BINUS UNIVERSITY ONLINE

People Innovation Excellence

Algorithm and Programming
COMP6112036

Nama : Annisa Baizan
Nim : 2802676625

Tugas Personal ke-2 Week 4

Soal 1:

Analisis Kesalahan dan Perbaikannya:

1	Kurang tepat	`*(i + ptr - 1) dan *(i + ptr - 2)`

2	Perbaikan	`*(i + ptr - 1) → *(ptr + i - 1)`
		`dan`
		`*(i + ptr - 2) → *(ptr + i - 2)`

Penjelasan:

Penulisan tidak idiomatik atau tidak mengikuti *best practice*

Ekspresi 1 sah secara sintaksis maupun semantik, tetapi tidak mengikuti konvensi yang umum/biasa/lazim digunakan oleh mayoritas programmer. Gaya ini membingungkan karena tidak mencerminkan praktik penulisan yang baik (*best practice)* dan akan mengurangi tingkat *readability* (keterbacaan) kode. Hal ini berpotensi menimbulkan kesalahan interpretasi, baik saat kode direvisi/ditinjau kembali (*maintainability*) maupun ketika dibaca oleh programmer lain.

Hal ini dapat dijelaskan lebih lanjut melalui poin-poin berikut:

1. **Kesalahan pada operasi pointer dalam loop perhitungan deret Fibonacci.**
   Ekspresi 1 menyiratkan bahwa indeks (i) digunakan seolah-olah sebagai pointer, karena langsung ditambahkan ke ptr dengan urutan operasi yang tidak eksplisit. Struktur seperti ini menyamarkan maksud sebenarnya dari operasi pointer tersebut.
2. **Indeks tidak boleh digunakan secara langsung sebagai pointer.**
   Meskipun **Ekspresi 1 dan 2** setara secara matematis karena operator + dan - memiliki *precedence* (tingkat prioritas) yang sama dan dievaluasi dari kiri ke kanan. Dalam konteks pointer, bentuk `ptr + i` lebih eksplisit untuk menunjukkan bahwa **i** adalah **offset** terhadap pointer.

BINUS UNIVERSITY ONLINE

People
Innovation
Excellence

Algorithm and Programming
COMP6112036

Nama : Annisa Baizan
Nim : 2802676625

Nomor 1 Code Perbaikan :

```c
#include <stdio.h>

int main()
{
    int fib[10];
    int *ptr = fib;
    *ptr = 0;
    *(ptr + 1) = 1;

    for (int i = 2; i < 10; i++)
    {
        *(ptr + i) = *(ptr + i - 1) + *(ptr + i - 2); // Perbaikan di sini
    }

    for (int i = 0; i < 10; i++)
    {
        printf("%d ", *(ptr + i));
    }

    return 0;
}
```

Hasil :

```
WSL at Annisa-Baizan-471 in ProgramC on Git main X? via C v13.3.0-gcc
❯ runc '/mnt/d/Kuliah/Algorithm_and_Programming/ProgramC/Tugas/pointerAssignmentRight.c'
[GCC] Compiling: gcc "/mnt/d/Kuliah/Algorithm_and_Programming/ProgramC/Tugas/pointerAssignmentRight.c"
-o "/mnt/d/Kuliah/Algorithm_and_Programming/ProgramC/Tugas/pointerAssignmentRight" -lm
[OK] Build successful in 261ms
[OUTPUT]
0 1 1 2 3 5 8 13 21 34
[EXIT] Exit code: 0
```

Link GitHub: pointerAssignmentRight.c

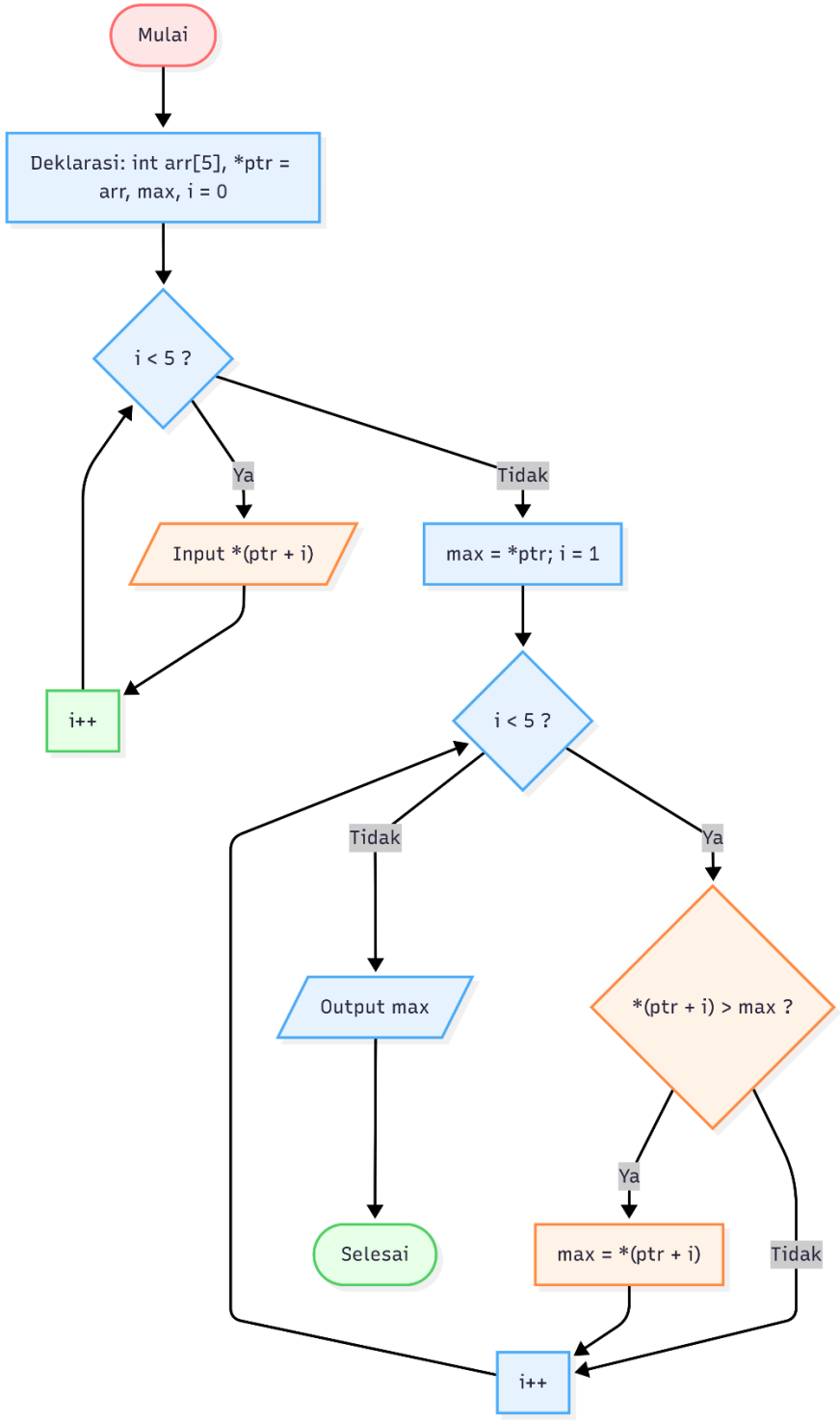
BINUS UNIVERSITY ONLINE

People
Innovation
Excellence

Algorithm and Programming
COMP6112036

Nama : Annisa Baizan
Nim : 2802676625

Soal 2:



BINUS UNIVERSITY ONLINE

People
Innovation
Excellence

Algorithm and Programming
COMP6112036

Nama : Annisa Baizan
Nim : 2802676625

Referensi

Martin, R. C. (2008). Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall.

Oualline, S. (2003). Practical C Programming (3rd ed.). O'Reilly.

Strunk, W., & White, E. B. (2000). The Elements of Style (4th ed.). Pearson.

Kernighan, B. W., & Pike, R. (1999). The Practice of Programming. Addison-Wesley.

Kernighan, B. W., & Ritchie, D. M. (1988). The C Programming Language (2nd ed.). Prentice Hall.

ISO/IEC. (2018).Information technology — Programming languages — C. International Organization for Standardization.

Google. (t.t.). Google C++ Style Guide. https://google.github.io/styleguide/cppguide.html

Stroustrup, B. & Sutter, H. (Eds.). (t.t.). C++ Core Guidelines. https://isocpp.github.io/CppCoreGuidelines/CppCoreGuidelines

Dafer, S. M. (2023, November 20). Code Clarity vs Performance: Frustrating mistakes and examples in programming. Medium, https://stevedafer.medium.com/code-clarity-vs-performance-frustrating-mistakes-and-examples-in-programming-3e644064921f

Stack Exchange. (2022). Why is ptr + i preferred over i + ptr in C when accessing array elements? Software Engineering Stack Exchange. https://softwareengineering.stackexchange.com/questions/442986/why-is-ptr-i-preferred-over-i-ptr-in-c-when-accessing-array-elements

Buse, R. P. L., & Weimer, W. R. (2010). Learning a Metric for Code Readability. IEEE Transactions on Software Engineering, 36(4), 546–558. https://doi.org/10.1109/TSE.2009.70

***Source*:** ChatGPT-4o, DeepSeek-V3, Gemini 2.5 Pro, Claude Sonnet 4. Diakses pada 29 Juni 2025.