot \sqrt{\frac{|p|}{3}} \cos\left(\frac{\varphi}{3}\right)$
 $x_1 = -\frac{(-4)}{3} + 2 \cdot \sqrt{\frac{|-7.3|}{3}} \cos\left(\frac{0.95622171}{3}\right)$
 $x_1 = \frac{4}{3} + 2 \cdot \sqrt{\frac{7.3}{3}} \cos(0.318741)$
 $x_1 = 4.302776$
- $x_2 = -\frac{r}{3} - 2 \cdot \sqrt{\frac{|p|}{3}} \cos\left(\frac{\varphi - \pi}{3}\right)$
 $x_2 = -\frac{(-4)}{3} - 2 \cdot \sqrt{\frac{|-7.3|}{3}} \cos\left(\frac{0.95622171 - 3.14159}{3}\right)$
 $x_2 = \frac{4}{3} - 2 \cdot 1.56347192 \cdot \cos(-0.728456981)$
 $x_2 = -1$
- $x_3 = -\frac{r}{3} - 2 \cdot \sqrt{\frac{|p|}{3}} \cos\left(\frac{\varphi + \pi}{3}\right)$
 $x_3 = -\frac{(-4)}{3} - 2 \cdot \sqrt{\frac{|-7.3|}{3}} \cos\left(\frac{0.95622171 + 3.14159}{3}\right)$
 $x_3 = \frac{4}{3} - 2 \cdot 1.56347192 \cdot \cos(1.365938)$
 $x_3 = 0.697224$

Dengan demikian, berdasarkan algoritma Cardano, polinomial $x^3 - 4x^2 - 2x + 3 = 0$ memiliki tiga akar diantaranya adalah 4.302776 + 0.0i, -1.0 + 0.0i, dan 0.697224 + 0.0i.

4.3 Pembangunan Program Kalkulator

4.3.1 Analisis

Analisa kebutuhan program kalkulator dilakukan untuk mengetahui kebutuhan fungsional dan kebutuhan non fungsional. Kebutuhan fungsional program ditunjukkan pada Tabel 4.2, sedangkan kebutuhan non fungsional tertera pada Tabel 4.3.

Tabel 4.2 Kebutuhan Fungsional Kalkulator Pencarian Akar Kompleks Polinomial Derajat n

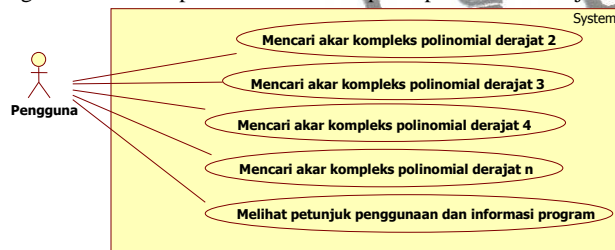
Kode	Kebutuhan Fungsional
F.1	Pengguna dapat mencari akar-akar kompleks polinomial derajat dua
F.2	Pengguna dapat mencari akar-akar kompleks polinomial derajat tiga
F.3	Pengguna dapat mencari akar-akar kompleks polinomial derajat empat
F.4	Pengguna dapat mencari akar-akar kompleks polinomial derajat n
F.5	Pengguna dapat melihat petunjuk penggunaan dan informasi program

Tabel 4.3 Kebutuhan Non Fungsional Kalkulator Pencarian Akar Kompleks Polinomial Derajat n

Kode	Kebutuhan Non Fungsional
NF.1	Desain antarmuka yang <i>user friendly</i> dan sederhana, sehingga mudah digunakan untuk orang awam
NF.2	Program dapat terkoneksi dengan program Matlab
NF.3	Program dapat menyelesaikan persamaan polinomial dalam waktu singkat

4.3.2 Perancangan

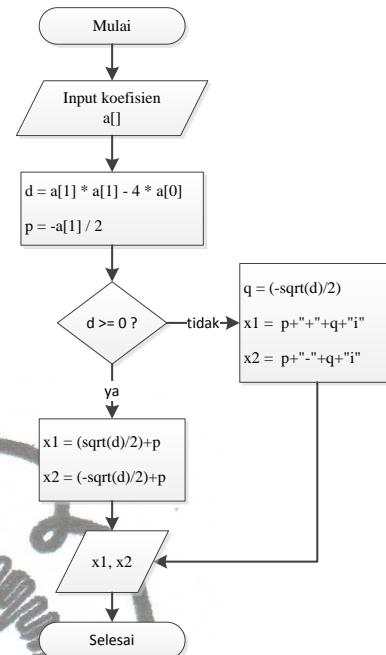
Perancangan program kalkulator dilakukan dengan menggunakan pendekatan UML. Gambar 4.1 menunjukkan *use case diagram* program kalkulator pencarian akar kompleks polinomial derajat n .



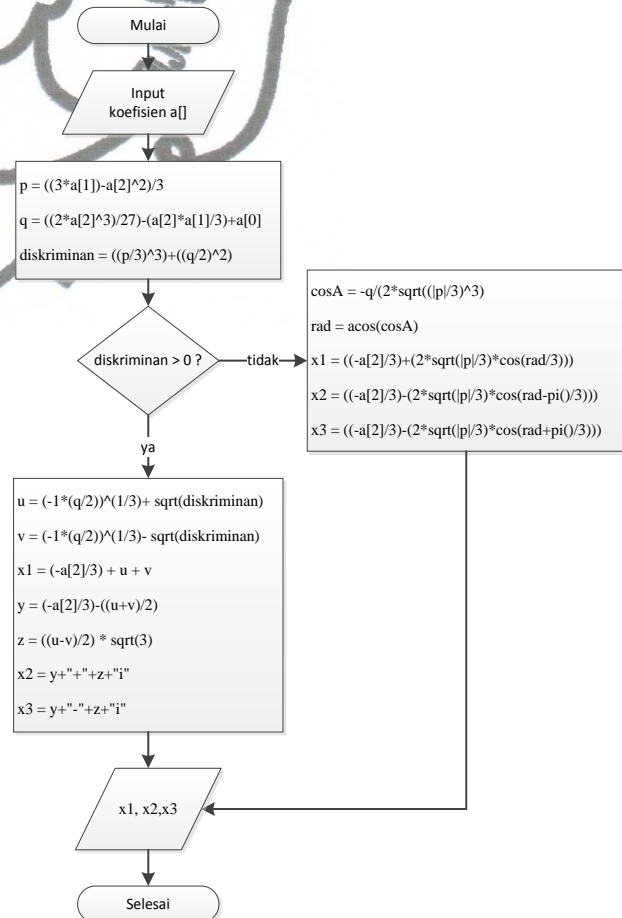
Gambar 4.1 Use Case Diagram Program Kalkulator

4.3.3 Implementasi

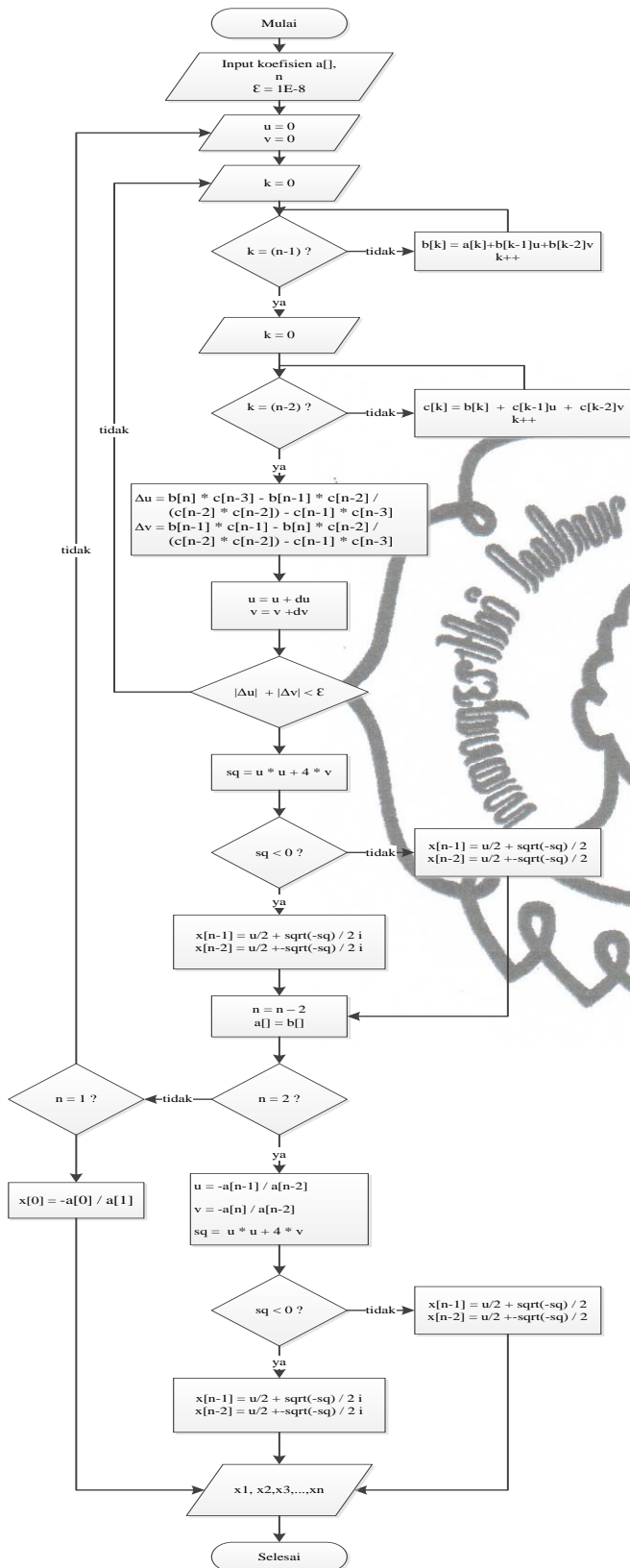
Implementasi program kalkulator dilakukan menggunakan Bahasa pemrograman Java dengan tampilan Java Swing GUI. Gambar 4.2 merupakan *flowchart* rumus kuadrat, Gambar 4.3 merupakan *flowchart* Cardano, Gambar 4.4 merupakan *flowchart* Bairstow, Gambar 4.5 merupakan *flowchart* Revisi Bairstow, Gambar 4.6 merupakan *flowchart* Muller, dan Gambar 4.7 merupakan *flowchart* Viète's,



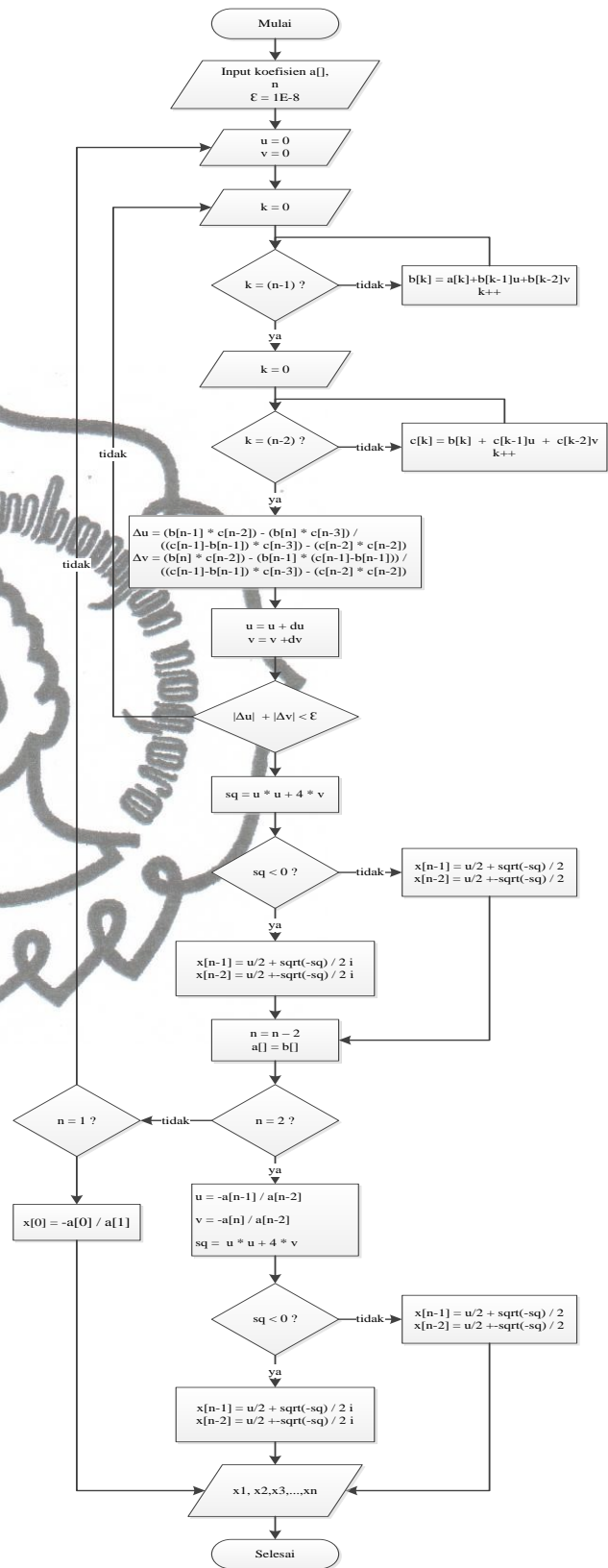
Gambar 4.2 Flowchart Rumus Kuadrat



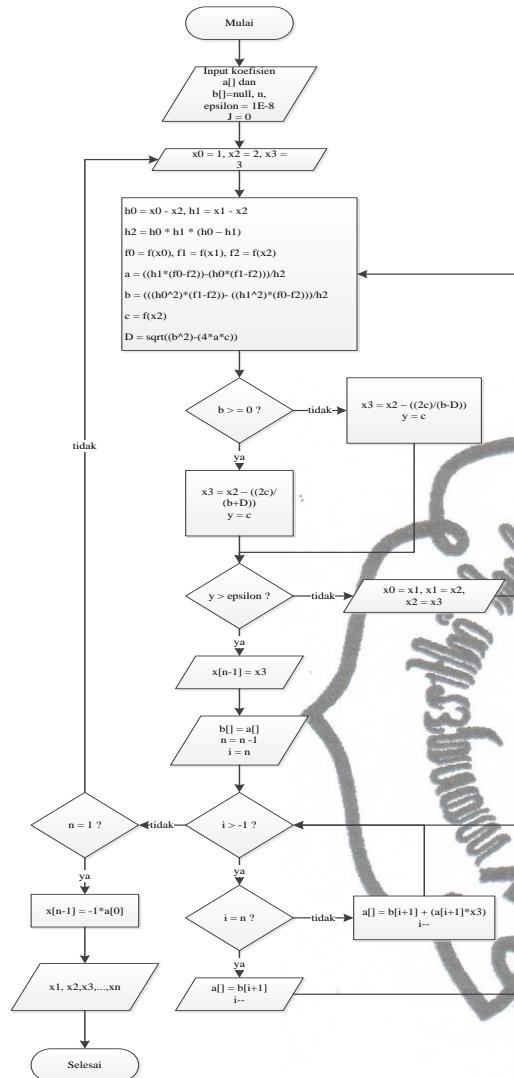
Gambar 4.3 Flowchart Algoritma Cardano



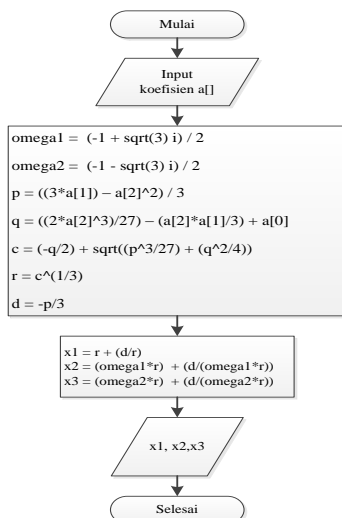
Gambar 4.4 *Flowchart* Algoritma Bairstow



Gambar 4.5 *Flowchart* Algoritma Revisi Bairstow



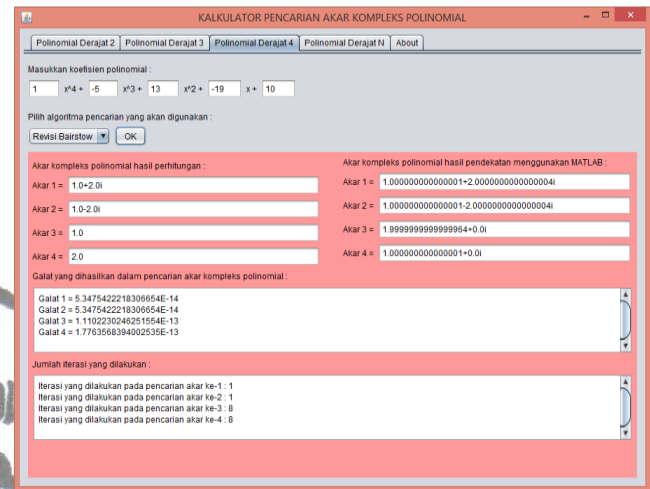
Gambar 4.6 Flowchart Algoritma Muller



Gambar 4.7 Flowchart Algoritma Viete's

4.4 Hasil Implementasi Program Kalkulator

Gambar 4.8 merupakan hasil implementasi program kalkulator berupa tampilan antarmuka program kalkulator pencarian akar kompleks polinomial.



Gambar 4.8 Antarmuka program kalkulator pencarian akar kompleks polinomial

4.5 Pengujian Program Kalkulator

Pengujian program kalkulator dilakukan secara manual menggunakan *black box testing* untuk mengetahui kualitas kinerja program kalkulator. Hasil pengujian menunjukkan seluruh fungsional pada program kalkulator dapat berfungsi sebagaimana mesinya. Dari seluruh fungsional dengan jumlah keseluruhan 17 kasus uji, 100% fungsional sistem dapat berjalan dengan baik. Hasil pengujian dapat dilihat pada Tabel 4.4.

Tabel 4.4 Hasil Pengujian Kalkulator menggunakan *Blackbox*

No	Deskripsi	Jumlah Kasus Uji	Hasil	
			Diterima	Tidak diterima
1	Mencari akar kompleks polinomial derajat 2	4	4	0
2	Mencari akar kompleks polinomial derajat 3	4	4	0
3	Mencari akar kompleks polinomial derajat 4	4	4	0
4	Mencari akar kompleks polinomial derajat n	6	6	0
5	Melihat petunjuk penggunaan dan informasi program	1	1	0
Jumlah		19	19	0

4.6 Penyelesaian Polinomial dan Analisa Perbandingan Galat

4.6.1 Polinomial Derajat Dua

Pencarian akar kompleks polinomial derajat dua dilakukan menggunakan rumus kuadrat pada kelima sampel menunjukkan rata-rata galat yang dihasilkan sangat kecil yaitu sebesar 2.81952966549601E-07%. Hasil pencarian akar kompleks polinomial derajat dua dapat dilihat pada Tabel 4.5.

Tabel 4.5 Hasil pencarian akar kompleks derajat dua

No	Sampel Polinomial	Perhitungan Rumus Kuadrat		Perhitungan Matlab
		Akar	Galat(%)	Akar
1.	$x^2 + 18x + 81$	-9.0	1.409765E-6	-9.000000
		-9.0	1.409765E-6	-8.999999
2.	$x^2 - 13x - 30$	15.0	0.0	15.0+0.0i
		-2.0	0.0	-2.0+0.0i
3.	$x^2 + 10x - 14$	-11.244998	1.783497E-14	-11.244998
		1.244998	1.579686E-14	1.244998
4.	$x^2 + 5x + 11$	-2.5+2.179449i	2.146547E-13	-2.500000+2.179449i
		-2.5-2.179449i	2.146547E-13	-2.500000-2.179449i
5.	$x^2 - 3x + 5$	1.5+1.65831i	9.930137E-15	1.500000+1.658312i
		1.5-1.65831i	9.930137E-15	1.500000-1.658312i

4.6.2 Polinomial Derajat Tiga

Penyelesaian polinomial derajat tiga dilakukan menggunakan algoritma Cardano, Viete's, Bairstow, dan revisi Bairstow. Rata-rata yang dihasilkan pada penyelesaian lima sampel polinomial derajat tiga dapat dilihat pada Tabel 4.6. Rata-rata galat didapat dari jumlah seluruh galat yang dihasilkan dibagi banyaknya akar yang dihasilkan.

Tabel 4.6 Rata-rata galat pencarian akar kompleks polinomial derajat tiga

Algoritma	Rata-Rata galat (%)
Cardano	1.43568059121049E-13
Viete's	64.15201457
Bairstow	1.92955658379859E-09
Revisi Bairstow	1.48444243802628E-08

Berdasarkan Tabel 4.6 algoritma terbaik dalam menyelesaikan polinomial derajat tiga adalah algoritma Cardano dengan rata-rata galat yaitu 1.43568059121049E-13%.

4.6.3 Polinomial Derajat Empat

Tabel 4.7 menampilkan rata-rata galat yang dihasilkan pada pencarian akar kompleks polinomial derajat empat. Berdasarkan Tabel 4.7, ketiga algoritma dapat menyelesaikan sampel-sampel polinomial derajat empat dengan sangat baik dimana rata-rata galat yang dihasilkan sangat rendah dan rata-rata iterasi yang dilakukan sedikit. Algoritma terbaik dalam penyelesaian polinomial derajat empat adalah algoritma Revisi Bairstow dengan rata-rata galat yaitu 1.34421873307271E-09% dan rata-rata iterasi yaitu 4.

Tabel 4.7 Rata-rata galat dan iterasi pencarian akar kompleks polinomial derajat empat

Algoritma	Galat (%)	Iterasi
Bairstow	2.7918843298595E-08	5
Revisi Bairstow	1.34421873307271E-09	4
Muller	4.09468833907407E-06	6

4.6.4 Polinomial Derajat N

Pencarian akar kompleks polinomial derajat n dilakukan dengan menggunakan algoritma Bairstow, revisi Bairstow, Muller, serta kombinasi algoritma. Dalam mencari akar kompleks polinomial, algoritma Bairstow, revisi Bairstow, dan Muller akan mereduksi

polinomial hingga semua akar didapat. Pada algoritma Bairstow, reduksi polinomial dilakukan berdasarkan polinomial kuadrat yang dapat setelah variabel u dan v diketahui. Berikut ini reduksi polinomial berdasarkan algoritma Bairstow :

$$(x^2 + ux + v)(x^{n-2} + b_1x^{n-3} + b_2x^{n-4} + \dots + b_{n-3}x + b_{n-2}) + remainder$$

polinomial tereduksi

Pada algoritma Muller, akar kompleks polinomial didapat satu per satu pada setiap proses, sehingga polinomial tereduksi didapat dengan membagi persamaan polinomial dengan akar yang telah didapat. Reduksi polinomial ini yang menjadi dasar kombinasi algoritma. Dengan demikian, pencarian akar kompleks polinomial pada kombinasi algoritma awalnya dilakukan dengan menggunakan algoritma 1 kemudian polinomial direduksi hingga didapat polinomial derajat tiga atau empat. Selanjutnya, pencarian akar polinomial derajat tiga atau empat tersebut dilakukan menggunakan algoritma 2. Tabel 4.8 menampilkan rata-rata galat dan iterasi yang dihasilkan pada pencarian akar kompleks polinomial derajat n .

Tabel 4.8 Rata-rata galat dan iterasi pada pencarian akar kompleks polinomial derajat n

No	Kombinasi Algoritma	Galat (%)	Iterasi
1	Bairstow	2.62347809508259E-09	12
2	Bairstow – Revisi Bairstow	8.8918779643174E-09	10
3	Bairstow – Muller	1.53555473779719E-06	19
4	Bairstow – Cardano	5.51058633925391E-10	12
5	Revisi Bairstow	1.69204593737167E-08	8
6	Revisi Bairstow – Bairstow	2.84040213353398E-09	19
7	Revisi Bairstow – Muller	7.08557635107149	8
8	Revisi Bairstow – Cardano	3.72289154574655E-09	7
9	Muller	2.99043000191301E-05	24
10	Muller – Bairstow	2.09544651882645	13
11	Muller – Revisi Bairstow	2.09544649363791	21
12	Muller – Cardano	4.27781615793958E-13	10

Berdasarkan Tabel 4.8, diketahui bahwa algoritma terbaik dalam menyelesaikan polinomial derajat n adalah Muller – Cardano dengan rata-rata galat yaitu 4.27781615793958E-13%. Namun, berdasarkan rata-rata jumlah iterasi, algoritma Revisi Bairstow – Cardano lebih unggul dengan rata-rata jumlah iterasi yaitu 7 iterasi.

5. KESIMPULAN DAN SARAN

Berdasarkan penelitian yang dilakukan, dapat disimpulkan bahwa:

- Media pembelajaran kalkulator pencarian akar kompleks polinomial derajat n telah berhasil dibangun. Perancangan kalkulator dilakukan dengan menggunakan pendekatan UML dan *flowchart*. Program kalkulator diimplementasikan menggunakan bahasa pemrograman Java. Hasil pengujian

commit to user

menggunakan metode *black box* menunjukkan bahwa 100% fungsional program kalkulator dapat berjalan dengan baik.

2. Algoritma terbaik dalam menyelesaikan polinomial derajat tiga adalah algoritma Cardano dengan rata-rata galat sebesar 1.43568059121049E-13%.
3. Algoritma terbaik dalam menyelesaikan polinomial derajat empat adalah revisi algoritma Bairstow yang menghasilkan rata-rata galat yaitu 1.34421873307271E-09% dan rata-rata jumlah iterasi yang dilakukan adalah 4.
4. Algoritma terbaik dalam penyelesaian polinomial derajat n adalah kombinasi algoritma Muller – Cardano dimana nilai rata-rata galat yaitu 4.27781615793958E-13%. Namun, berdasarkan rata-rata jumlah iterasi yang dilakukan, kombinasi algoritma Revisi Bairstow – Cardano lebih unggul dengan rata-rata jumlah iterasi yaitu tujuh iterasi.

Untuk pengembangan penelitian selanjutnya disarankan untuk dilakukan penelitian lebih lanjut mengenai media pembelajaran untuk model matematika, khususnya polinomial. Diharapkan kalkulator media pembelajaran dapat menyelesaikan persamaan polinomial dengan derajat yang lebih tinggi. Algoritma yang digunakan dalam media pembelajaran dapat diperluas dengan menggunakan metode numerik lainnya seperti metode Newton-Raphson dan Secant, sehingga dapat memperluas wawasan mengenai algoritma-algoritma tersebut.

6. DAFTAR PUSTAKA

- [1] Heinich, R., Molenda, M., & Russel, J. D., *“Instructional technology for teaching and learning: Designing instruction, integrating computers and using media”*, Upper Saddle River, NJ.: Merril Prentice Hall, 1996.
- [2] Susilana, R., and Riyana, C., *“Media Pembelajaran: Hakikat, Pengembangan, Pemanfaatan, dan Penilaian”*, CV. Wacana Prima, 2009.
- [3] Supriyanto, W. and Muhsin, A., *“Teknologi informasi perpustakaan”*, Kanisius, Yogyakarta, 2008
- [4] Munir, Rinaldi, *“Metode Numerik”*, Bandung : Informatika Bandung, 2003.
- [5] James, M.L. Smith, G.M. and Welford, J.C., *“Applied Numerical Methods for Digital Computation”*, New York: Harper & Row, 2000.
- [6] Press, W. H., Flannery, B. P., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P., *“Numerical Recipes: The Art of Scientific Computing (FORTRAN)”*. Cambridge University Press, New York, 1989.
- [7] Windra, I. Nst, M.L. and Dewi, M.P., *“Menentukan akar kompleks polinomial dengan Metode Bairstow, UNP Journal of Mathematics, 2(1), 2013.*
- [8] Conte, S.D. and Boor, C.W.D., *“Elementary numerical analysis: an algorithmic approach”*, 2nd edn, Amerika: McGraw-Hill Companies, Inc, 1972.
- [9] Mariana, Hendun, *“Aplikasi Metode Muller dan Metode Bairstow dengan bantuan Matlab dalam menentukan akar-akar polinomial”*, Skripsi, Universitas Islam Negeri (UIN) Malang, 2007.
- [10] Nickalls, R.W.D., *“A new approach to solving the cubic: Cardan’s solution revealed”*, *The Mathematical Gazette*, 77(480), pp.354-359, 1993.
- [11] Ming, OR. C., *“Solving cubic equations by Viète’s Substitutions”*, EduMath29, [on-line]: http://www.hkame.org.hk/uploaded_files/magazine/29/459.pdf.
- [12] Ayres, Frank. and Schmidt, P.A., *“Schaum’s Outlines: Matematika Universitas”*, 3rd edn, Jakarta: Erlangga, 2004.
- [13] Harris, J.W. and Stöcker, H., *“Handbook of mathematics and computational science”*, Springer Science & Business Media, 1998.
- [14] Bronshtein, I.N. and Semendyayev, K.A.F., *“Handbook of mathematics”*, 6th edn, Springer Science & Business Media, 2015.
- [15] Weisstein, Eric W., *“Cubic Formula.”* From MathWorld-- A Wolfram WebResource, [on-line]: <http://mathworld.wolfram.com/CubicFormula.html>.
- [16] Mathews, J.H., *“Tutorial Exercise for Solving Cubic Equations”*. Department of Mathematics, California State University Fullerton, [on-line]: <http://mathfaculty.fullerton.edu/mathews/cubics/A50/A00/CubicTutorial.html>.
- [17] Butt, R. *“Introduction to Numerical Analysis Using MATLAB®”*, Jones & Bartlett Learning, 2009.
- [18] Esfandiari, R.S., *“Numerical Methods for Engineers and Scientists using MATLAB®”*, CRC Press, 2013.
- [19] Mulyanto, A. R., *“Rekayasa Perangkat Lunak Jilid 1”*. Jakarta: Departemen Pendidikan Nasional, 2008.
- [20] Sikha, Bagui and Richard Earp. *“Database Desain Using Entity-Relationship Diagrams”*. Auerbach Publication, 2003.
- [21] Luthfi, H. W., & Riasti, B. K., *“Sistem Informasi Perawatan Dan Inventaris Laboratorium Pada Smk Negeri 1 Rembang Berbasis Web”*, *Indonesian Jurnal on Computer Science - Speed (IJCSS)* 15, 10(1), 2013.
- [22] Simarmata, Janer, *“Rekayasa Perangkat Lunak”*, Penerbit ANDI: Yogyakarta, 2010.