Nama : Revito Manuel H. M

NIM : 2305551130

Kelompok : 24

|  |
| --- |
| **MODUL I**  **“DASAR PEMROGRAMAN, FUNGSI, DAN PROSEDUR”** |

**Tujuan**

1. Mampu menerapkan dasar pemrograman bahasa C/C++ dalam pembuatan program.
2. Mampu menggunakan dan menerapkan metode fungsi dan prosedur pada sebuah program.

**Tugas Pendahuluan**

1. Sebutkan dan jelaskan secara rinci tipe data dasar yang digunakan dalam bahasa C/C++ dan keperluan memori untuk masing-masing tipe data serta berikan contoh pendeklarasian tipe data tersebut!
2. Jelaskan pengertian dan kegunaan dari flowchart! Lalu gambarkan dajelaskan fungsi masing-masing simbol yang terdapat dalam flowchart!
3. Jelaskan tentang konsep variabel lokal dan variabel global serta perbedaannya!
4. Jelaskan tentang konsep fungsi dan prosedur, perbedaan keduanya, serta aplikasinya dalam pemrograman dengan C/C++!
5. Jelaskan tentang konsep passing by value, passing by reference dan perbedaannya, serta buatlah contoh program dengan Bahasa C/C++!
6. Jelaskan yang dimaksud dengan seni penulisan program beserta contohnya!
7. Buatlah flowchart dan pseudocode (notasi algoritma) dari program pada soal di bawah ini, lalu lakukan trace terhadap flowchart yang Anda buat sehingga yakin bahwa flowchart tersebut sudah benar.

**Jawaban**

# Tipe Data Dasar dalam Bahasa C/C++

Dalam bahasa pemrograman C/C++, terdapat beberapa tipe data dasar yang digunakan untuk mendefisinikan jenis data yang akan disimpan dalam variabel, di mana setiap tipe data memiliki keperluan memori yang berbeda-beda. Berikut adalah beberapa tipe data dasar yang umum digunakan dalam bahasa C/C++.

## Int

Tipe data ini digunakan pada saat ingin menyimpan suatu bilangan bulat *(integer)* dalam bahasa C/C++, dapat berupa bilangan positif, negatif, maupun nol. Dalam sebagian besar *platform*, tipe data ini biasanya membutuhkan 4 byte (32 bit). Berikut adalah contoh pendeklarasian tipe data ‘int’.

|  |
| --- |
| int umur = 19; |

**Kode Program 1.1** Contoh Pendeklarasian Tipe Data ‘int’

Kode Program 1.1 merupakan contoh pendeklarasian tipe data ‘int’ dengan *input* berupa angka. Tipe data ini hanya dapat menyimpan bilangan bulat tanpa desimal. Jika *input* yang dimasukkan berupa angka desimal, huruf, atau jenis lain di luar bilangan bulat, maka bisa menimbulkan kesalahan dalam sebagian besar bahasa pemrograman.

## Float

Tipe data ini digunakan pada saat ingin menyimpan suatu bilangan pecahan *(floating-point)* dalam bahasa C/C++, dapat berupa bilangan riil dengan desimal. Tipe data ini biasanya membutuhkan 4 byte (32 bit). Berikut adalah contoh pendeklarasian tipe data ‘float’.

|  |
| --- |
| float nilai = 9.90; |

**Kode Program 1.2** Contoh Pendeklarasian Tipe Data ‘float’

Kode Program 1.2 merupakan contoh pendeklarasian tipe data ‘float’ dengan *input* berupa angka desimal. Tipe data ini hanya dapat menyimpan bilangan desimal. Jika *input* yang dimasukkan di luar dari bilangan desimal, maka bisa menimbulkan kesalahan dalam sebagian besar bahasa pemrograman.

## Char

Tipe data ini digunakan untuk menyimpan karakter, seperti huruf, angka, atau simbol. Tipe data ini biasanya membutuhkan 1 byte (8 bit). Berikut adalah contoh pendeklarasian tipe data ‘char’.

|  |
| --- |
| char kategori\_nilai = A; |

**Kode Program 1.3** Contoh Pendeklarasian Tipe Data ‘char’

Kode Program 1.3 merupakan contoh pendeklarasian tipe data ‘char’ dengan *input* berupa karakter. Tipe data ini hanya dapat menyimpan satu karakter saja. Jika *input* yang dimasukkan lebih dari satu karakter, maka bisa menimbulkan kesalahan dalam sebagian besar bahasa pemrograman.

## Double

Tipe data ini hampir sama kegunaannya dengan ‘float’, namun memiliki presisi yang lebih tinggi, jadi dapat digunakan pada saat ingin menyimpan suatu bilangan dengan lebih banyak angka desimal. Tipe data ini biasanya membutuhkan 8 byte (64 bit). Berikut adalah contoh pendeklarasian tipe data ‘double’.

|  |
| --- |
| double nilai = 9.8788879; |

**Kode Program 1.4** Contoh Pendeklarasian Tipe Data ‘double’

Kode Program 1.4 merupakan contoh pendeklarasian tipe data ‘double’ dengan *input* berupa angka desimal. Tipe data ini hanya dapat menyimpan bilangan desimal dan dapat menampung bilangan desimal lebih banyak dari pada tipe data ‘float’. Jika *input* yang dimasukkan di luar dari bilangan desimal, maka bisa menimbulkan kesalahan dalam sebagian besar bahasa pemrograman.

## Bool

Tipe data ini digunakan untuk menyimpan nilai kebenaran, yaitu ‘true’ dan ‘false’. Namun, tipe data ini hanya bisa digunakan dalam bahasa C++ dan biasanya membutuhkan 1 byte (8 bit). Berikut adalah contoh pendeklarasian tipe data ‘bool’.

|  |
| --- |
| bool benar = true; |

**Kode Program 1.5** Contoh Pendeklarasian Tipe Data ‘bool’

Kode Program 1.5 merupakan contoh pendeklarasian tipe data ‘bool’ yang hanya dapat menyimpan *input* berupa *true* atau *false*. Jika *input* yang dimasukkan di luar dari dua *input* tersebut, maka bisa menimbulkan kesalahan dalam sebagian besar bahasa pemrograman.

## Short dan long

Tipe data ini digunakan pada saat ingin menyimpan bilangan bulat, tetapi dengan ukuran yang beragam tergantung pada *platform* juga *compiler*. Tipe data ‘short’ biasanya berukuran lebih kecil daripada ‘int’, sedangkan ‘long’ dapat lebih besar. Berikut adalah contoh pendeklarasian kedua tipe data tersebut.

|  |
| --- |
| short nilai = 100;  long volume = 1000000L; |

**Kode Program 1.6** Contoh Pendeklarasian Tipe Data ‘short’ dan ‘long’

Kode Program 1.6 merupakan contoh pendeklarasian tipe data ‘short’ dan ‘long’ dengan *input* berupa bilangan bulat. Jika *input* yang dimasukkan di luar dari bilangan bulat, maka bisa menimbulkan kesalahan dalam sebagian besar bahasa pemrograman.

# Flowchart

Diagram alir atau yang biasa disebut *flowchart* adalah gambaran dari alur suatu proses atau algoritma yang digunakan dalam pemrograman, rekayasa perangkat lunak, dan bidang lainnya. Umumnya *flowchart* dibuat untuk merancang, mendokumentasikan, serta memberikan pemahaman secara visual tentang bagaimana suatu sistem atau algoritma bekerja. Selain itu *flowchart* juga dapat dijadikan sebagai media komunikasi yang efektif pada saat menjelaskan konsep atau prosedur kepada orang lain.

## Simbol-simbol pada flowchart

Dalam *flowchart*, terdapat simbol-simbol yang dijadikan sebagai penggambaran berbagai elemen dalam algoritma atau proses. Berikut adalah beberapa simbol dasar yang umum dipakai dalam *flowchart*.

Tabel . Simbol-Simbol Flowchart

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama** | **Gambar** | **Kagunaan** |
| 1 | *Process* | Action or Process Symbol | Menunjukkan aktivitas atau langkah-langkah dalam suatu proses. |
| 2 | *Decision* | Decision Symbol | Menunjukkan suatu kondisi atau keputusan dalam suatu algoritma. |
| 3 | Terminal | Start and End Symbol | Menunjukkan awal atau akhir suatu algoritma atau subproses. |
| 4 | *Arrows* |  | Mengindikasikan arah aliran suatu *flowchart*. |
| 5 | *Imput/output* | Input/Output Symbol | Memasukkan data *(input)* atau menampilkan hasil dari suatu algoritma *(output)*. |
| 6 | *Manual input* | Manual Input Symbol | Representasi langkah-langkah atau operasi yang dilakukan secara manual oleh manusia, yang tidak terotomatisasi dalam suatu algoritma. |
| 7 | *Connector* | Connector Symbol | Menghubungkan dua bagian dari *flowchart* pada halaman yang berbeda dalam dokumen yang lebih besar untuk menghindari kebingungan dalam pemahaman alur proses. |
| 8 | *Document* | Document Symbol | Menunjukkan dokumen yang terkait dengan proses atau algoritma yang sedang digambarkan dalam suatu *flowchart*. |
| 9 | *Multiple docuents* | Multipe Documents Symbol | Menggambarkan dokumen dengan jumlah banyak pada suatu algoritma yang dijelaskan dalam *flowchart*. |
| 10 | *Database* | Database Symbol | Memberikan pemahaman pada suatu tindakan yang melibatkan akses atau manipulasi data dalam sistem penyimpanan data. |

Tabel 2.1 merupakan daftar beberapa simbol yang umum digunakan dalam pembuatan suatu *flowchart*. Simbol-simbol ini dapat dimanfaatkan untuk membantu dalam memahami alur kerja proses secara visual yang memungkinkan seseorang untuk melihat langkah-langkah, hubungan antar komponen, serta aliran data yang terlibat dalam proses tersebut dengan jelas.

# Konsep Variabel Lokal dan Variabel Global

Konsep variabel lokal dan variabel global merupakan sebuah prinsip dasar dalam pemrograman yang menunjukkan cara suatu variabel dideklarasikan dan diakses dalam suatu program. Kedua variabel tersebut memiliki karakteristik yang sama, namun terdapat perbedaan dalam pendeklarasiannya. Variabel lokal dideklarasikan di dalam suatu blok kode tertentu dan hanya dapat digunakan dan diakses pada tempat di mana mereka dideklarasikan. Apabila diakses di luar blok kodenya, maka akan menimbulkan suatu *error*. Sedangkan variabel global adalah variabel yang pendeklarasiannya di luar semua blok kode dan dapat digunakan di seluruh program, termasuk dalam blok instruksi maupun fungsi. Selain itu, perbedaan selanjutnya adalah pada saat penggunaannya. Variabel lokal digunakan untuk menyimpan data yang hanya relevan untuk bagian tertentu dalam program, sedangkan variabel global untuk data yang dibutuhkan pada banyak bagian program. Dari sisi keamanan, variabel lokal lebih aman karena hanya dapat diakses di dalam blok kode tertentu dibandingkan variabel global yang dapat diakses dan diubah oleh banyak bagian program.

# Konsep Fungsi dan Prosedur

Konsep ini merupakan bagian penting dalam pemrograman untuk mengorganisir dan memecah kode ke dalam bagian-bagian yang lebih kecil serta dapat digunakan kembali. Dalam bahasa C/C++, fungsi dan prosedur memiliki konsep yang sama meskipun memiliki perbedaan dalam nilai yang dikembalikan.

## Fungsi

Fungsi adalah suatu bloke kode yang dapat menerima *input*, mengoperasikannya, serta menampilkan sebuah *output* (nilai yang dikembalikan). Untuk mengembalikan nilai, suatu fungsi biasanya diakhiri dengan kata kunci ‘return’dalam bahasa pemrograman C/C++. Berikut merupakan pengaplikasian fungsi dalam pemrograman dengan bahasa C/C++.

|  |
| --- |
| #include <stdio.h>  int penjumlahan(int x, int y) {  return x + y;  } |

**Kode Program 4.1** Aplikasi Fungsi dalam Bahasa Pemrograman C

Kode Program 4.1 merupakan contoh pengaplikasian konsep fungsi dalam pemrograman dengan bahasa C. X dan y pada kode program di atas merupakan dua buah parameter yang masing-masing akan diberikan nilai *integer* sebagai argumen. Dengan kata kunci ‘return’, maka kode program di atas akan mengembalikan nilai berupa hasil dari penjumlahan variabel x dan y.

## Prosedur

Prosedur adalah blok kode yang menerima *input*, mengoperasikannya, namun tidak mengembalikan suatu nilai. Dalam C/C++, prosedur biasanya dicirikan dengan kata kunci ‘void’ dan tidak mengembalikan nilai. Berikut adalah pengaplikasian prosedur dalam pemrograman dengan bahasa C/C++.

|  |
| --- |
| #include <stdio.h>  void pesan\_pembuka () {  printf(“Welcome! \n”);  } |

**Kode Program 4.2** Aplikasi Prosedur dalam Bahasa Pemrograman C

Kode Program 4.2 merupakan contoh pengaplikasian konsep prosedur dalam pemrograman dengan bahasa C. Dengan kata kunci ‘void’, maka kode program di atas tidak akan mengembalikan nilai apapun saat dijalankan. Saat kode program ini dijalankan, maka output yang akan ditampilkan adalah *“Welcome!”*.

## Perbedaan Fungsi dan Prosedur

Fungsi biasanya digunakan pada saat ingin melakukan penghitungan atau operasi yang mengembalikan hasil, sedangkan ketika ingin menjalankan suatu tugas tanpa mengembalikan nilai maka menggunakan prosedur. Apabila ingin menggunakan fungsi, maka tipe datanya harus ditentukan terlebih dahulu. Sedangkan saat ingin menggunakan prosedur, maka tipe data yang digunakan adalah ‘void’.

# Konsep Mengirim Argumen

Mengirim argumen ke sebuah fungsi dalam bahasa pemrograman dapat dilakukan dengan berbagai cara, contohnya dengan konsep *passing by value* dan *passing by reference*. Namun, penggunaan konsep tersebut tergantung pada bahasa pemrogramaan yang digunakan.

## **Passing by Value**

Konsep ini merupakan konsep yang mengirim nilai dari argumen ke dalam fungsi dan salinan nilai argumen tersebut akan digunakan di dalam fungsi. Dalam konsep ini, segala perubahan yang terjadi di dalam fungsi tidak akan memengaruhi variabel asli di luar fungsi terkait. Berikut merupakan contoh penerapan konsep *passing by value* dalam program dengan bahasa C.

|  |
| --- |
| #include <stdio.h>  void incrementByValue(int a) {  a++;  }  int main() {  int nilai = 9;  printf(“Nilai sebelum fungsi: %d\n”, nilai);  incrementByValue(nilai);  printf(“Nilai setelah fungsi: %d\n”, nilai);  return 0;  } |

**Kode Program 5.1** Penerapan Konsep Passing by Value dalam Bahasa Pemrograman C

Kode Program 5.1 merupakan contoh penerapan konsep *passing by value* dalam suatu program. Dengan kata kunci ‘incrementByValue’, maka *output* yang akan ditampilkan oleh sistem pada kedua kasus tetap bernilai 9. Karena pada dasarnya dengan kata kunci tersebut, nilai yang akan dicetak adalah nilai dari parameter ‘nilai’ yang sebelumnya sudah dideklarasikan terlebih dahulu.

## Passing by Reference

Berbeda dari konsep *passing by value*, *pasing by reference* merupakan suatu konsep di mana fungsi akan bekerja langsung dengan variabel asli di luar fungsi dan setiap perubahan yang terjadi di dalam fungsi tersebut akan memengaruhi variabel aslinya. Dalam konsep ini, alamat memori (referensi) dari argumen akan dikirim ke dalam suatu fungsi. Berikut merupakan contoh penerapan konsep *passing by reference* dalam program dengan bahasa C.

|  |
| --- |
| #include <stdio.h>  void incrementByReference(int \*a) {  (\*a)++;  }  int main() {  int nilai = 9;  printf(“Nilai sebelum fungsi: %d\n”, nilai);  incrementByReference(&nilai);  printf(“Nilai setelah fungsi: %d\n”, nilai);  return 0;  } |

**Kode Program 5.2** Penerapan Konsep Passing by Reference dalam Bahasa Pemrograman C

Kode Program 5.2 merupakan contoh penerapan konsep *passing by reference* dalam suatu program. Saat kode program di atas dijalankan, maka *output* yang akan ditampilkan oleh sistem ada dua, yaitu 9 sebagai nilai sebelum fungsi dan 10 sebagai nilai setelah fungsi. Nilai yang ditampilkan bisa berbeda karena nilai variabel ‘nilai’ telah diubah oleh fungsi‘incrementByReference’.

# Art of Programming

Seni penulisan program adalah seni yang mengarah pada proses penciptaan dan perancangan sebuah program komputer dengan cara yang efektif, efisien, juga terstruktur. Tidak hanya berbicara tentang aspek visual suatu program, *art of programming* juga mencakup berbagai aspek, termasuk pemilihan struktur data yang sesuai, pemahaman akan aturan *syntax* serta gaya penulisan bahasa pemrograman yang digunakan, pengorganisasian kode, dan juga pemahaman mendalam tentang algoritma. Selain itu, seni penulisan program juga menyangkut kemampuan untuk memecahkan suatu permasalahan yang kompleks dengan memberikan solusi dalam bentuk kode yang dapat dibaca dan dimengerti oleh manusia juga komputer. Berikut merupakan contoh penerapan *art of programming* dalam program dengan bahasa C.

|  |
| --- |
| #include <stdio.h>  int main() {  int n, bil, terbesar;  printf(“Masukkan jumlah bilangan yang dimiliki:  scanf(“%d”, &n);  printf(“Masukkan bilangan ke-1: “);  scanf(“%d”, &terbesar);  for (int i = 2; i <= n; i++) {  printf(“Masukkan bilangan ke-%d: ”, i);  scanf(“%d”, &bil);  if (bil > terbesar) {  terbesar = bil;  }  }  printf(“Bilangan terbesar adalah : %d\n”, terbesar);  return 0;  } |

**Kode Program 6.1** Penerapan Art of Programming dalam Bahasa Pemrograman C

Kode Program 6.1 merupakan contoh penerapan *art of programming* dalam pemrograman dengan bahasa C. Kode program di atas akan menentukan bilangan terbesar berdasarkan data yang akan dimasukkan. Dengan demikian, kode program di atas menunjukkan prinsip-prinsip dasar pemrograman yang cermat dan terstruktur agar mudah dipahami.

**DAFTAR PUSTAKA**

Faulina, A. R. (2023, Maret 14). *https://www.sekawanmedia.co.id/blog/pengertian-dan-simbol-flowchart/*. Diambil kembali dari https://www.sekawanmedia.co.id/: https://www.sekawanmedia.co.id/blog/pengertian-dan-simbol-flowchart/

Flowchart, Z. (2023, Agustus 16). *https://www.zenflowchart.com/simbol-flowchart#Top*. Diambil kembali dari https://www.zenflowchart.com/: https://www.zenflowchart.com/simbol-flowchart#Top

Koding, M. (2016, November 10). *https://www.mahirkoding.com/variabel-global-dan-variabel-lokal-dalam-bahasa-c/*. Diambil kembali dari https://www.mahirkoding.com/: https://www.mahirkoding.com/variabel-global-dan-variabel-lokal-dalam-bahasa-c/

MateriDosen. (2017, Januari 6). *https://www.materidosen.com/2017/01/perbedaan-fungsi-dan-prosedur-dalam.html*. Diambil kembali dari https://www.materidosen.com/: https://www.materidosen.com/2017/01/perbedaan-fungsi-dan-prosedur-dalam.html

Rouse, M. (2017, Februari 2). *https://www.techopedia.com/definition/28423/the-art-of-computer-programming-taocp*. Diambil kembali dari https://www.techopedia.com/: https://www.techopedia.com/definition/28423/the-art-of-computer-programming-taocp

Tekno, H. T. (2022, Maret 9). *https://kumparan.com/how-to-tekno/tipe-data-c-ini-struktur-dan-variabelnya-1xeJz0I1kPx/3*. Diambil kembali dari https://kumparan.com/: https://kumparan.com/how-to-tekno/tipe-data-c-ini-struktur-dan-variabelnya-1xeJz0I1kPx/3

Wibowo, A. (2018, Maret 12). *https://www.wapwibowo.com/2018/12/pass-by-value-dan-pass-by-reference.html*. Diambil kembali dari https://www.wapwibowo.com/: https://www.wapwibowo.com/2018/12/pass-by-value-dan-pass-by-reference.html