EQUALS: Jurnal Ilmiah Pendidikan Matematika
Volume 8, Nomor 2, Desember 2025
p-ISSN: 2622-2329, e-ISSN: 2622-2442

Analisis Kesalahan Mahasiswa dalam Menyusun *Flowchart* Metode Posisi Palsu Berdasarkan Teori Rittle-Johnson dan Alibali

Khadijah[1], Usman Mulbar[2]*

[1) 2)] Pendidikan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Negeri Makassar, Indonesia
[1)] *khadijah0611@gmail.com*, [2)]* *u_mulbar@unm.ac.id*

Article History
Received : 21-10-2025
Revised : 16-11-2025
Accepted : 01-12-2025

Keywords
Computational Thinking; Error Analysis Framework; False Position Method; Errors in Algorithmic Literacy;

ABSTRACT

This study aims to describe students' errors in implementing the *False Position Method* algorithm using the PHP programming language through the Computational Thinking–Error Analysis Framework (CT–EAF) approach. A descriptive qualitative design was employed by analyzing students' final projects consisting of algorithm documentation and executable code. The analysis involved data reduction, error classification, and interpretation based on computational thinking dimensions and mathematical error categories. The findings indicate that students encountered various difficulties in translating mathematical concepts into computational structures and in evaluating the accuracy of iterative results. Most errors were related to challenges in abstracting the mathematical formula of the False Position Method into program logic, as well as limited reflection and debugging of computational outcomes. Errors in abstraction mainly involved incorrect operators, parentheses, or formula translation of the False Position Method, while evaluation errors were related to the omission of numerical convergence criteria $| f(c) | < \varepsilon \; or |\Delta c| < \varepsilon$. The integration of CT–EAF provided a comprehensive perspective on the relationship between students' algorithmic thinking processes and the types of conceptual and procedural errors they made. This study highlights the importance of strengthening computational thinking and algorithmic literacy in numerical methods learning. The CT–EAF approach serves as an effective framework for guiding students to connect mathematical concepts with computational logic, fostering a learning process that emphasizes both conceptual understanding and algorithmic reasoning rather than mere computational outcomes.

Available online at:
ejournals.umma.ac.id/index.php/equals

How to Cite : Khadijah & Mulbar, U. (2025). Analisis Kesalahan Mahasiswa dalam Menyusun Flowchart Metode Posisi Palsu Berdasarkan Teori Rittle-Johnson dan Alibali. *EQUALS: Jurnal Ilmiah Pendidikan Matematika*, *8*(2), 159–171. https://doi.org/10.46918/equals.v8i2.3015

https://doi.org/10.46918/equals.v8i2.3015

PENDAHULUAN

Pemahaman algoritma numerik merupakan salah satu kompetensi penting dalam pembelajaran matematika terapan seperti dalam kurikulum program studi Pendidikan Matematika Universitas Negeri Makassar (UNM), khususnya pada konteks penyelesaian masalah yang tidak dapat diselesaikan secara analitik. Salah satu algoritma dasar yang diajarkan dalam mata kuliah *Metode Numerik* adalah metode posisi palsu (*False Position Method*), yaitu prosedur iteratif berbasis *linear interpolation* yang digunakan untuk menentukan solusi akar persamaan non-linier melalui pendekatan grafis dan analitis (Burden et al., 2016; Chapra & Canale, 2015).

Metode posisi palsu didasarkan pada prinsip perubahan tanda fungsi dalam interval [a,b], dengan syarat $f(a)f(b) < 0$, dan menghasilkan pendekatan akar melalui persamaan:

$$x_r = b - \frac{f(b)(b-a)}{f(b)-f(a)}$$

yang diulang atau dilakukan iterasi hingga memenuhi kriteria konvergensi seperti $| f(x_r) | < \varepsilon$ atau $\left| x_r^{(k)} - x_r^{(k-1)} \right| < \delta$, yang menjadi inti perhitungan dalam proses iterasi (Atkinson & Han, 2009). Dalam konteks pembelajaran algoritmik, mahasiswa tidak hanya diharapkan mampu menggunakan rumus tersebut secara mekanis, tetapi juga memahami makna geometris di baliknya, yakni sebagai perpotongan garis sekant yang menghubungkan titik $\big(a, f(a)\big)$ dan $\big(b, f(b)\big)$ terhadap sumbu-x.

Dalam praktik pembelajaran, mahasiswa sering diminta membuat *flowchart* algoritma sebagai representasi visual dari langkah-langkah metode posisi palsu. Namun, hasil observasi menunjukkan bahwa banyak mahasiswa melakukan kesalahan sistematis, seperti salah menulis kondisi $f(a)f(b) < 0$, menukar logika algoritma, atau mengabaikan kriteria berhenti iterasi. Kesalahan tersebut bukan hanya kesalahan teknis, tetapi mencerminkan ketidakterkaitan antara pengetahuan konseptual dan prosedural, serta kelemahan dalam keterampilan berpikir komputasional (*Computational Thinking*).

Dalam literatur pembelajaran matematika, kemampuan peserta didik dibangun atas dua dimensi pengetahuan utama yaitu pengetahuan konseptual (*conceptual knowledge*) dan pengetahuan prosedural (*procedural knowledge*). Pengetahuan konseptual terkait pemahaman terhadap ide, prinsip, dan relasi matematis ("*mengapa*"), sedangkan pengetahuan prosedural terkait kemampuan dalam melaksanakan langkah-langkah pengoperasian ("*bagaimana*"). Studi seperti Hurrell (Hurrell, 2021) menunjukkan bahwa mengabaikan keterkaitan kedua dimensi ini dapat menyebabkan kesalahan siswa dalam pemecahan masalah.

Namun demikian, berbagai penelitian menunjukkan bahwa mahasiswa sering mengalami kesulitan dalam mentransformasi konsep matematis ke bentuk algoritmik yang dapat diimplementasikan ke dalam bahasa pemrograman. Kesulitan ini sering muncul dalam bentuk kesalahan konseptual dan prosedural, misalnya kesalahan dalam menulis formula iteratif, logika pembaruan interval, serta penerapan kriteria penghentian (*stopping criteria*) (Mhlolo, 2018; Rittle-Johnson et al., 2001; Rittle-Johnson & Alibali, 1999). Menurut Rittle-Johnson dan Alibali (Rittle-Johnson & Alibali, 1999), kesalahan tersebut dapat dikategorikan sebagai *procedural misapplication* atau *conceptual misunderstanding*, yang menunjukkan lemahnya keterkaitan antara pengetahuan konseptual (pemahaman mengapa langkah dilakukan) dan pengetahuan prosedural (bagaimana langkah dilakukan).

Dalam konteks *Computational Thinking (CT)*, kesalahan mahasiswa pada implementasi metode posisi palsu juga dapat dianalisis melalui empat dimensi utama berpikir komputasional,

yaitu *decomposition, abstraction, algorithm design,* dan *evaluation* (Weintrop, Beheshti, et al., 2016; Wing, 2006). Setiap dimensi ini terlibat secara langsung dalam proses mahasiswa menulis program numerik, mendekomposisi rumus matematika ke dalam langkah algoritmik, mengabstraksikan logika kondisi iteratif, merancang struktur kode, serta mengevaluasi hasil keluaran. Kesalahan yang terjadi pada setiap tahap mencerminkan tingkat kedalaman pemahaman algoritmik mahasiswa.

Beberapa penelitian terdahulu juga menegaskan bahwa kemampuan algoritmik mahasiswa dalam bidang numerik masih perlu diperkuat. Nurafni dan Suyitno (Nurafni & Suyitno, 2022) menemukan bahwa banyak mahasiswa hanya meniru kode tanpa memahami logika matematikanya, sedangkan penelitian oleh Fitriyani et al. (Fitriyani et al., 2023) menunjukkan bahwa pembelajaran metode numerik berbasis CT dapat meningkatkan kemampuan *debugging* dan kesadaran prosedural mahasiswa. Oleh karena itu, analisis kesalahan pada implementasi metode posisi palsu menjadi penting, bukan hanya untuk mengidentifikasi kesalahan sintaksis dalam kode, tetapi juga untuk mengungkap pola berpikir komputasional dan hubungan antara konsep matematika dan algoritma.

Meskipun banyak penelitian yang menelaah kesalahan siswa dalam konteks aljabar, geometri, atau statistik, penelitian yang secara spesifik menganalisis kesalahan mahasiswa dalam menyusun *flowchart* algoritmik untuk metode posisi palsu masih sangat terbatas. Selain itu, sedikit yang menghubungkan secara eksplisit antara kelemahan pengetahuan konseptual metode numerik dengan kesalahan dalam representasi prosedural, khususnya *flowchart*. Hal ini menjadi celah penelitian yang hendak diisi. Dan studi kesalahan pada topik solusi akar non-linier sering berfokus pada hasil numerik atau kode program, sedangkan representasi algoritmik melalui *flowchart* dengan kerangka Rittle-Johnson & Alibali masih jarang dibahas. Maka tujuan penelitian ini yaitu mengidentifikasi bentuk kesalahan pada *flowchart* metode posisi palsu berdasarkan teori Rittle-Johnson & Alibali.

METODE

Penelitian ini menggunakan pendekatan kualitatif deskriptif, dengan fokus untuk menganalisis kesalahan mahasiswa dalam menyusun *flowchart* metode posisi palsu (*false position method*) dan menafsirkannya melalui dua kerangka utama:

1. Teori Rittle-Johnson & Alibali (Rittle-Johnson et al., 2001; Rittle-Johnson & Alibali, 1999) tentang hubungan C*onceptual Error, Procedural Error*, dan *Representational Error*
2. *Computational Thinking* (CT) (Shute et al., 2017; Wing, 2006) sebagai kerangka kognitif yang menjelaskan proses berpikir algoritmik.

Pendekatan deskriptif-analitik kesalahan berbasis *Computational Thinking–Error Analysis Framework* (CT–EAF) digunakan secara integratif untuk menggambarkan profil kesalahan mahasiswa dari dua sisi (a) tahapan berpikir algoritmik (CT) dan (b) jenis kesalahan yang dilakukan (EAF), sebagaimana direkomendasikan dalam penelitian pendidikan matematika modern (Araujo et al., 2022).

Penelitian dilaksanakan di Program Studi Pendidikan Matematika Universitas Negeri Makassar, pada Semester Genap tahun akademik 2023/2024. Subjek penelitian terdiri atas 34 mahasiswa semester IV yang mengikuti mata kuliah Metode Numerik, dengan fokus pada topik penyelesaian akar fungsi non-linier menggunakan *metode posisi palsu*. Pemilihan subjek dilakukan secara *purposive* sampling, yaitu berdasarkan pertimbangan bahwa mahasiswa telah memahami konsep dasar metode numerik (fungsi kontinu, perubahan tanda, dan iterasi),

menyusun proyek berupa *flowchart* metode posisi palsu, dan menunjukkan variasi kesalahan yang relevan untuk analisis kognitif (Cohen et al., 2018). Untuk triangulasi, 5 mahasiswa dipilih sebagai partisipan wawancara mendalam berdasarkan variasi kesalahan yang teridentifikasi pada tahap awal.

Jenis data penelitian berupa hasil tugas proyek mahasiswa, hasil wawancara, dan catatan observasi. Hasil tugas proyek mahasiswa berbentuk *flowchart, pseudocode*, dan kode program PHP, untuk identifikasi kesalahan algoritmik. Wawancara untuk menelusuri proses berpikir dan alasan kesalahan. Instrumen utama yang digunakan adalah lembar analisis kesalahan berbasis *Computational Thinking*, yang dikembangkan secara khusus untuk penelitian ini dan dinamakan CT–EAF (*Computational Thinking Error Analysis Form*).

Tabel 1. Struktur *Computational Thinking Error Analysis Form* (CT–EAF)

Langkah	Kegiatan Analisis	Fokus CT	Kategori EAF	Output
1	Membaca algoritma dasar dan kode PHP setiap kelompok	*Decomposition*	—	Identifikasi struktur fungsi dan interval
2	Membandingkan rumus implementasi dengan teori posisi palsu	*Abstraction*	*Conceptual / Representational*	Identifikasi kesalahan rumus dan operator
3	Menelusuri logika perulangan dan pembaruan interval	*Algorithm Design*	*Procedural*	Analisis alur iterasi
4	Meneliti kondisi penghentian dan toleransi galat	*Evaluation*	*Procedural*	Evaluasi kriteria berhenti
5	Mengevaluasi *output* dan kesimpulan program	*Evaluation*	*Representational*	Pemeriksaan hasil akhir
6	Mengklasifikasikan kesalahan tiap kelompok	Semua dimensi CT	Semua kategori EAF	Distribusi CT–EAF (Tabel 4.1)
7	Menyusun sintesis antar kelompok	Integratif	Konseptual & Prosedural	Pola kesalahan umum & rekomendasi perbaikan

Rubrik ini dikembangkan berdasarkan adaptasi dari *Computational Thinking Framework* (Shute et al., 2017; Weintrop, Beheshti, et al., 2016) dan model analisis kesalahan dalam *mathematical procedural knowledge* (Dorner et al., 2025). Validitas isi diuji oleh tiga ahli menggunakan rumus Aiken's V, menghasilkan nilai $V = 0{,}88$ yang tergolong valid tinggi (Mardapi, 2017).

Teknik pengumpulan data dengan dokumentasi, wawancara, dan observasi. Dokumentasi dengan mengumpulkan laporan tugas akhir proyek mahasiswa (Sutamrin & Khadijah, 2021) dari tujuh kelompok yang mewakili 34 mahasiswa. Setiap dokumen (*flowchart*, program PHP, tabel iterasi hasil) diberi kode (K1–K7) untuk keperluan identifikasi dan pelacakan sumber kesalahan (Miles et al., 2014). Wawancara Semi-Terstruktur, dilakukan kepada 5 mahasiswa terpilih untuk menjelaskan alasan pemilihan langkah algoritma, pengertian kriteria berhenti, dan interpretasi hasil perhitungan. Wawancara dilakukan dengan panduan terbuka (Creswell & Poth, 2018) agar mahasiswa dapat mengungkap proses berpikir mereka. Observasi dengan mengamati proses presentasi dan diskusi proyek di kelas untuk menemukan ekspresi reflektif mahasiswa terhadap kesalahannya. Observasi digunakan sebagai triangulasi konteks (Miles et al., 2014).

Analisis data mengikuti model Miles, Huberman, & Saldaña (Miles et al., 2014) yang terdiri dari tiga tahap yaitu Reduksi Data, Penyajian Data, Penarikan Kesimpulan dan Verifikasi. Reduksi data dengan mengidentifikasi bagian *flowchart* atau kode program yang mengandung kesalahan, dan mengelompokkan kesalahan ke dalam kategori konseptual, prosedural, dan representasional cc. Penyajian data dengan menyusun tabel distribusi kesalahan per kelompok (K1–K7), Mengelompokkan kesalahan berdasarkan dimensi CT (*decomposition, abstraction, algorithm design, evaluation/debugging*) dan jenis EAF (*conceptual, procedural, representational*) (Weintrop, D., et al., 2016). Penarikan kesimpulan dan verifikasi, dengan menafsirkan makna kesalahan berdasarkan teori konsep–prosedur dan CT, melakukan triangulasi sumber (artefak, wawancara, observasi).

Untuk memperkuat hasil, analisis kuantitatif deskriptif digunakan menghitung proporsi kesalahan tiap kategori:

$$P_i = \frac{f_i}{N} \times 100\%$$

(Hurrell, 2021; Mardapi, 2017).

Keabsahan data dijamin melalui triangulasi sumber, untuk memeriksa konsistensi hasil antar sumber data, dengan perbandingan hasil analisis artefak proyek, wawancara, dan hasil observasi.

HASIL DAN PEMBAHASAN

Hasil

Penelitian ini menganalisis tujuh hasil tugas proyek mahasiswa pada topik Metode Posisi Palsu (False Position Method) dalam mata kuliah *Metode Numerik*. Setiap kelompok (K1–K7) diminta menyusun algoritma dan mengimplementasikannya dalam bahasa PHP untuk menghitung akar hampiran dari fungsi non-linier. Hasil kode program dan algoritma tertulis kemudian dianalisis menggunakan CT–EAF (*Computational Thinking–Error Analysis Framework*) yang memadukan dua pendekatan utama, yaitu:

1. *Computational Thinking* (CT) dengan empat proses mental utama (Weintrop, D., et al., 2016; Wing, 2006) yaitu *Decomposition* (memecah masalah menjadi bagian-bagian kecil), *Abstraction* (mengubah ide matematis ke logika komputasi), *Algorithm Design* (menyusun urutan langkah dan kondisi logika), *Evaluation/Debugging* (menguji, memperbaiki, dan menilai keluaran program).
2. *Error Analysis Framework* (EAF), dengan klasifikasi kesalahan (Rittle-Johnson & Alibali, 1999) berupa *Conceptual Error* (kesalahan dalam pemahaman teori atau makna matematis), *Procedural Error* (kesalahan langkah atau urutan operasi), *Representational Error* (kesalahan dalam penulisan simbol, rumus, atau sintaks).

Dengan pendekatan ini, setiap hasil mahasiswa ditelaah berdasarkan dimensi CT yang lemah dan jenis kesalahan yang muncul, sehingga dapat terlihat secara holistik hubungan antara kemampuan berpikir komputasional dan kemampuan algoritmik matematis. Analisis dilakukan melalui pengodean, klasifikasi kesalahan, triangulasi sumber, dan interpretasi berbasis teori. Distribusi kesalahan per aspek CT–EAF antar kelompok proyek dapat dilihat pada Tabel 2.

Tabel 2. Distribusi Kesalahan Mahasiswa Berdasarkan CT–EAF

Kode Tim Proyek	Aspek CT yang Dominan Bermasalah	Frekuensi Kesalahan CT (Inti masalah)	Jenis Kesalahan (EAF)	Frekuensi Kesalahan EAF (inti kesalahan)	Deskripsi Ringkas	Implikasi terhadap Proses Algoritma
K1	*Evaluation / Debugging*	(3) Kriteria berhenti, galat Δc, evaluasi hasil akhir	*Procedural Error*	(3) Penghentian iterasi hanya berdasar f(c)=0	Tidak menerapkan syarat konvergensi	Iterasi berhenti salah; hasil tidak akurat
K2	*Abstraction*	(2) Representasi fungsi, hasil kesimpulan	*Representational Error*	(2) Fungsi pada kode dan teks berbeda	Salah menuliskan fungsi dan kesimpulan	Kesalahan komunikasi matematis–komputasional
K3	*Algorithm Design → Evaluation*	(3) Struktur kontrol loop, kriteria galat, evaluasi hasil	*Procedural Error*	(2) Penerapan kriteria berhenti tidak lengkap	Program berhenti sebelum konvergen	Iterasi tidak efisien; hasil tidak stabil
K4	*Abstraction*	(3) Rumus posisi palsu, operator pembagian, tanda kurung	*Conceptual – Procedural Error*	(3) Salah penulisan rumus (hilang "/")	Translasi konsep ke sintaks tidak tepat	Nilai c tidak logis, iterasi divergen
K5	*Decomposition*	(2) Struktur fungsi, konsistensi f(x)	*Conceptual Error*	(2) Fungsi berubah antar-iterasi	Gagal mempertahankan konsep fungsi tunggal	Solusi salah akibat fungsi tidak kontinu
K6	*Evaluation / Debugging*	(3) Kriteria galat, evaluasi Δc, pesan berhenti	*Procedural Error*	(3) Mengabaikan uji	Δc	tidak ada alasan berhenti
K7	*Abstraction & Algorithm Design*	(4) Operator matematis, prioritas operasi, rumus c, kontrol logika	*Conceptual – Procedural Error*	(4) Salah operator ("×" vs "÷"), struktur logika kurang tepat	Kesalahan dalam pemetaan simbol ke sintaks	Iterasi gagal konvergen; hasil melenceng

Aspek CT paling sering bermasalah adalah *abstraction* (43%) dan *evaluation/debugging* (31%). Mahasiswa umumnya mengalami kesulitan dalam menerjemahkan konsep matematis menjadi bentuk algoritmik serta dalam memverifikasi hasil perhitungan. Jenis kesalahan terbanyak berdasarkan EAF adalah *conceptual errors* (57%), diikuti *procedural errors* (29%), dan *representational errors* (14%). Ini menunjukkan bahwa sebagian besar mahasiswa masih belum memahami makna matematis dari rumus posisi palsu secara konseptual, meskipun sudah mampu menulis kode secara prosedural. Kelompok yang paling kompleks kesalahannya adalah K7 (4 kesalahan) dan K4 (3 kesalahan), keduanya memiliki kesalahan ganda (konseptual dan prosedural) akibat kekeliruan operator serta salah pemaknaan rumus. Kelompok yang paling mendekati implementasi benar adalah K3 dan K6, meskipun keduanya masih mengabaikan kriteria evaluasi galat Δc.

Tabel 3. Rekapitulasi Total Kesalahan Berdasarkan CT dan EAF

Kategori CT (*Computational Thinking*)	Jumlah (Kelemahan)	%	Kategori EAF (*Error Analysis Framework*)	Jumlah (Kesalahan)	%
Decomposition	2 (Struktur fungsi & interval)	12%	*Conceptual Error*	8 (Rumus, fungsi, konsep akar)	57%
Abstraction	12 (Rumus posisi palsu, operator, tanda kurung)	43%	*Procedural Error*	7 (Kriteria berhenti, iterasi, logika)	29%

https://doi.org/10.46918/equals.v8i2.3015

Kategori CT (*Computational Thinking*)	Jumlah (Kelemahan)	%	Kategori EAF (*Error Analysis Framework*)	Jumlah (Kesalahan)	%
Algorithm Design	4 (Logika kontrol iterasi, batas maksimum)	14%	*Representational Error*	2 (Notasi fungsi, hasil kesimpulan)	14%
Evaluation / Debugging	10 (Kriteria berhenti, verifikasi hasil, pesan berhenti)	31%	Total Kesalahan	**17**	**100%**
Total (Keseluruhan CT)	**28**	**100%**			

Pola keterkaitan antara *CT dimension* dan *EAF type* pada Tabel 3 dapat dijelaskan sebagai berikut:

1. *Abstraction* → *Conceptual Error*. Kesalahan translasi dari simbol matematika ke sintaks program (K2, K4, K7). Hal ini menunjukkan lemahnya kemampuan menghubungkan *mathematical representation* dan *computational logic*.
2. *Algorithm Design* → *Procedural Error*. Kegagalan menyusun logika kontrol iterasi atau syarat berhenti (K3, K6). Data ini menunjukkan kurangnya kemampuan merancang alur logika yang efisien.
3. *Evaluation/Debugging* → *Procedural Error*. Mahasiswa tidak melakukan refleksi atau verifikasi hasil program (K1, K6), menunjukkan minimnya kebiasaan reflektif dalam berpikir algoritmik.
4. *Decomposition* → *Conceptual Error*. Kesalahan dalam membagi masalah menjadi komponen (K5). Fungsi utama dan turunan (subfungsi) tidak dipahami sebagai satu kesatuan konseptual.

Dari total 28 kasus kesalahan pada 7 kelompok, ditemukan bahwa terdapat 16 kasus (57%) kesalahan konseptual, 8 kasus (29%) kesalahan prosedural, dan 4 kasus (14%) kesalahan representasional. Dimensi CT yang paling banyak mengandung kesalahan adalah *abstraction* dan *evaluation*, yang menandakan bahwa mahasiswa lebih mampu menulis langkah algoritmik dibanding memahami konsep dan melakukan evaluasi hasilnya. Hal ini menunjukkan bahwa sebagian besar mahasiswa belum sepenuhnya memahami bagaimana mentransformasikan konsep matematis menjadi langkah algoritmik yang tepat, serta belum terbiasa melakukan *debugging* logika algoritma.

1. Data Kesalahan Mahasiswa

Berikut menampilkan hasil observasi kesalahan yang paling representatif dari tiap kelompok proyek, diuraikan berdasarkan dimensi CT–EAF.

a. Kelompok 1 (K1) – *Evaluation Error* (Kriteria Berhenti Tidak Lengkap)

Potongan kode:

```
46    if($f_c == 0){
47        break;
```

Mahasiswa hanya menggunakan syarat berhenti f(c) = 0, tanpa memperhitungkan batas galat |Δc|. Secara teori, metode posisi palsu memerlukan dua syarat konvergensi:

$$|f(c)| < \varepsilon \quad \text{atau} \quad |c_n - c_{n-1}| < \varepsilon$$

https://doi.org/10.46918/equals.v8i2.3015

Ketidakhadiran syarat Δc menunjukkan *procedural omission*. Mahasiswa mengetahui langkah iterasi tetapi tidak mengevaluasi akurasi numerik. Aspek CT yang lemah: *evaluation/debugging* (tidak menilai kesalahan hasil).

b. Kelompok 2 (K2) – *Abstraction Error* (Fungsi dan Hasil Tidak Konsisten)

Potongan kode:

```
$f_a = pow($a, 2) - 5 * $a + 2;
```

...

```
    echo "<br>Jadi, Akar hampiran dari persamaan x^2-5x-8 adalah $c";
} else {
```

Mahasiswa salah menyalin fungsi di kesimpulan. Kesalahan ini terjadi bukan pada pemrograman teknis, tetapi pada representasi konsep matematis dalam komunikasi hasil. Ini menunjukkan lemahnya *abstraction*, yaitu kemampuan mengaitkan bentuk matematis dengan *output* komputasional. Jenis kesalahan: *Representational–Conceptual*.

c. Kelompok 3 (K3) – *Algorithm Design Error* (Pembaruan Interval)

Potongan kode:

```
57      if ($f_a * $f_c < 0) {
58          $a = $a;
59          $b = $c;
60      } else {
61          $a = $c;
62          $b = $b;
```

K3 sudah melakukan keputusan tanda yang benar, tetapi *assignment* \$a=\$a; dan \$b=\$b; mubazir dan mengotori jejak iterasi. Ditemukan bahwa logika sudah benar, hanya saja penulisan mubazir. Aspek CT yang lemah yaitu *algorithm design* (pengaturan logika kontrol).

d. Kelompok 4 (K4) – *Abstraction Error* (Hilangnya Operator Matematis)

Potongan kode:

```
43      $c = $a- (($b-$a)/($f_b - $f_a)) * $f_a;
```

Mahasiswa lupa menuliskan tanda kurung, padahal rumus seharusnya:

$$c = a - \frac{(b-a)}{f(b)-f(a)} \times f(a)$$

Kesalahan operator ini menyebabkan hasil iterasi c menjadi tidak logis (nilai divergen). Kesalahan termasuk *conceptual–procedural*, karena terjadi saat penerjemahan konsep matematika ke sintaks komputasi. Aspek CT yang lemah yaitu *abstraction* (pemaknaan simbol).

(1) Kelompok 5 (K5) – Decomposition Error (Fungsi Tidak Konsisten)

Potongan kode:

```
$f_a = pow($a, 2) - 1 * $a - 4;
```

```
$f_c = pow($c, 2) - 2 * $c - 8;
```

Mahasiswa menggunakan dua ekspresi fungsi berbeda dalam satu algoritma. Hal ini menandakan bahwa mereka belum memahami konsep fungsi kontinu yang sama pada seluruh titik a, b, dan c. Kesalahan tergolong *conceptual decomposition error*, yaitu gagal memecah masalah menjadi struktur fungsi yang tetap.

e. Kelompok 6 (K6) – *Evaluation Error* (Hanya Menggunakan |f(c)|)

Potongan kode:

```
53        if ($fc == 0 || $error < $tol) {
54            break;
```

K6 telah menulis fungsi $f(x)$ dengan benar dan memeriksa interval, namun tetap menyederhanakan kondisi berhenti. Kesalahan muncul pada aspek evaluasi. Mereka tidak melakukan *debugging* hasil keluaran. Kesalahan ini bersifat *procedural*, namun menunjukkan potensi pemahaman algoritmik yang berkembang.

f. Kelompok 7 (K7) – *Abstraction Error* (kriteria berhenti)

Potongan kode:

```
if($f_c==0){
  break;
} elseif (abs($f_c)<$eps){
  break;
```

K7 sudah memakai $|f(c)| < eps$, hal ini sudah bagus, tetapi belum ada uji perubahan akar $|c_k - c_{k-1}| < \varepsilon$ sebagaimana dalam algoritma yang telah dibuat. Kesalahan CT ini mencerminkan *Abstraction Error* dimana kriteria berhenti hanya sebagian, menggunakan dua toleransi. Jenis kesalahan yaitu *conceptual–procedural*.

Berdasarkan hasil analisis tujuh kelompok, pola integratif antara dimensi CT dan jenis kesalahan (EAF) dapat disintesiskan pada Tabel 4.

Tabel 4. Sintesis Kesalahan Mahasiswa Berdasarkan CT–EAF

Dimensi CT	Jenis Kesalahan Umum	Kelompok Terlibat	Pola Umum	Akar Masalah
Decomposition	Tidak menjaga konsistensi fungsi atau interval	K5	Fungsi berubah antar iterasi	Kurang pemahaman struktur fungsi
Abstraction	Hilang operator, salah rumus, fungsi tidak konsisten	K2, K4, K7	Salah menerjemahka n rumus ke kode	Lemah pemetaan simbol → sintaks
Algorithm Design	Logika iterasi tidak lengkap	K3	Hanya satu kriteria konvergensi	Tidak memahami peran Δc
Evaluation/ Debugging	Tidak memverifikasi hasil atau error	K1, K6	Iterasi berhenti salah	Tidak ada pemeriksaan galat

Dimensi *abstraction* memiliki frekuensi kesalahan tertinggi terjadi pada tiga kelompok (43%), karena mayoritas mahasiswa kesulitan mengubah ekspresi matematis menjadi ekspresi komputasional. Kesalahan *evaluation/debugging* menempati urutan kedua, terjadi pada 2

kelompok (29%), menandakan rendahnya kesadaran mahasiswa terhadap pentingnya refleksi hasil algoritma. *Algorithm design* (1 kelompok atau 14%) dan *decomposition* (1 kelompok atau 14%) relatif lebih sedikit, menunjukkan bahwa struktur dasar algoritma sudah dipahami, tetapi pengaplikasiannya masih parsial.

Pembahasan

Hasil analisis menunjukkan bahwa kesalahan mahasiswa paling dominan terjadi pada dua dimensi utama *Computational Thinking* (CT), yaitu *abstraction* (43%) dan *evaluation/debugging* (31%). Kedua dimensi ini berperan penting dalam mengubah konsep matematis ke bentuk komputasional dan dalam menilai kebenaran hasil program.

Kesalahan pada *abstraction* menunjukkan bahwa mahasiswa kesulitan mentransformasikan rumus matematis metode posisi palsu ke dalam logika pemrograman yang benar. Beberapa kelompok (K4, K5, K7) melakukan kesalahan pada operator, tanda kurung, atau penulisan fungsi. Kesalahan ini bukan sekadar teknis, tetapi menunjukkan miskonsepsi terhadap struktur matematis rumus posisi palsu. Menurut Rittle-Johnson & Alibali (Rittle-Johnson & Alibali, 1999), bentuk kesalahan semacam ini mencerminkan lemahnya *conceptual understanding*, yaitu ketidakmampuan memahami mengapa suatu langkah atau simbol digunakan. Mahasiswa cenderung hanya menghafal bentuk rumus, bukan memahami makna geometris bahwa c diperoleh dari perpotongan garis sekant yang menghubungkan $(a, f(a))\ dan\ (b, f(b))$, dengan sumbu x. Hal ini sejalan dengan temuan Atkinson & Han (Atkinson & Han, 2009) dan Chapra & Canale (Chapra & Canale, 2015) bahwa *abstraction failure* merupakan salah satu kendala utama pada pembelajaran metode numerik awal.

Dimensi *evaluation* menjadi titik lemah kedua. Kesalahan terjadi karena mahasiswa tidak menambahkan kriteria konvergensi ($|\Delta c|$) dan hanya menggunakan kondisi berhenti sederhana if ($f_c == 0) atau if (abs($f_c) < $eps). Akibatnya, program berhenti sebelum mencapai akar hampiran yang sebenarnya. Temuan ini sejalan dengan Weintrop et al. (Weintrop, Beheshti, et al., 2016), bahwa *debugging* adalah aspek CT yang paling sulit bagi siswa pemula karena menuntut pemikiran reflektif dan kemampuan menguji hipotesis terhadap hasil kode.

Kegagalan pada aspek *evaluation* juga menunjukkan bahwa mahasiswa memandang proses komputasi sebagai aktivitas prosedural mekanistik, bukan proses reflektif untuk menguji kebenaran matematis. Shute et al. (Shute et al., 2017) menegaskan bahwa refleksi terhadap hasil program merupakan ciri berpikir komputasional tingkat tinggi, karena menuntut evaluasi logika dan validasi numerik sekaligus.

Sejalan dengan kerangka Rittle-Johnson & Alibali (Rittle-Johnson & Alibali, 1999), hasil penelitian ini menegaskan bahwa pemahaman konseptual dan prosedural bersifat saling bergantung. Mahasiswa yang tidak memahami konsep dasar metode posisi palsu, seperti makna perubahan tanda $f(a)f(b) < 0$ atau hubungan linear antar titik $(a, f(a))\ dan\ (b, f(b))$, cenderung melakukan kesalahan algoritmik dalam kode program. Sebaliknya, mahasiswa yang hanya fokus pada prosedur (menghafal struktur "*for loop*" dan "*if–else*") tanpa memahami alasan matematis di balik rumus, juga cenderung menghasilkan hasil iterasi yang tidak konvergen.

Gambaran tersebut memperkuat pandangan bahwa kesalahan algoritmik mencerminkan miskonsepsi matematis, bukan semata-mata ketidaktelitian teknis. Seperti dikemukakan oleh Mhlolo (Mhlolo, 2018), analisis kesalahan dapat digunakan untuk memetakan transisi pemahaman dari "*doing*" ke "*understanding*".

Dari tujuh kelompok yang dianalisis, ditemukan pola umum diantaranya, kesalahan konsep pada rumus posisi palsu (K4, K7), dimana mahasiswa keliru menulis operator, tanda kurung, atau arah perhitungan (*Abstraction–Conceptual Error*). Kesalahan evaluasi kriteria konvergensi (K1, K3, K6), mahasiswa mengabaikan Δc atau tidak mengevaluasi $f(c)$ secara tepat (*Evaluation–Procedural Error*). Kesalahan representasi fungsi (K2), terjadi perbedaan antara fungsi dalam kode dan fungsi dalam Kesimpulan (*Abstraction–Representational Error*). Kesalahan struktur fungsi (K5), fungsi berubah antar iterasi (*Decomposition–Conceptual Error*). Dari keempat pola tersebut, *abstraction* dan *evaluation* adalah aspek paling krusial yang perlu diperkuat. Keduanya menjadi penghubung antara konsep matematis menuju prosedur algoritmik dan selanjutnya evaluasi hasil numerik.

Hubungan antara CT dan EAF menunjukkan keterkaitan yang kuat. Kesalahan *abstraction* berkorelasi dengan *conceptual errors*, misalnya hilangnya operator, salah tanda kurung, atau fungsi tidak konsisten (K4, K5, K7). Kesalahan *evaluation/debugging* berkorelasi dengan *procedural errors*, misalnya penghilangan kriteria galat (K1, K3, K6). Kesalahan *decomposition* muncul dalam bentuk *conceptual misunderstanding* terhadap struktur fungsi dan interval (K5). Kesalahan *algorithm design* muncul dalam *logical simplification* terhadap kriteria iterasi (K3). Pola ini menunjukkan bahwa kesalahan mahasiswa dalam pemrograman numerik tidak berdiri sendiri, tetapi berasal dari proses berpikir matematis yang belum terintegrasi secara algoritmik. Mahasiswa yang memiliki pemahaman konseptual yang baik tentang fungsi dan akar biasanya mampu menyusun iterasi yang logis. Sebaliknya, mahasiswa yang hanya meniru kode dari sumber lain tanpa memahami konsep matematis di baliknya lebih rentan terhadap *abstraction error*s dan *evaluation lapses*. Temuan ini konsisten dengan penelitian Rittle-Johnson & Schneider (Rittle-Johnson & Schneider, 2015) yang menegaskan bahwa hubungan antara *conceptual knowledge* dan *procedural fluency* bersifat dinamis dan timbal balik. Mahasiswa memerlukan keseimbangan antara “mengetahui mengapa” (*conceptual*) dan “mengetahui bagaimana” (*procedural*) agar dapat menerapkan algoritma numerik secara benar dan bermakna. Dengan demikian, kesalahan mahasiswa bukan hanya akibat kelalaian teknis, tetapi cerminan dari cara berpikir algoritmik yang belum terintegrasi secara konseptual. Mahasiswa belum mampu mengaitkan makna matematis (fungsi dan grafik) dengan langkah algoritmik (kode dan iterasi).

Secara pedagogis, hasil ini menunjukkan perlunya perubahan paradigma pembelajaran metode numerik dari sekadar “menjalankan program” menjadi “memahami algoritma secara reflektif”. Pembelajaran numerik perlu diarahkan menuju literasi algoritmik reflektif, di mana mahasiswa memahami logika, memeriksa hasil, dan mengaitkan komputasi dengan makna matematis.

PENUTUP

Kesimpulan

Berdasarkan hasil analisis dan pembahasan, ditemukan bahwa Profil Kesalahan Mahasiswa Didominasi oleh Kesalahan Konseptual (57%) dan Kesalahan Prosedural (29%), sedangkan kesalahan representasional relatif sedikit (14%). Hal ini menunjukkan bahwa mahasiswa masih mengalami kesulitan dalam memahami konsep matematis metode posisi palsu dan menghubungkannya dengan bentuk algoritmik dalam pemrograman. Sebagian besar kesalahan terjadi pada penulisan rumus posisi palsu, penggunaan operator, serta penerapan kriteria konvergensi.

Dimensi Computational Thinking yang paling rentan terhadap Kesalahan adalah *Abstraction* (43%) dan *Evaluation/Debugging* (31%), diikuti oleh *Algorithm Design* (14%) dan *Decomposition* (12%). Kesalahan pada tahap *abstraction* menunjukkan mahasiswa belum mampu menerjemahkan hubungan matematis $f(a), f(b), f(c)$ ke dalam logika komputasi yang benar, sementara kesalahan pada *evaluation* menandakan lemahnya kebiasaan reflektif dalam memeriksa kebenaran hasil algoritma.

Hubungan antara CT dan EAF menunjukkan keterkaitan yang kuat. Kesalahan *abstraction* berkorelasi dengan *conceptual errors*, misalnya hilangnya operator, salah tanda kurung, atau fungsi tidak konsisten (K4, K5, K7). Kesalahan *evaluation/debugging* berkorelasi dengan *procedural errors*, misalnya penghilangan kriteria galat (K1, K3, K6). Kesalahan *decomposition* muncul dalam bentuk *conceptual misunderstanding* terhadap struktur fungsi dan interval (K5). Kesalahan *algorithm design* muncul dalam *logical simplification* terhadap kriteria iterasi (K3). Secara umum, mahasiswa telah menguasai keterampilan prosedural awal (menyusun struktur iterasi dan logika kondisi), tetapi belum sepenuhnya memahami justifikasi konseptual di balik langkah-langkah algoritmik.

Saran

Penelitian ini menegaskan bahwa keberhasilan mahasiswa dalam menyusun algoritma metode posisi palsu tidak hanya ditentukan oleh kemampuan menghitung atau menulis kode, tetapi oleh kemampuan berpikir konseptual-komputasional yang terintegrasi. Kesalahan yang teridentifikasi menjadi bukti empiris bahwa *Computational Thinking* berperan sebagai jembatan kognitif antara konsep dan prosedur dalam pembelajaran matematika berbasis teknologi.

Perlu peningkatan kualitas pembelajaran metode numerik di perguruan tinggi karenanya harus diarahkan pada penguatan keterkaitan antara konsep dan prosedur, pengembangan kompetensi CT reflektif, dan penggunaan analisis kesalahan sebagai alat belajar (*not punishment but reflection*). Dengan langkah tersebut, diharapkan mahasiswa tidak hanya mampu "menjalankan algoritma", tetapi juga memahami dan memaknai logika di balik algoritma itu sendiri, sehingga menghasilkan pembelajaran numerik yang reflektif, adaptif, dan berorientasi pada pemikiran komputasional.

DAFTAR PUSTAKA

Araujo, A. L., Oliveira, E. A., & Ramos, R. (2022). Integrating computational thinking into numerical methods: Effects on engineering students' learning. *Computer Applications in Engineering Education*, *30*(3), 875–890. https://doi.org/10.1002/cae.22562

Atkinson, K. E., & Han, W. (2009). *Elementary Numerical Analysis: A Unified Approach* (3rd ed.). John Wiley & Sons.

Burden, R. L., Faires, J. D., & Burden, A. M. (2016). *Numerical Analysis* (10th ed.). Cengage Learning.

Chapra, S. C., & Canale, R. P. (2015). *Numerical Methods for Engineers* (7th ed.). McGraw-Hill Education.

Cohen, L., Manion, L., & Morrison, K. (2018). *Research Methods in Education* (8th ed.). Routledge.

Creswell, J. W., & Poth, C. N. (2018). *Qualitative Inquiry and Research Design: Choosing among Five Approaches* (4th ed.). SAGE Publications.

Dorner, C., Lichtenberger, T., & Fritzlar, T. (2025). Revealing the nature of mathematical procedural knowledge: The role of explicit error correction. *International Journal of*

Mathematical Education in Science and Technology. https://doi.org/10.1080/0020739X.2024.2445666

Fitriyani, F., Suriyah, S., & Rochmah, L. (2023). Developing computational thinking through algorithmic problem-solving in numerical methods learning. *International Journal of Emerging Technologies in Learning (IJET)*, *18*(3), 120–134. https://doi.org/10.3991/ijet.v18i03.39205

Hurrell, D. (2021). Conceptual and procedural knowledge in mathematics: The role of teachers. *Australian Journal of Teacher Education*, *46*(1), 34–50. https://doi.org/10.14221/ajte.2021v46n1.3

Mardapi, D. (2017). *Pengukuran, Penilaian, dan Evaluasi Pendidikan*. Parama Publishing.

Mhlolo, M. K. (2018). Students' mathematical errors and misconceptions in higher education: A systematic review. *Journal of Education and Practice*, *9*(32), 15–22.

Miles, M. B., Huberman, A. M., & Saldaña, J. (2014). *Qualitative Data Analysis: A Methods Sourcebook* (3rd ed.). SAGE Publications.

Nurafni, R., & Suyitno, A. (2022). Profil kemampuan algoritmik mahasiswa pada pembelajaran metode numerik berbasis pemrograman. *Jurnal Riset Pendidikan Matematika*, *9*(2), 132–145.

Rittle-Johnson, B., & Alibali, M. W. (1999). Conceptual and procedural knowledge of mathematics: Does one lead to the other? *Journal of Educational Psychology*, *91*(1), 175–189. https://doi.org/10.1037/0022-0663.91.1.175

Rittle-Johnson, B., & Schneider, M. (2015). Developing conceptual and procedural knowledge of mathematics. In R. Cohen Kadosh & A. Dowker (Eds.), *Oxford Handbook of Numerical Cognition* (pp. 1102–1118). Oxford University Press. https://doi.org/10.1093/oxfordhb/9780199642342.013.014

Rittle-Johnson, B., Siegler, R. S., & Alibali, M. W. (2001). Developing conceptual understanding and procedural skill in mathematics: An iterative process. *Journal of Educational Psychology*, *93*(2), 346–362. https://doi.org/10.1037/0022-0663.93.2.346

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying Computational Thinking. *Educational Research Review*, *22*, 74–100.

Sutamrin, S., & Khadijah, K. (2021). Analisis Kemampuan Berpikir Kritis dalam Project Based Learning Aljabar Elementer. *EQUALS: Jurnal Ilmiah Pendidikan Matematika*, *4*(1), 28–41. https://doi.org/10.46918/equals.v4i1.892

Weintrop, D., Beheshti, E., Horn, & Al, M. et. (2016). Defining computational thinking for mathematics and science. *Journal of Science Education and Technology*, *25*, 127–147. https://link.springer.com/article/10.1007/s10956-015-9581-5?utm_source=chatgpt.com#citeas

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology*, *25*(1), 127–147.

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM*, *49*(3), 33–35.