1. **Tujuan Praktikum**

Mahasiswa mampu memahami dan menerapkan konsep tipe data, variabel, konstanta, dan operator dalam bahasa pemrograman C.

1. **Teori Singkat**

**Tipe Data**

Informasi yang dinyatakan dalam bentuk konstan atau variabel disebut data. Konstan melambangkan nilai yang tetap, sedangkan variabel mengacu pada nilai yang dapat dimodifikasi selama proses berjalan. Data dapat dikelompokkan menjadi lima jenis utama, yang dikenal sebagai tipe data dasar. Kelima tipe data dasar tersebut meliputi:

* Integer (bilangan bulat)
* Float (bilangan real dengan presisi tunggal)
* Double (bilangan real dengan presisi ganda)
* Char (karakter)
* Void (tak-berjenis)

Tipe data dalam bahasa pemrograman C adalah konsep yang menentukan jenis nilai yang dapat disimpan dalam variabel serta cara di mana nilai tersebut akan diolah. Tipe data menggambarkan ukuran memori yang diperlukan untuk menyimpan nilai tertentu dan operasi yang dapat dilakukan pada nilai tersebut. Dalam bahasa pemrograman C, terdapat beberapa tipe data dasar yang dapat digunakan.

| **Tipe Data** | | **Ukuran (bytes)** | **Format Specifier** | **Deskripsi** |
| --- | --- | --- | --- | --- |
| int | | Setidaknya 2, biasanya 4 | %d, %i | Menyimpan bilangan bulat. |
| char | | 1 | %c | Menyimpan karakter tunggal. |
| float | | 4 | %f | Menyimpan bilangan desimal dengan presisi standar. |
| double | | 8 | %lf | Menyimpan bilangan desimal dengan presisi lebih tinggi daripada float. |
| short int | | Biasanya 2 | %hd | Menyimpan bilangan bulat dengan rentang lebih kecil daripada int. |
| unsigned int | | Setidaknya 2, biasanya 4 | %u | Menyimpan bilangan bulat non-negatif. |
| long int | | Setidaknya 4, biasanya 8 | %ld, %li | Menyimpan bilangan bulat dengan rentang lebih besar daripada int. |
| long long int | | Setidaknya 8 | %lld, %lli | Menyimpan bilangan bulat dengan rentang lebih besar daripada long int. |
| unsigned long int | | Setidaknya 4 | %lu | Menyimpan bilangan bulat non-negatif dengan rentang lebih besar. |
| unsigned long long int | | Setidaknya 8 | %llu | Menyimpan bilangan bulat non-negatif dengan rentang lebih besar. |
| signed char | | 1 | %c | Menyimpan karakter dengan tanda. |
| unsigned char | | 1 | %c | Menyimpan karakter tanpa tanda. |
| long double | | Setidaknya 10, biasanya 12 atau 16 | %Lf | Menyimpan bilangan desimal dengan presisi yang lebih tinggi daripada double. |
| **Konsep/Tipe Data** | **Deskripsi** | | | |
| Tipe Data Turunan | Tipe data yang berasal dari tipe data dasar. Misalnya: array, pointer, tipe data fungsi, struktur, dll. Tutorial selanjutnya akan membahasnya. | | | |
| Bool | Tipe data boolean untuk menyimpan nilai kebenaran (true atau false). | | | |
| Enumerasi (Enum) | Tipe data yang memungkinkan Anda mendefinisikan konstanta dengan label untuk meningkatkan keterbacaan kode. | | | |
| Tipe Data Kompleks | Tipe data seperti complex (meskipun tidak standar dalam C) untuk bekerja dengan bilangan kompleks dalam matematika. | | | |

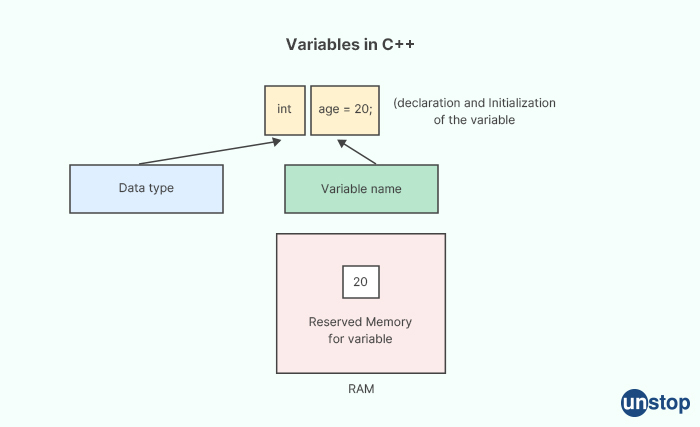
**Variabel**

Aturan dalam penulisan variable:

* Nama variabel hanya boleh terdiri dari huruf (baik huruf kapital maupun huruf kecil), angka, dan garis bawah (underscore).
* Huruf pertama dari sebuah variabel haruslah huruf atau garis bawah.
* Tidak ada aturan mengenai seberapa panjang nama variabel (identifier) dapat menjadi. Namun, Anda mungkin akan menghadapi masalah pada beberapa kompiler jika nama variabel lebih panjang dari 31 karakter. Catatan: Selalu disarankan untuk memberikan nama yang bermakna pada variabel. Sebagai contoh: 'firstName' adalah nama variabel yang lebih baik daripada 'fn'. C adalah bahasa pemrograman yang berjenis tipe data kuat. Ini berarti bahwa tipe data variabel tidak dapat diubah setelah dideklarasikan. Sebagai contoh: int number = 5; // variabel integer number = 5.5; // error double number; // error Di sini, tipe data dari variabel 'number' adalah int. Anda tidak dapat memberikan nilai pecahan (desimal) 5.5 pada variabel ini. Juga, Anda tidak dapat mengubah ulang tipe data variabel menjadi double. Untuk menyimpan nilai desimal dalam C, Anda perlu mendeklarasikan tipe datanya sebagai double atau float."

**Deklarasi variabel**

Deklarasi variabel dalam bahasa pemrograman C adalah langkah awal dalam menginformasikan kompiler tentang tipe data dan nama variabel yang akan digunakan dalam program. Ini memungkinkan kompiler mengalokasikan ruang memori yang diperlukan untuk variabel tersebut. Perhatikan ilustrasi dibawah ini.

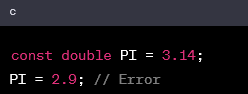


**Konstanta**

Jika Anda ingin mendefinisikan sebuah variabel yang nilainya tidak dapat diubah, Anda dapat menggunakan kata kunci ‘**const**’ atau ‘**define**’. Ini akan membuat sebuah konstanta. Sebagai contoh,

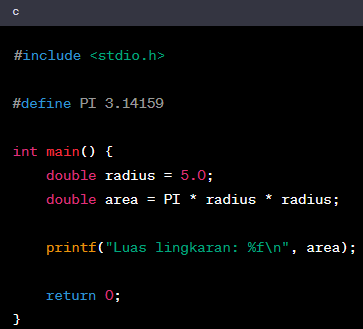


Perhatikan bahwa kita telah menambahkan kata kunci ‘**const**’. Di sini, PI adalah konstanta simbolik; nilainya tidak dapat diubah.



Pada contoh di atas, konstanta PI diberikan nilai awal 3.14. Namun, ketika kita mencoba untuk mengubah nilai PI menjadi 2.9, akan menghasilkan kesalahan (error) karena nilai konstanta tidak dapat diubah setelah dideklarasikan.

Selain menggunakan itu, untuk mendefinisikan konstanta bias menggunakan keyword ‘**define**’, berikut contoh penggunaannya:



**Operator**

Operator adalah simbol atau tanda dalam pemrograman yang digunakan untuk melakukan operasi tertentu pada satu atau lebih operand (nilai atau variabel). Operasi ini bisa berupa perhitungan matematika, pembandingan, logika, dan lain-lain. Operator memungkinkan Anda untuk menggabungkan nilai-nilai atau melakukan manipulasi data dalam berbagai cara.

| **Jenis Operator** | **Contoh Operator** | **Deskripsi** |
| --- | --- | --- |
| Operator Matematika | + | Penambahan: Menambahkan nilai dari operand. |
| - | Pengurangan: Mengurangkan nilai operand pertama dengan nilai operand kedua. |
| \* | Perkalian: Mengalikan nilai operand. |
| / | Pembagian: Membagi nilai operand pertama dengan nilai operand kedua. |
| % | Modulus: Menghasilkan sisa hasil bagi dari pembagian operand pertama oleh operand kedua. |
| Operator Pembanding | == | Sama dengan: Memeriksa apakah dua nilai sama. |
| != | Tidak sama dengan: Memeriksa apakah dua nilai tidak sama. |
| < | Kurang dari: Memeriksa apakah nilai pertama lebih kecil dari nilai kedua. |
| > | Lebih dari: Memeriksa apakah nilai pertama lebih besar dari nilai kedua. |
| <= | Kurang dari atau sama dengan: Memeriksa apakah nilai pertama kurang dari atau sama dengan nilai kedua. |
| >= | Lebih dari atau sama dengan: Memeriksa apakah nilai pertama lebih besar dari atau sama dengan nilai kedua. |

**Escape Sequence**

| **Escape Sequence** | **Karakter** | **Deskripsi** |
| --- | --- | --- |
| \b | Backspace | Memundurkan kursor satu karakter. |
| \f | Form feed | Menggeser kursor ke halaman berikutnya. |
| \n | Newline | Pindah ke baris berikutnya (baris baru). |
| \r | Return | Pindah ke awal baris saat ini (kembali). |
| \t | Horizontal tab | Menambahkan tab horizontal. |
| \v | Vertical tab | Menambahkan tab vertikal. |
| \\ | Backslash | Menampilkan karakter garis miring terbalik. |
| \' | Single quotation mark | Menampilkan tanda kutip satu. |
| \" | Double quotation mark | Menampilkan tanda kutip ganda. |
| \? | Question mark | Menampilkan tanda tanya. |
| \0 | Null character | Menampilkan karakter null (nol). |

1. **Alat dan Bahan**
2. Komputer
3. DevC++
4. Jobsheet
5. **Langkah Kerja**
6. Hidupkan komputer.
7. Buka dan jalankan tools IDE DevC++
8. Buat file project latihan.

**Kerjakan Latihan (Percobaan) berikut**

1. Menghitung kebutuhan kalori manusia dewasa berdasarkan jenis kelamin, jenis pekerjaan, umur, tinggi badan, berat badan

Rumus Harris-Benedict:

Jumlah kebutuhan kalori per hari didapatkan dengan memperhitungkan BMR (Basal Metabolic Rate) dan tingkat aktivitas harian seseorang. Rumus yang paling banyak digunakan oleh ahli gizi untuk menghitung BMR adalah Rumus Harris-Benedict. Rumus ini dihitung berdasarkan usia, jenis kelamin, berat badan, dan tinggi badan.

Untuk laki-laki:

* (88,4 + 13,4 x berat dalam kilogram) + (4,8 x tinggi dalam sentimeter) - (5,68 x usia dalam tahun)

Untuk wanita:

* (447,6 + 9,25 x berat dalam kilogram) + (3,10 x tinggi dalam sentimeter) - (4,33 x usia dalam tahun)

Hasil perhitungan BMR kemudian dikalikan dengan angka aktivitas harian rata-rata orang tersebut. Angka ini berkisar antara 1,2–1,9 tergantung dari seberapa tinggi aktivitas harian seseorang. Semakin jarang seseorang melakukan aktivitas fisik, semakin rendah pula angka aktivitas hariannya.

* faktor aktivitas fisik:

rendah: 1.2

sedang: 1.3

berat : 1.4

* Perhitungan BMR:

pria = (88.4 \* 13.8 \* BB) + (4.8 \* TB) - (5.68 \* USIA)

wanita = (447.6 + 9.25 \* BB) + (3.10 \* TB) - (4.33 \* USIA)

* Total kebutuhan kalori:

kalori pria:

* kalori aktivitas rendah = BMR pria \* 1.2
* kalori aktivitas sedang = BMR pria \* 1.3
* kalori aktivitas berat = BMR pria \* 1.4

kalori wanita:

* kalori aktivitas rendah = BMR wanita \* 1.2
* kalori aktivitas sedang = BMR wanita \* 1.3
* kalori aktivitas berat = BMR wanita \* 1.4

Berikut perhitungan dasar untuk perempuan:

* BMR = 655 + (1,8 x tinggi dalam cm) + (9,6 x berat dalam kilogram) - (4,7 x umur dalam tahun)

Untuk perempuan dengan usia 30, tinggi 167,6 cm dan berat badan 68 kilogram, maka perhitungan BMR-nya akan menjadi: 655 + 301,7 + 653 - 141 = sekitar 1.470 kalori per hari.

Berikut perhitungan dasar untuk pria:

* BMR = 66 + (13,7 x berat dalam kilogram) + (5 x tinggi dalam cm) - (6,7 x umur dalam tahun).

Untuk contoh hasil perempuan, misalnya, maka bisa mengalikan hasil BMR tersebut dengan tingkat aktivitas. Kali jumlah BMR dengan 1,2 jika kamu tidak aktif; 1,375 jika melakukan olahraga ringan satu hingga tiga kali seminggu; 1,55 jika melakukan olahraga ringan 6-7 kali seminggu; kalikan dengan 1,75 jika sangat aktif (olahraga keras setiap hari atau berolahraga dua kali sehari) dan 1,9 jika ekstra aktif (latihan keras dua kali atau lebih per hari).

Jadi untuk seseorang dengan bobot 68 kilogram yang cukup aktif, maka akan menjadi: 1,470 x 1,55 = sekitar 2.280. Hasil itu merupakan berapa banyak kalori yang dibakar tubuh pada hari tertentu.

|  |
| --- |
| /\* Nama File : latihan2A.c  Programmer : tuliskan nama dan nim anda  Tgl. pembuatan : tuliskan tanggal hari praktikum anda  Deskripsi : program ini melakukan perhitungan kebutuhan kalori  manusia dewasa berdasarkan jenis kelamin, jenis pekerjaan,  umur, tinggi badan, berat badan  \*/  #include <stdio.h>  #include <stdlib.h>  //konstanta  #define RENDAH 1.2  #define SEDANG 1.3  #define BERAT 1.4  main()  {  double beratBadan, tinggiBadan, usia;  double bmrPria, bmrWanita;  double kaloriPriaRendah, kaloriPriaSedang, kaloriPriaBerat;  double kaloriWanitaRendah, kaloriWanitaSedang, kaloriWanitaBerat;    printf("\n\tMenghitung Kebutuhan Kalori Harian\n");  printf("\t===================================\n\n");    //kebutuhan masukkan  printf("Entri Data Tubuh:\n");  printf("-----------------\n");  printf("Berat Badan dalam kg : ");  scanf("%lf", &beratBadan); fflush(stdin);  printf("Tinggi Badan dalam meter: ");  scanf("%lf", &tinggiBadan); fflush(stdin);  printf("Usia dalam tahun : ");  scanf("%lf", &usia); fflush(stdin);    //menghitung BMR  bmrPria = 66 + (13.7 \* beratBadan) + (5 \* tinggiBadan) - (6.7 \* usia);  bmrWanita = 655 + (1.8 \* tinggiBadan) + (9.6 \* beratBadan) - (4.7 \* usia);    //menghitung kebutuhan kalori pria  kaloriPriaRendah = bmrPria \* RENDAH;  kaloriPriaSedang = bmrPria \* SEDANG;  kaloriPriaBerat = bmrPria \* BERAT;    //menghitung kebutuhan kalori wanita  kaloriWanitaRendah = bmrWanita \* RENDAH;  kaloriWanitaSedang = bmrWanita \* SEDANG;  kaloriWanitaBerat = bmrWanita \* BERAT;    //menamp[ilkan informasi ke layar  printf("\nKebtuhan Kalori Pria:\n");  printf("\tPria Aktivitas Rendah : %0.2lf Kalori per hari\n", kaloriPriaRendah);  printf("\tPria Aktivitas Sedang : %0.2lf Kalori per hari\n", kaloriPriaSedang);  printf("\tPria Aktivitas Berat : %0.2lf Kalori per hari\n", kaloriPriaBerat);    printf("\nKebtuhan Kalori Wanita:\n");  printf("\tWanita Aktivitas Rendah: %0.2lf Kalori per hari\n", kaloriWanitaRendah);  printf("\tWanita Aktivitas Sedang: %0.2lf Kalori per hari\n", kaloriWanitaSedang);  printf("\tWanita Aktivitas Berat : %0.2lf Kalori per hari\n", kaloriWanitaBerat);  } |

1. Menghitung kecepatan suatu kendaraan berdasarkan jarak tempuh dan waktu tempuh

Analisis Kebutuhan:

* Kebutuhan output
  + Tampilan:

Jarak Tempuh (dlm km): XXX Km

Waktu tempuh: XX:XX:XX

Kecepatan (km/jam) : XX km/jam

* Kebutuhan input
  + Tampilan:

Ketikkan jarak tempuh dalam kilometer: ??

Ketikkan waktu tempuh (format: jj:mm:dd): ??:??:??

* Kebutuhan proses
  + Menghitung total detik dari format jj:mm:dd menjadi detik

totalDetik  jj \* 3600 + mm \* 60 + dd

* + Menkonversi jarak dalam Km menjadi meter:

Meter  km \* 1000.0

* + Menghitung kecepatan:

Cepat  meter/totalDetik;

* Kebutuhan variable:
  + Untuk menyimpan nilai total detik, nama variable totalDetik bertipe integer
  + Untuk menyimpan nilai jam, nama variable jj bertipe integer
  + Untuk menyimpan nilai menit, nama variable mm bertipe integer
  + Untuk menyimpan nilai detik, nama variable dd bertipe integer
  + Untuk menyimpan nilai jarak dalam Km, nama variable km bertipe real
  + Untuk menyimpan nilai jarak dalam meter, nama variable meter bertipe real
  + Untuk menyimpan nilai kecepatan, nama variable cepat bertipe real
  + Untuk menyimpan tanda pemisah format waktu (:), nama variable titikDua bertipe char
* Algoritma HitungKecepatan (input, output)

{Menghitung kecepatan kendaraan berdasarkan jarak tempuh dan waktu tempu}

Deklarasi

Variable:

totalDetik, jj, mm, dd: integer

meter, km, cepat: real

titikDua: char

deskripsi

start

read(km)

read(jj, titikDua, mm, titikDua, dd)

//Menghitung total detik dari format jj:mm:dd menjadi detik

totalDetik  jj \* 3600 + mm \* 60 + dd

//Menkonversi jarak dalam Km menjadi meter:

Meter  km \* 1000.0

//Menghitung kecepatan:

Cepat  meter/totalDetik;

Write(cepat)

Stop

|  |
| --- |
| /\* Nama File : latihan2B.c  Programmer : tuliskan nama dan nim anda  Tgl. pembuatan : tuliskan tanggal hari praktikum anda  Deskripsi : program ini melakukan perhitungan kecepatan suatu  kendaraan berdasarkan jarak tempuh dan waktu tempuh  \*/  #include <stdio.h>  #include <stdlib.h>  main(){  int totalDetik, jj, mm, dd;  double meter, km, cepat;  char titikDua;    printf("Program Menghitung Kecepatan\n\n");  printf("Ketikkan jarak tempuh dalam Km: ");  scanf("%lf", &km); fflush(stdin);  printf("Ketikkan waktu tempuh (ex: 02:05:15): ");  scanf("%d%c%d%c%d", &jj, &titikDua, &mm, &titikDua, &dd);  //Menghitung total detik dari format jj:mm:dd menjadi detik  totalDetik = jj \* 3600 + mm \* 60 + dd;  //Menkonversi jarak dalam Km menjadi meter:  meter = km \* 1000.0;  //Menghitung kecepatan:  cepat = meter/totalDetik;  printf("Jarak Tempu : %0.2lf Km\n", km);  printf("Waktu Tempu : %02d%c%02d%c%02d\n", jj, titikDua, mm, titikDua,dd);  printf("Kecepatan : %0.2lf m/detik\n", cepat);  } |

1. **Evaluasi**

Buatlah Algoritma dalam bentuk Flowchart kemudian tuliskan ke dalam program Bahasa C untuk permasalahan dibawah ini, dan simpan dengan nama TUGAS2A.

1. Menkonversi waktu jam ke total detik dan sebaliknya

Langkah-langkah:

* konversi jj:mm:dd ke total detik
* baca waktu dalam format jj:mm:dd
* konversi ke total detik
* Total detik = jj x 3600 + mm x 60 + dd
* tampilkan informasi
* konversi total detik ke format jj:mm:dd
* baca nilai total detik
* konversi ke format jj:mm:dd
* Jj = total detik/3600
* Mm = total detik % 3600 / 60
* Dd = total detik % 3600 % 60
* tampilkan informasi

1. Menghitung biaya cicilan kredit barang berdasarkan harga pokok, bunga kredit, uang muka, banyaknya kali (waktu) cicilan.
2. **Kesimpulan**

Setelah menyelsaikan jobsheet ini mahasiswa mampu memahami dan mengidentifikasi berbagai tipe data dalam pemrograman, serta mengenal prinsip deklarasi variabel sesuai tipe data yang dipilih. Mereka akan mampu menjelaskan penggunaan konstanta dengan kata kunci **const** atau praprosesor **#define** untuk mengamankan nilai tetap. Mahasiswa diharapkan memiliki kemampuan mengoperasikan operator matematika, perbandingan, dan logika untuk mengolah data dengan benar, serta mampu memahami urutan escape yang membantu mengatasi karakter khusus dalam string.

1. **Referensi**

|  |
| --- |
| Utama |
| * Rinaldi Munir. 2016. *Algoritma dan Pemrograman*. Bandung. Informatika ITB * Noel Kalicharan. 2015. *Learn to Program with C*. New York, Springer-Science * Harry H. Chaudhary. 2014. *C Programming Step by Step*. LLC USA. Amazong Inc. |
| Pendukung |
| * Mike McGrath. 2015. *Coding for Beginners*. Leamington Spa. Easy Step Limited. * Dan Gookin. 2014. *Beginning Programming with C for Dummies*. New Jersey. John Wiley & Sons. * [www.tutorialspoint.com](http://www.tutorialspoint.com) * [www.javatpoint.com](http://www.javatpoint.com) * www.programiz.com |