LAPORAN TUGAS PROGRAMMING LANGUAGE PRAGMATICS

OLEH:

Arkan Ramadhan Nugraha	241524033
Julian Dio Saputra	241524049
Salman Alfarisi	241524060

TEKNIK INFORMATIKA JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA POLITEKNIK NEGERI BANDUNG

2025

1. Kita memilih paradigma impertative
2. Paradigma imperatif
    a. Karakteristik dan teknik khusus
    Pemrograman imperatif adalah paradigma pemrograman yang menekankan urutan langkah-langkah dan mengubah state dalam program Pemrograman imperatif menentukan urutan perintah di mana perintah dieksekusi secara berurutan dan mengubah state dalam program hingga hasil akhir tercapai.

    Ciri khas

    - Fokus pada algoritma dan urutan eksekusi
    - Didasarkan pada struktur kontrol, assignment, dan I/O (input/output).
    - Menggunakan loop kondisional (for, while).
    - Menggunakan statement yang dapat diubah dan dimanipulasi pada setiap eksekusi
    - Fokus pada tugas-tugas kecil untuk menyelesaikan program, bukan hanya hasil akhir.

    Teknik Khusus

    - Menggunakan kontrol alur (if, for, while) untuk mengatur eksekusi.
    - Pecah program menjadi fungsi/prosedur yang melakukan tugas tertentu.
    - Membagi kode menjadi modul-modul independen untuk mempermudah maintenance dan reuse.

    b. Bahasa Pemrograman yang mendukung paradigma ini dan karakteristiknya
        i. C
            1. Tipe variabel harus dideklarasikan secara eksplisit.
            2. Strong typing.
            3. Efisien dan dekat dengan hardware
            4. Mendukung modular programming dan prosedural programming
            5. Memiliki kontrol alur lengkap (if, for, while, switch)
        ii. Pascal
            1. Strong typing.
            2. Mendukung modular programming dan prosedural programming
            3. Sintaks jelas dan mudah dipelajari.
            4. Memiliki kontrol alur yang lengkap
        iii. Python
            1. Dukungan struktur data built-in (list, dict, set)
            2. Sintaks sederhana dan mudah dibaca
            3. Mendukung gaya imperatif melalui statement, loop, dan fungsi.
            4. Modular programming melalui module dan function.
            5. Python mendukung multi-paradigma, tapi juga bisa digunakan secara imperatif

c. Kasus yang memerlukan paradigma tersebut dan contohnya
Paradigma ini cocok jika:
- Program harus mengontrol state secara eksplisit
- Algoritma bersifat step-by-step
- Perlu efisiensi tinggi dan manipulasi memori langsung

Contoh kasus:

- Sistem operasi (Linux kernel ditulis mayoritas C)
- Embedded system (mikrokontroler)

d. Kelebihan dan kekurangan
Kelebihan
- Bergantung pada intruksi yang ditentukan untuk mencapai hasil akhir program. Oleh karena itu, kodenya mudah dipahami dan straightforward (lugas).
- Urutan pelaksanaan operasi sepenuhnya dikontrol oleh pengembang.
- Bug dapat mudah dilacak karena program disusun dari blok-blok kode untuk melakukan tugas yang kecil dan didasarkan pada sifat langkah-demi-langkah
- Alokasi memori dan manipulasi langsung terkait. Karena itu, pemanfaatan memori sangat efisien

Kekurangan

- Dalam program besar, control flow dan perubahan variabel meningkatkan kompleksitas kode.
- Dapat adanya rangkaian perintah-perintah yang panjang, menyebabkan kesusahan dalam keterbacaan (readability) maupun pemeliharaan kode dalam program besar.

3. Lakukan evaluasi terhadap salah satu Bahasa pemrograman dari paradigma tersebut berdasarkan kriteria *readability, writability, reliability, dan cost.*
   a. Readability
      - Overall Simplicity
        Bahasa C sederhana, hanya punya 32 keyword inti. Namun, kompleksitas muncul dari fitur tingkat rendah seperti pointer dan manajemen memori manual.
      - Feature Multiplicity
        Terdapat beberapa cara melakukan operasi yang sama, misalnya i++, ++i, dan i = i + 1, Hal ini kadang membingungkan bagi pemula.
      - Minimal Operator Overloading

C tidak mendukung operator overloading (berbeda dengan C++). Hal ini membuat kode lebih mudah dibaca karena operator selalu memiliki arti yang sama.

- Orthogonality

  Orthogonality C rendah. Contoh: array tidak bisa dikembalikan dari fungsi, sementara struct bisa. Inkonsistensi ini menurunkan readability.

- Control Statements

  C memiliki kontrol alur yang lengkap (if, else, for, while, switch). Hal ini membantu readability karena struktur alur program mudah dikenali.

- Data Types and Structures

  C menyediakan tipe data dasar (int, float, char) dan struct, enum, serta pointer. Namun, string harus dikelola manual menggunakan array of char, yang menambah kerumitan.

- Syntax Considerations

  Identifier fleksibel, dapat menggunakan huruf, angka, dan underscore. Kata kunci singkat dan terbatas, sehingga mudah diingat. Namun, banyak simbol khusus (*, ->, &) membuat kode sulit dipahami bagi pemula.

b. Writability

- Simplicity and Orthogonality

  Sintaks C sederhana dan ringkas, tapi orthogonality rendah menyebabkan keterbatasan. Misalnya, pointer bisa digunakan di banyak konteks tapi butuh aturan tambahan.

- Support for Abstraction

  C mendukung fungsi (function) dan struct untuk abstraksi. Namun, tidak ada OOP (class, inheritance), sehingga tingkat abstraksi terbatas.

- Expressivity

  Bahasa C ekspresif untuk operasi tingkat rendah. Contoh: i++ jauh lebih ringkas daripada i = i + 1. Dukungan loop (for, while) membuat penulisan program singkat.

- Memory Management

  Programmer harus melakukan manajemen memori sendiri dengan malloc() dan free(). Hal ini memberi fleksibilitas tinggi, tetapi juga menambah beban penulisan program.

c. Reliability

1. Type Checking

   Bahasa C sangat sensitif dengan type checking walaupun terbilang lemah. C memungkinkan untuk mengonversi tipe antar data secara implisit yang dapat menyebabkan bug.

2. Exception Handling

Bahasa C tidak mempunyai exception handling bawaan. Pada Bahasa pemrograman modern terdapat sintaks untuk mengurus error yang terjadi seperti try-catch. Dalam Bahasa C, setiap kemungkinan error di definisikan satu-persatu untuk setiap function call

3. Aliasing
   Di dalam Bahasa C terdapat pointer dimana multiple pointers dapat menunjuk memory location yang sama, terdapat juga global variable yang bisa di akses semua modul.
4. Readibility and Writability
   Bahasa C tidak mempunyai garbage collection bawaan sehingga berpotensi untuk terjadi memory leak.
   Di dalam Bahasa C juga programmer harus mengatur memory yang dialokasikannya sendiri (manual) seperti malloc/free

d. Cost
   1. Training programmers to use language
      Untuk memahami bahasa C, perlu waktu yang lumayan terutama untuk memahami memory management. Bahasa C sulit untuk pemula karena sintaks yang kompleks dan low-access resource tapi direkomendasikan untuk membangun fundamental pemrograman.
   2. Writing Programs (closeness to particular applications)
      Penulisan program tergantung dari kompleksnya aplikasi yang ingin dibuat
   3. Compiling programs
      Bahasa C dapat meng-compile dengan sangat cepat karena sudah native compiled sehingga memiliki waktu compile yang cenderung rendah
   4. Executing programs
      Bahasa C memiliki direct hardware access yang dapat memaksimalkan performance. Karena hal itu runtime dengan program Bahasa C cenderung cepat

e. Other
   1. Portability
      Program yang dibuat dengan Bahasa C bisa berjalan dan di compile di system apapun bahkan tanpa atau tidak ada perubahan
   2. Generality
      TIdak terbatas pada domain tertentu, sehingga bisa digunakan untuk aplikasi apapun walaupun kadang di domain tertentu bukan yang terbaik. Bahasa C juga telah memenuhi standar ANSI C. Bahasa C sudah terdefinisi dengan jelas mulai dari sintaks nya hingga implementasi di memori fisik. Bahasa C juga bahkan sering digunakan oleh bahasa pemrograman lain. Selain itu, C telah ada selama puluhan tahun dan masih digunakan hingga sekarang, sehingga telah terbukti stabil dan teruji. Selain itu, karena C merupakan bahasa yang sangat terkenal dan banyak digunakan, sumber belajar, dokumentasi, dan contoh kode tersedia secara luas.

Referensi

https://www.geeksforgeeks.org/software-engineering/language-evaluation-criteria/

https://www.geeksforgeeks.org/c/c-programming-language-standard/

https://www.geeksforgeeks.org/c/c-language-introduction/

https://www.geeksforgeeks.org/system-design/what-is-imperative-programming/

https://en.wikipedia.org/wiki/Imperative_programming

7.1: Programming Language Foundations - Engineering LibreTexts