**MAKALAH**

**OPERATOR DALAM BAHASA C**



Oleh:

Nama : L Hafid Alkhair

NIM : 2023903430060

Kelas : TRKJ 1.C

Jurusan : Teknologi Informasi dan Komputer

Program Studi : Teknologi Rekayasa Komputer Jaringan

Dosen Pembimbing : Indrawati, SST. MT



**JURUSAN TEKNOLOGI, KOMPUTER, DAN INFORMASI**

**PRODI TEKNOLOGI REKAYASA KOMPUTER DAN JARINGAN**

**POLITEKNIK NEGERI LHOKSEUMAWE**

**TAHUN AJARAN 2022/2023**

**OPERATOR**

Sebelum masuk ke jenis-jenis operator di dalam bahasa C, terdapat istilah operand dan operator. Operand adalah nilai asal yang dipakai dalam sebuah proses operasi. Sedangkan Operator adalah instruksi yang diberikan untuk mendapatkan hasil dari proses tersebut.

Biasanya operator berupa karakter matematis atau perintah singkat sederhana. Sebagai contoh, pada operasi: 10 + 2. Angka 10 dan 2 disebut sebagai operand, sedangkan tanda tambah (karakter +) adalah operator.

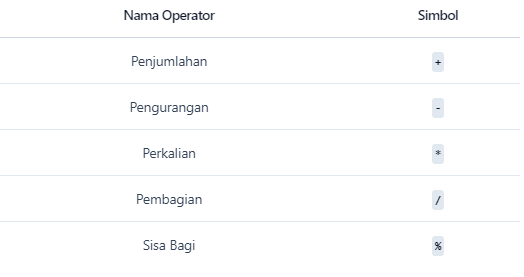
1. **ARITMATIKA**

Operator adalah simbol atau tanda yang jika diletakkan pada dua buah operand (data) dapat menghasilkan sebuah hasil, operator berupa simbol yang digunakan untuk menyusun suatu ekspresi dengan melibatkan satu atau beberapa operand. Contohnya pada matematika dimana tanda tambah (‘+’) jika diletakkan di antara dua buah angka akan menghasilkan angka lain hasil pertambahan dari dua angka tersebut.

Enam Macam Operator yang Harus diketahui pada C

1. Operator Artimatika;
2. Operator Penugasan;
3. Operator Pembanding;
4. Operator Logika;
5. Operator Bitwise;
6. dan Operator Lain-lain.
7. **Operator Aritmatika**

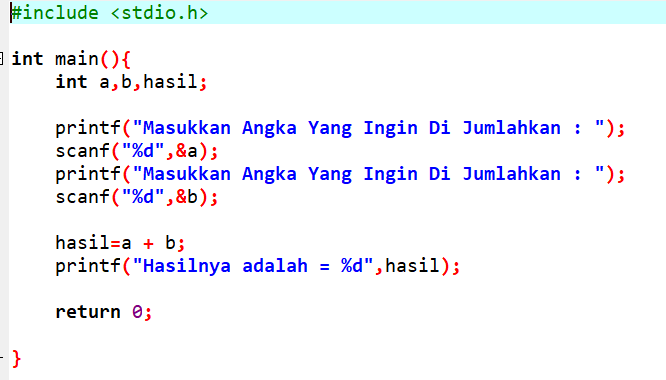
Operator aritmatika merupakan operator yang biasa kita temukan dalam operasi matematika. Sedangkan aritmatika adalah cabang ilmu matematika yang membahas perhitungan sederhana. Mulai dari penambahan, pembagian, pengurangan, dan perkalian.Namun, ada salah satu sifat operator yang juga digunakan daam operator aritmatika, yaitu Unary. Sifat operator ini menjelasakan bahwa simbol (+) merupakan bentuk positif dalam sebuah bilangan dan (-) merupakan bentuk negatifnya.

Operator ini terdiri dari:

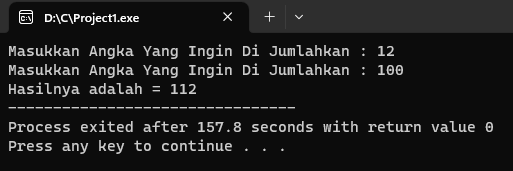
Mari kita coba dalam contoh program:

Buatlah program baru bernama operator\_aritmatika.c, kemudian isi dengan kode berikut.

Contoh Program Penjumlahan



Output :



Analisi ;

Disini berisi code code baru, yaitu int , code ini berfungsi untuk menentukan jenis variable, int bisa disebut integer , int ini digunakan untuk inputan angka-angka saja, int ini tidak mensupport angka koma atau decimal, anda bisa menggunakan code float untuk angka decimal, dan selanjutnya berisi variable, variable ini nanti berfungsi untuk menampung data yang dimasukkan pada code scanf.



Lalu ke tahap proses nya, jika rumus penjumlahan adalah 1+1=2 , jadi pada C sama, tetapi menggunakan variable, disini kita membuat 3 variable yaitu a,b, dan hasil, variable a sebagai penyimpan angka pertama dan variable b seabagai penyimpan angka kedua, dan variable hasil adalah penyimpan hasil dari rumus penjumlahan, untuk rumus nya sama seperti penjumlahan , seperti berikut :



Dan terakhir printf untuk menghasilkan hasil dari rumus penjumlahan tadi.

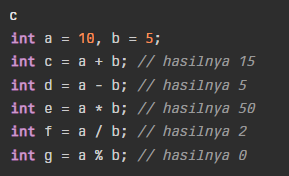


Jika anda ingin menggunakan rumus penjumlahan pada C , rumus nya berbeda , dimana = (samadengan) berada di belakang, tapi di C =(samadengan) berada di depan.

Operator aritmatika dalam bahasa C digunakan untuk melakukan operasi matematika pada variabel. Berikut adalah jenis-jenis operator aritmatika dalam bahasa C yang perlu diketahui:

* Penjumlahan (+): digunakan untuk menjumlahkan dua nilai atau variabel. Contoh: a + b.
* Pengurangan (-): digunakan untuk mengurangkan dua nilai atau variabel. Contoh: a - b.
* Perkalian (\*): digunakan untuk mengalikan dua nilai atau variabel. Contoh: a \* b.
* Pembagian (/): digunakan untuk membagi dua nilai atau variabel. Contoh: a / b.
* Modulus (%): digunakan untuk mendapatkan sisa pembagian dua nilai atau variabel. Contoh: a % b

Aturan prioritas operator berlaku di sini, dimana operator perkalian dan pembagian akan dijalankan terlebih dahulu daripada operator pengurangan dan penambahan. Untuk operator yang sama kuat (seperti penambahan dan pengurangan), akan diproses dari kiri ke kanan. Tanda kurung bisa dipakai untuk memprioritaskan operator tertentu.

Contoh penggunaan operator aritmatika dalam bahasa C:

1. **Pengertian Operator Assignment / Penugasan**

Pengertian Operator Assignment / Penugasan

Operator assignment adalah operator untuk memasukkan suatu nilai ke dalam variabel. Operator ini sebenarnya sudah sering kita pakai sepanjang tutorial bahasa C di Duniailkom. Dalam bahasa C, operator assignment menggunakan tanda sama dengan ( = ).

Pembacaan operasi assignment dilakukan dari kanan ke kiri, bukan dari kiri ke kanan seperti yang biasa kita pahami dalam matematika.

Kode berikut:

a = 500;

Berarti “memasukkan nilai 500 ke dalam variabel a”. Dalam bahasa pseudo code (jika anda membaca buku tentang algoritma), ini biasa ditulis dengan simbol panah ke kiri:

a <- 500

Operator assignment ini disebut juga sebagai operator penugasan.

Dalam prakteknya, operator assignment juga bisa dipakai “bertingkat” seperti contoh berikut:

**#include <stdio.h>**

**int main(void)**

**{**

**int a, b, c, d, e;**

**a = 5;**

**b = 3;**

**b = b + 1;**

**c = a + b;**

**d = c + c + a;**

**e = (c + d)\* a;**

**printf("Isi variabel a: %d \n",a);**

**printf("Isi variabel b: %d \n",b);**

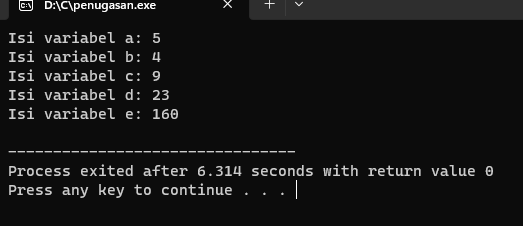
**printf("Isi variabel c: %d \n",c);**

**printf("Isi variabel d: %d \n",d);**

**printf("Isi variabel e: %d \n",e);**

**return 0;**

**}**

Hasil

Di awal kode program saya mendefinisikan 5 variabel, yakni a, b, c, d, dan e sebagai integer. Kemudian menginput angka 5 ke dalam variabel a (baris 6), dan angka 3 ke dalam variabel b (baris 7).

Di baris 8, perintah b = b + 1 mungkin bisa membuat bingung. Tapi intinya adalah, operator assignment di proses dari kanan ke kiri, jadi operasi tersebut bisa dibaca: “tambah isi variabel b dengan 1, lalu simpan kembali ke dalam variabel b”. Karena variabel b sudah berisi angka 3, maka hasil akhirnya variabel b akan bernilai 4.

Di baris 9, perintah c = a + b akan di proses sebagai c = 5 + 4, hasilnya c berisi angka 9.

Di baris 10, perintah d = c + c + a akan di proses sebagai d = 9 + 9 + 5, hasilnya c berisi angka 23.

Terakhir, di baris 11, perintah e = (c + d)\* a akan di proses sebagai e = (9 + 23) \* 5, hasilnya c berisi angka 160

1. **Operator Perbandingan / Relasional Bahasa C**

Operator perbandingan dipakai untuk membandingkan 2 buah nilai, apakah nilai tersebut sama besar, lebih kecil, lebih besar, dst. Hasil dari operator perbandingan ini adalah boolean True atau False, namun karena bahasa C tidak memiliki tipe data boolean bawaan, maka hasilnya adalah integer 1 atau 0.

Di dalam bahasa C, terdapat 6 operator perbandingan:

Operator Penjelasan Contoh Hasil

== Sama dengan 5 == 5 1 (true)

!= Tidak sama dengan 5 != 5 0 (false)

> Lebih besar 5 > 6 0 (false)

< Lebih kecil 5 < 6 1 (true)

>= Lebih besar atau sama dengan5 >= 3 1 (true)

<= Lebih kecil atau sama dengan 5 <= 5 1 (true)

Operator perbandingan ini biasa dipakai dalam proses pengambilan keputusan atau percabangan kode program. Sebagai contoh, jika angka pertama lebih besar dari kedua, maka jalankan perintah ini. Atau jika string password = ‘qwerty’ maka berikan hak akses.

Operator perbandingan kadang disebut juga dengan operator relasional karena kita membandingkan hubungan (relasi) sebuah nilai dengan nilai lainnya.

**#include <stdio.h>**

**int main(void)**

**{**

**int a;**

**a = 5 == 5;**

**printf("Isi variabel a: %d \n",a);**

**a = 5 != 5;**

**printf("Isi variabel a: %d \n",a);**

**a = 5 > 5;**

**printf("Isi variabel a: %d \n",a);**

**a = 5 >= 5;**

**printf("Isi variabel a: %d \n",a);**

**a = 5 < 5;**

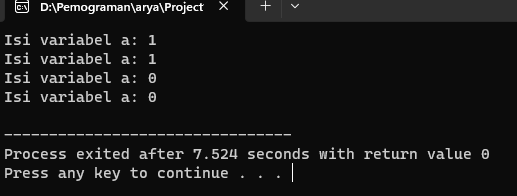
**printf("Isi variabel a: %d \n",a);**

**a = 5 <= 5;**

**printf("Isi variabel a: %d \n",a);**

**return 0;**

**}**



Jika yang dibandingkan berupa karakter, hasilnya dilihat dari urutan karakter ASCII. Dalam tabel ASCII, karakter ‘A’ memiliki nomor urut yang lebih rendah daripada ‘B’, sehingga ‘A’ > ‘B’ adalah false (0).

Contoh lainya

**#include <stdio.h>**

**int main() {**

**int a = 10;**

**int b = 10;**

**if (a == b) {**

**printf("a sama dengan b\n");**

**} else {**

**printf("a tidak sama dengan b\n");**

**}**

**if (a < b) {**

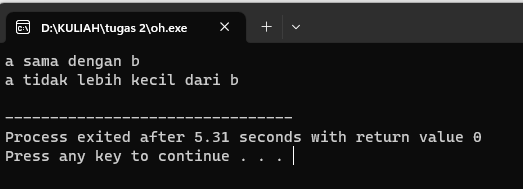
**printf("a lebih kecil dari b\n");**

**} else {**

**printf("a tidak lebih kecil dari b\n");**

**}**

**return 0;**

**}**

Dalam contoh di atas, kita menggunakan operator perbandingan (== dan <) untuk membandingkan nilai a dan b, dan kemudian mencetak hasil perbandingan tersebut ke layar. Hasil dari perbandingan tersebut akan menghasilkan nilai true atau false, yang digunakan untuk mengontrol alur program dengan menggunakan struktur kondisional if.

**Karakter ASCII**

Karakter ASCII (American Standard Code for Information Interchange) adalah representasi numerik dari karakter yang digunakan dalam komunikasi teks dan pemrosesan data. Setiap karakter dalam ASCII diberikan kode numerik antara 0 hingga 127. Berikut ini adalah beberapa karakter ASCII yang umum digunakan:

**Huruf Alphabet Besar (A-Z):**

'A' memiliki nilai ASCII 65.

'B' memiliki nilai ASCII 66.

...

'Z' memiliki nilai ASCII 90.

**Huruf Alphabet Kecil (a-z):**

'a' memiliki nilai ASCII 97.

'b' memiliki nilai ASCII 98.

...

'z' memiliki nilai ASCII 122.

**Angka (0-9):**

'0' memiliki nilai ASCII 48.

'1' memiliki nilai ASCII 49.

...

'9' memiliki nilai ASCII 57.

**Karakter Khusus Umum:**

Spasi (' ') memiliki nilai ASCII 32.

Tanda koma (',') memiliki nilai ASCII 44.

Tanda titik ('.') memiliki nilai ASCII 46.

Tanda tanya ('?') memiliki nilai ASCII 63.

Tanda seru ('!') memiliki nilai ASCII 33.

**Simbol Matematika:**

'+' memiliki nilai ASCII 43.

'-' memiliki nilai ASCII 45.

'\*' memiliki nilai ASCII 42.

'/' memiliki nilai ASCII 47.

'=' memiliki nilai ASCII 61.

**Karakter Khusus Lainnya:**

Tanda kutip satu (''') memiliki nilai ASCII 39.

Tanda kutip dua ('"') memiliki nilai ASCII 34.

Tanda kurung buka ('(') memiliki nilai ASCII 40.

Tanda kurung tutup (')') memiliki nilai ASCII 41.

Tanda kurung kurawal buka ('{') memiliki nilai ASCII 123.

Tanda kurung kurawal tutup ('}') memiliki nilai ASCII 125.

**Karakter Kontrol:**

P

Karakter kontrol, seperti Newline (ASCII 10) dan Tab (ASCII 9), digunakan untuk mengatur tata letak dan tampilan teks.

Ini hanya beberapa contoh karakter ASCII yang umum digunakan. Terdapat banyak karakter lainnya dalam kode ASCII, termasuk karakter khusus dan karakter kontrol yang memiliki nilai numerik unik. Pemrograman sering menggunakan representasi ASCII karakter dalam kode program untuk berbagai tugas, seperti manipulasi teks.

1. **Increment dan Decrement Bahasa C**

Operator increment dan decrement adalah sebutan untuk operasi seperti a++, dan a--. Ini sebenarnya penulisan singkat dari operasi a = a + 1 serta a = a – 1.

Increment digunakan untuk menambah variabel sebanyak 1 angka, sedangkan decrement digunakan untuk mengurangi variabel sebanyak 1 angka.

Penulisannya menggunakan tanda tambah 2 kali untuk increment, dan tanda kurang 2 kali untuk decrement. Penempatan tanda tambah atau kurang ini boleh di awal seperti ++a dan --a, atau di akhir variabel seperti a++ dan a--.

Dengan demikian terdapat 4 jenis increment dan decrement dalam bahasa C:

Operator Contoh Penjelasan

Pre-increment ++a Tambah a sebanyak 1 angka, lalu tampilkan hasilnya

Post-increment a++ Tampilkan nilai a, lalu tambah a sebanyak 1 angka

Pre-decrement --a Kurangi a sebanyak 1 angka, lalu tampilkan hasilnya

Post-decrement a-- Tampilkan nilai a, lalu kurangi a sebanyak 1 angka

Untuk melihat apa perbedaan dari setiap operator akan kita bahas dengan contoh kode program.

Contoh Kode Program Increment dan Decrement

**#include <stdio.h>**

**int main(void)**

**{**

**int a = 5;**

**a++;**

**printf("Isi variabel a: %d \n",a);**

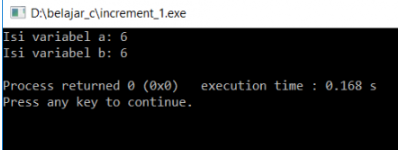
**int b = 5;**

**++b;**

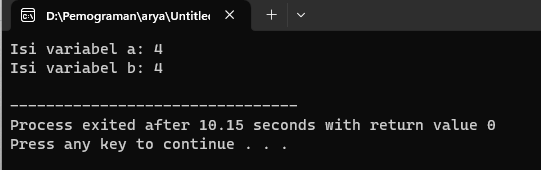
**printf("Isi variabel b: %d \n",b);**

**return 0;**

**}**



Di sini saya mengisi angka 5 sebagai nilai awal untuk variabel a dan b. Hasilnya, setelah operasi a++ dan ++b, kedua variabel sama-sama berisi angka 6. Artinya variabel a dan b akan bertambah 1 angka.

Hal yang sama juga berlaku di decrement:

Saya kembali mengisi angka 5 sebagai nilai awal untuk variabel a dan b. Hasilnya, setelah operasi a-- dan --b, kedua variabel sama-sama berisi angka 4. Artinya variabel a dan b akan berkurang 1 angka.

Perbedaan Pre-Increment dengan Post-Increment

Jadi apa beda antara pre-increment (++a) dengan post-increment (a++)? Perbedaannya baru terlihat jika kita akses pada saat operasi berlangsung, seperti contoh berikut:

**#include <stdio.h>**

**int main(void)**

**{**

**int a = 5;**

**printf("Post Increment \n");**

**printf("Isi variabel a: %d \n",a);**

**printf("Isi variabel a: %d \n",a++);**

**printf("Isi variabel a: %d \n",a);**

**printf("\n");**

**int b = 5;**

**printf("Pre Increment \n");**

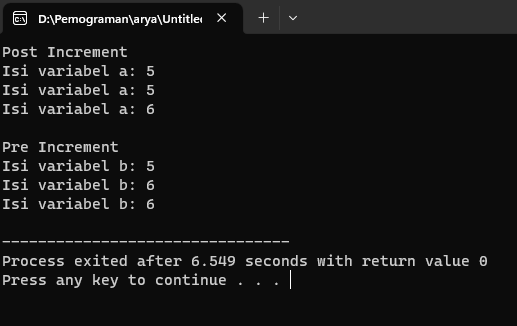
**printf("Isi variabel b: %d \n",b);**

**printf("Isi variabel b: %d \n",++b);**

**printf("Isi variabel b: %d \n",b);**

**return 0;**

**}**

Hasil ;

Perbedaannya ada di baris 8 dan 16, yakni ketika kita akses langsung pada saat operasi increment dilakukan.

Terlihat bahwa post-increment (a++), akan menampilkan hasilnya terlebih dahulu, baru nilai variabel a naik 1 angka, namun dengan pre-increment (++b), variabel b akan ditambahkan 1 angka, baru nilainya ditampilkan.

Begitu juga hal nya dengan operasi post-decrement dan pre-decrement:

**#include <stdio.h>**

**int main(void)**

**{**

**int a = 5;**

**printf("Post Decrement \n");**

**printf("Isi variabel a: %d \n",a);**

**printf("Isi variabel a: %d \n",a--);**

**printf("Isi variabel a: %d \n",a);**

**printf("\n");**

**int b = 5;**

**printf("Pre Decrement \n");**

**printf("Isi variabel b: %d \n",b);**

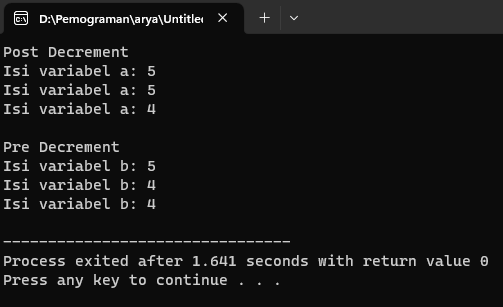
**printf("Isi variabel b: %d \n",--b);**

**printf("Isi variabel b: %d \n",b);**

**return 0;**

**}**

Hasil :



Terlihat bahwa post-decrement (a--), akan menampilkan hasilnya terlebih dahulu, baru nilai variabel a dikurangi 1 angka. Dengan pre-decrement (--b), variabel b langsung dikurangi 1 angka, lalu nilainya ditampilkan.

Operator increment dan decrement yang kita bahas disini nantinya banyak di terapkan dalam perulangan (looping). Selain itu kebiasaan banyak programmer lebih sering menggunakan post-increment (a++) dan post-decrement (a--). Nama bahasa pemrograman C++ juga terinspirasi dari post increment dari bahasa C, yakni penambahan dari bahasa C.

1. **Operator Logika Bahasa C**

Operator logika adalah operator yang digunakan untuk membuat kesimpulan logis dari 2 kondisi boolean: true atau false. Karena bahasa C tidak memiliki tipe data boolean, maka kita menggunakan angka 1 untuk pengganti true, dan angka 0 untuk pengganti false.

|  |  |  |  |
| --- | --- | --- | --- |
| Operator | Nama | Penjelasan | Contoh |
| && | And | Akan menghasilkan 1 jika kedua operand 1 | 1 && 0, hasilnya: 0 |
| || | Or | Akan menghasilkan 1 jika salah satu operand 1 | 1 || 0, hasilnya: 1 |
| ! | Not | Akan menghasilkan 1 jika operand 0 | !0, hasilnya: 1 |

Pada prakteknya, operator logika ini banyak dipakai untuk menggabungkan beberapa hasil operasi perbandingan (contohnya akan kita lihat sesaat lagi).

Catatan: operator OR menggunakan karakter pipe ” | “, bukan huruf L kecil. Karakter pipe ini bergabung dengan tombol “\” dan ditekan menggunakan tombol shift.

Contoh Kode Program Operator Logika Bahasa C

Dalam bentuk paling sederhana, operator logika bisa diproses untuk integer 1 dan 0. Dimana 1 mewakili true, dan 0 mewakili false. Berikut contoh kode programnya:

**#include <stdio.h>**

**int main(void)**

**{**

**int a;**

**a = 0 && 1;**

**printf("Isi variabel a: %d \n",a);**

**a = 1 && 1;**

**printf("Isi variabel a: %d \n",a);**

**a = 0 || 1;**

**printf("Isi variabel a: %d \n",a);**

**a = 0 || 0;**

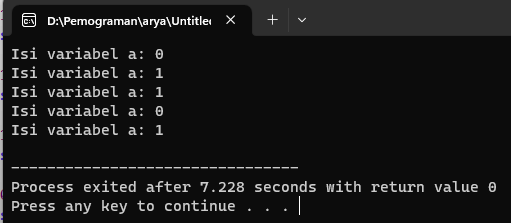
**printf("Isi variabel a: %d \n",a);**

**a = !0;**

**printf("Isi variabel a: %d \n",a);**

**return 0;**

**}**



Di sini saya membuat beberapa percobaan menggunakan operator logika. Rumus yang dipakai adalah sebagai berikut:

* Operator && hanya akan menghasilkan 1 jika kedua operand bernilai 1, selain itu hasilnya 0.
* Operator || hanya akan menghasilkan 0 jika kedua operand bernilai 0, selain itu hasilnya 1.
* Operator ! Akan membalikkan logika, !0 menjadi 1, !1 menjadi 0.

Kita juga bisa menggabungkan lebih dari satu operasi seperti contoh berikut:

**#include <stdio.h>**

**int main(void)**

**{**

**int a;**

**a = (0 && 1) || (1 || 0);**

**printf("Isi variabel a: %d \n",a);**

**a = !0 && (0 || 1);**

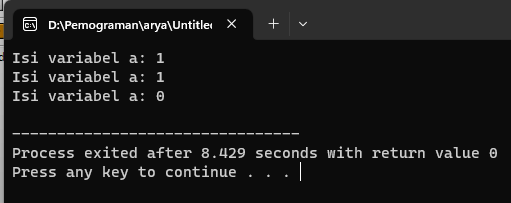
**printf("Isi variabel a: %d \n",a);**

**a = ((1 && 1) || (1 || 0)) && !1;**

**printf("Isi variabel a: %d \n",a);**

**return 0;**

**}**



Untuk operasi seperti ini, akan diproses dari kiri ke kanan, kecuali ditemukan tanda kurung maka itulah yang akan diproses terlebih dahulu.

* Di baris 7, operasi (0 && 1) || (1 || 0) akan diproses menjadi 0 || 1, hasilnya 1.
* Di baris 10, operasi !0 && (0 || 1) akan diproses menjadi 1 && 1, hasilnya 1.
* Di baris 13, operasi ((1 && 1) || (1 || 0)) && !1 akan diproses menjadi (1 || 1) && 0, kemudian menjadi 1 && 0, hasilnya 0.

Nilai boolean true (1) dan false (0) ini biasanya di dapat dari hasil operasi perbandingan. Inilah praktek yang sering dibuat untuk operator logika, berikut contohnya:

**#include <stdio.h>**

**int main(void)**

**{**

**int a;**

**a = (5 > 4) && (10 > 9);**

**printf("(5 > 4) && (10 > 9): %d \n",a);**

**a = (15 <= 15) && (15 < 15);**

**printf("(15 <= 15) && (15 < 15): %d \n",a);**

**a = ('a' == 'a') || ('a' == 'b');**

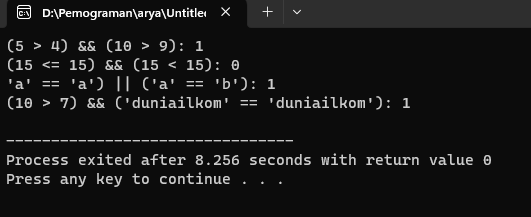
**printf("'a' == 'a') || ('a' == 'b'): %d \n",a);**

**a = (10 > 7) && ('duniailkom' == 'duniailkom');**

**printf("(10 > 7) && ('duniailkom' == 'duniailkom'): %d \n",a);**

**return 0;**

**}**



* Di baris 7, operasi (5 > 4) && (10 > 9) akan diproses menjadi 1 && 1, hasilnya 1.
* Di baris 10, operasi (15 <= 15) && (15 < 15) akan diproses menjadi 1 && 0, hasilnya 0.
* Di baris 13, operasi (‘a’== ‘a’) || (‘a’ == ‘b’) akan diproses menjadi 1 || 0, hasilnya 1.
* Di baris 16, operasi (10 > 7) && (‘duniailkom’ == ‘duniailkom’) akan diproses menjadi 1 && 1, hasilnya 1.

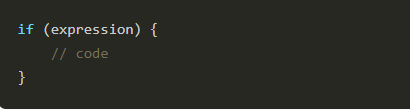
Sama seperti operasi perbandingan, operasi logika ini akan banyak dipakai pada percabangan kode program, misalnya untuk bisa login seseorang harus memiliki username DAN password yang sesuai. Jika salah satu saja tidak terpenuhi, maka tidak bisa login.

**PENGKONDISIAN**

Pengkondisian adalah `instruksi` pada pemrogaman untuk menjalankan suatu baris program sesuai dengan `kondisi tertentu`. Misal jika ada dua kondisi, A dan B dan ada juga baris program A dan B. Jika kondisi A terpenuhi maka program akan menjalankan baris program A, sedangkan baris program B tidak akan dijalankan begitu pula sebaliknya.

Dalam kehidupan sehari-hari sering kita jumpai keadaan atau kondisi seperti ini. Misal “jika saya terlambat melakukan absensi, maka gaji saya akan dipotong, jika saya tidak terlambat maka gaji saya tidak akan dipotong”. Anbi akan bahas penggunaan if statement pada bahasa pemrogaman C.

**If**

****

If statements digunakan untuk mengevaluasi ekspresi yang ada didalam tanda kurung (). Jika hasilnya TRUE maka kode yang ada didalam if akan dijalankan. Namun jika hasilnya False maka kode didalam if tidak akan dijalankan dan hanya dilewati begitu saja. Contoh program sederhana if statement adalah sebagai berikut:

**// Program untuk menampilkan bilangan positif**

**#include <stdio.h>**

**int main() {**

**int bilangan;**

**printf("Input Bilangan: "); scanf(" %d", &bilangan);**

**// nilainya TRUE jika bilangan lebih dari 0**

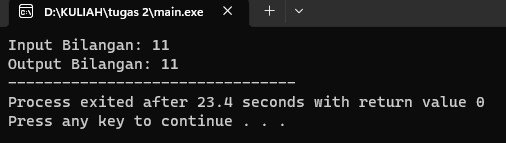
**if (bilangan > 0) {**

**printf("Output Bilangan: %d", bilangan);**

**}**

**return 0;**

**}**



**IF ELSE**

**if (expression) {**

**// jika TRUE, kode ini yang dijalankan**

**} else {**

**// jika FALSE, kode ini yang dijalankan**

**}**

Hampir sama seperti if diatas, bedanya kita memiliki kondisi kedua jika kondisi pertama tidak terpenuhi. Jika hasilnya True maka kode program yang ada didalam if akan dijalankan, sedangkan kode yang ada didalam else akan diabaikan. Namun jika hasilnya false maka kode program yang ada didalam if yang diabaikan, dan akan menjalankan kode yang ada didalam else. Contoh program sederhananya adalah sebagai berikut:

**// Program untuk menentukan bilangan ganjil genap**

**#include <stdio.h>**

**int main() {**

**int bilangan;**

**printf("Input Bilangan: "); scanf(" %d", &bilangan);**

**// True if the remainder is 0**

**if (bilangan%2 == 0) {**

**printf("%d bilangan genap", bilangan);**

**}**

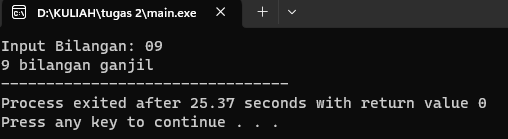
**else {**

**printf("%d bilangan ganjil", bilangan);**

**}**

**return 0;**

**}**

****

**if else (ladder)**

**if (expression1) {**

**// kode 1**

**} else if (expression2) {**

**// kode 2**

**} else if (expression3) {**

**// kode 3**

**} else {**

**// kode n**

**}**

Pengkondisian ini digunakan jika ada lebih dari dua kondisi yang akan dievaluasi. Konsepnya sama seperti pengkondisian sebelumnya, jika kondisi pertama hasilnya True, maka kode dari kondisi pertama akan dijalankan dan kondisi lainnya akan diabaikan. Jika kondisi kedua yang hasilnya True, maka kondisi lainnya yang akan diabaikan. Jika semua kondisi nilainya false, maka kode yang ada didalam else lah yang akan dijalankan. Contoh program sederhananya adalah sebagai berikut:

**// Program untuk menentukan equality**

**#include <stdio.h>**

**int main() {**

**int bilangan1, bilangan2;**

**printf("Bilangan 1: "); scanf(" %d", &bilangan1);**

**printf("Bilangan 2: "); scanf(" %d", &bilangan2);**

**// jika bilangan1 lebih kecil dari bilangan2**

**if(bilangan1 < bilangan2) {**

**printf("Hasilnya: %d < %d", bilangan1, bilangan2);**

**}**

**// jika bilangan1 lebih besar dari bilangan2**

**else if (bilangan1 > bilangan2) {**

**printf("Hasilnya: %d > %d", bilangan1, bilangan2);**

**}**

**// jika kedua bilangan sama besar**

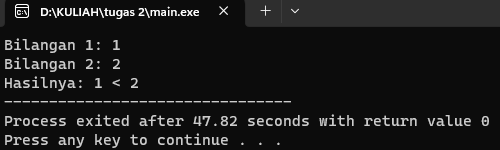
**else {**

**printf("Hasilnya: %d = %d", bilangan1, bilangan2);**

**}**

**return 0;**

**}**

****

**Ternary operator**

Ternary operator digunakan untuk mempersingkat penulisan if else saat if else hanya digunakan untuk memberikan nilai pada satu variabel berdasarkan kondisi tertentu. Syntax penulisannya adalah sebagai berikut.

* variabel=kondisi?jika\_benar:jika\_salah;
* return(kondisi?jika\_benar:jika\_salah);

Contoh kode programnya bisa kalian lihat di bawah ini.

**#include <iostream>**

**using namespace std;**

**int main(){**

**int angka;**

**string ganjil\_genap;**

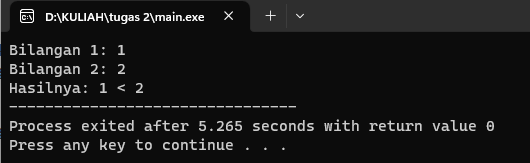
**cout << "angka:"; cin >> angka;**

**ganjil\_genap=(angka % 2 ==0)?"genap":"ganjil";**

**cout << angka << " adalah angka " << ganjil\_genap;**

**return 0;**

**}**



**#include <iostream>**

**using namespace std;**

**int main(){**

**int angka;**

**string ganjil\_genap;**

**cout << "angka:"; cin >> angka;**

**if(angka % 2 ==0)ganjil\_genap="genap";**

**else ganjil\_genap="ganjil";**

**cout << angka << " adalah angka " << ganjil\_genap;**

**return 0;**

**}**

Penulisan dengan menggunakan if jadi lebih panjang, tapi hasilnya sama saja dengan contoh sebelumnya. Ternary operator bisa dibilang hanya salah satu alternatif singkat dari penggunaan if dan else.

Return dengan Ternary Operator

kondisi ? ekspresi\_jika\_true : ekspresi\_jika\_false;

kondisi adalah ekspresi boolean yang dievaluasi. Jika kondisi ini benar (true), maka ekspresi pertama (ekspresi\_jika\_true) akan dieksekusi; jika kondisi ini salah (false), maka ekspresi kedua (ekspresi\_jika\_false) akan dieksekusi

**#include <stdio.h>**

**int main() {**

**int angka = 10;**

**char \*pesan = (angka > 5) ? "Lebih besar dari 5" : "Tidak lebih besar dari 5";**

**printf("%s\n", pesan);**

**return 0;**

**}**

Ternary bisa digunakan untuk mengembalikan nilai di dalam function. Nilai akan dikembalikan fungsi sesuai kondisi di dalam ternary operator.

**#include <iostream>**

**using namespace std;**

**string fungsiku(int angka){**

**return (angka % 2 ==0)?"genap":"ganjil";**

**}**

**int main(){**

**int angka;**

**string ganjil\_genap;**

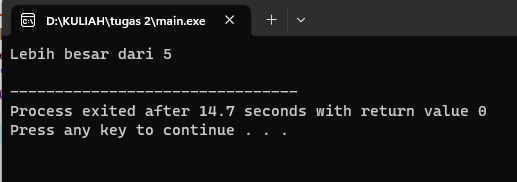
**ganjil\_genap=fungsiku(angka);**

**cout << "angka:"; cin >> angka;**

**cout << angka << " adalah angka " << ganjil\_genap;**

**return 0;**

**}**

****

Dalam contoh di atas, jika angka lebih besar dari 5, maka variabel pesan akan diisi dengan string "Lebih besar dari 5"; jika tidak, maka variabel pesan akan diisi dengan string "Tidak lebih besar dari 5". Pesan ini kemudian dicetak ke layar.

Operator ternary berguna untuk mengekspresikan kondisi sederhana dengan cara yang ringkas dan mudah dibaca dalam bahasa C.

PEMAKAIN IF DI DALAM IF

Didalam suatu pertanyaan *if* (atau *if-else*) bisa saja terdapat pernyataan *if* (atau *if-else*) yang lain. Bentuk seperti ini dinamakan sebagai nested if. Sebagai contoh ditunjukan pada programan berikut

#include <stdio.h>

main(){

float x,y;

printf("Masukan nilai x dan y.");

scanf("&f &f",x,y );

if (x > 0)

if(y > 0)

puts("x dan y positif");

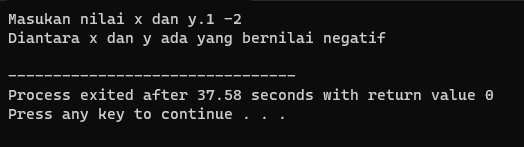
else

puts("Diantara x dan y ada yang bernilai negatif");

else

puts("Diantara x dan y ada yang bernilai negatif");

}



Tampak bahwa di dalam if-else terdapat if-else yang lain. Penulisan

Seperti

if ( x > 0)

if (y > 0)

puts("x dan y positif");

else

puts("Diantara x dan y ada yang bernilai negatif");

else

puts("Diantara x dan y ada yang bernilai negatif");

dengan kata-kunci if dan else yang berpasangan ditulis pada kolam yang sama dimaksudkan agar pernyataan mudah dibaca atau mengerti

if (x > 0)

puts(“ x dan y positif”);

else

puts(“Diantara x dan y ada yang bernilai negatif”);🡺bagian yang akan dijalanlan kalau x > 0

else

puts(“Diantara x dan y ada yang bernilai negatif”)🡺bagian yang akan di jalan kan kalau x ≤ 0

Untuk memahami tentang NESTED IF akan dibahas beberapa kemungkinan, dengan tujuan untuk menghindari salah penafsiran. Misalnya di tulis :

If (x > 0)

If(y > 0)

Puts(“x dan y positif”);

Else

Puts(“di antara x dan y ada yang bernilai negative”);

Pada contoh ini, else adalah pasasangan dari if (y > 0) . Seandainya x bernilai negataif, pengujian if tak akan menghasilkan keterangan apapun. Sedangkan pada penulisan

If ( x > y)

{

If (x > 0)

Puts(“ x dan y positf ”)

}

else

Puts(“salah satu diantara x atau y bernilai negatif ”)

Kalau x bernilai negative, pada layar akan mucul tulisan

**Diantara x dan y ada yang bernilai negative**

Sebab else adalah pasangan dari if ( x > 0 ). Kalau y bernilai negative , taka da keterangan yang muncul pada layar.

**Contoh if-else bertingkat**

**#include <stdio.h>**

**int main() {**

**int nilai = 75;**

**if (nilai >= 90) {**

**printf("Nilai: A\n");**

**} else if (nilai >= 80) {**

**printf("Nilai: B\n");**

**} else if (nilai >= 70) {**

**printf("Nilai: C\n");**

**} else if (nilai >= 60) {**

**printf("Nilai: D\n");**

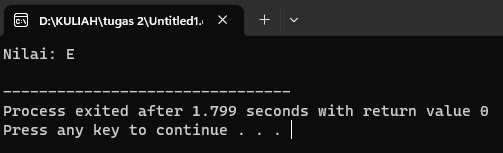
**} else {**

**printf("Nilai: E\n");**

**}**

**return 0;**

**}**



Dalam contoh di atas, kita menggunakan if-else bertingkat untuk menilai sebuah variabel nilai dan menentukan nilai huruf yang sesuai berdasarkan rentang nilai. Jika nilai lebih besar atau sama dengan 90, kita mencetak "A". Jika nilai di antara 80 dan 89, kita mencetak "B", dan seterusnya.

Ingatlah bahwa dalam if-else bertingkat, kondisi dievaluasi secara berurutan. Setelah salah satu kondisi yang benar ditemukan, blok pernyataan yang sesuai akan dieksekusi, dan eksekusi akan keluar dari struktur if-else. Oleh karena itu, jika nilai adalah 85, hanya "B" yang akan dicetak, bahkan jika kondisi "C" juga benar.

**Contoh lainnya**

**#include <stdio.h>**

**int main() {**

**int umur;**

**printf("Masukkan umur Anda: ");**

**scanf("%d", &umur);**

**if (umur < 0) {**

**printf("Masukkan umur yang valid\n");**

**} else if (umur < 18) {**

**printf("Anda adalah seorang anak\n");**

**} else if (umur < 60) {**

**printf("Anda adalah seorang dewasa\n");**

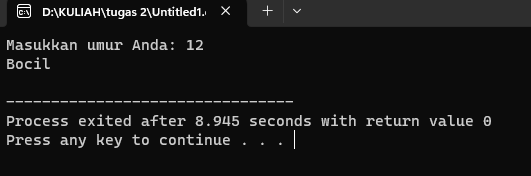
**} else {**

**printf("Anda adalah seorang senior\n");**

**}**

**return 0;**

**}**

****

SWITCHASE

Dalam bahasa pemrograman C, "switch" adalah salah satu jenis pernyataan kontrol yang digunakan untuk membuat percabangan dalam aliran program berdasarkan nilai ekspresi atau variabel tertentu. Pernyataan "switch" digunakan untuk menggantikan serangkaian pernyataan "if" yang memeriksa nilai yang sama berulang kali. Ini membuat kode lebih efisien dan mudah dibaca dalam situasi tertentu.

Berikut adalah sintaksis dasar pernyataan "switch" dalam bahasa C:

**switch (ekspresi) {**

**case nilai\_1:**

**// Blok kode yang akan dieksekusi jika ekspresi sama dengan nilai\_1**

**break;**

**case nilai\_2:**

**// Blok kode yang akan dieksekusi jika ekspresi sama dengan nilai\_2**

**break;**

**// Case-case lainnya**

**default:**

**// Blok kode yang akan dieksekusi jika ekspresi tidak cocok dengan nilai manapun**

**}**

Di bawah ini adalah contoh penggunaan pernyataan "switch" dalam bahasa C:

**#include <stdio.h>**

**int main() {**

**int pilihan;**

**printf("Pilih menu:\n");**

**printf("1. Mulai permainan\n");**

**printf("2. Keluar\n");**

**printf("Masukkan pilihan Anda: ");**

**scanf("%d", &pilihan);**

**switch (pilihan) {**

**case 1:**

**printf("Permainan dimulai!\n");**

**break;**

**case 2:**

**printf("Terima kasih telah bermain.\n");**

**break;**

**default:**

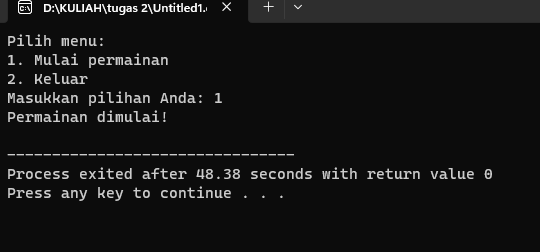
**printf("Pilihan tidak valid.\n");**

**break;**

**}**

**return 0;**

**}**



Dalam contoh ini, program meminta pengguna untuk memilih menu dengan memasukkan angka. Nilai yang dimasukkan oleh pengguna kemudian dibandingkan dengan nilai-nilai yang ada dalam blok "switch", dan blok kode yang sesuai dengan nilai yang cocok akan dieksekusi. Jika tidak ada yang cocok, maka blok "default" akan dieksekusi.

**DAFTAR PUSTAKA**

<http://www.sanabila.com/2015/06/pengertian-macam-macam-dan-penggunaan_23.html>

<https://www.anbidev.com/c-for-loop/>

<https://www.duniailkom.com/tutorial-belajar-c-jenis-jenis-operator-dalam-bahasa-c/>

<https://www.duniailkom.com/tutorial-belajar-c-jenis-jenis-operator-increment-dan-decrement-bahasa-c/>

<https://www.petanikode.com/c-operator/>

<https://www.anbidev.com/pengenalan-c/>