

MODUL PRAKTIKUM ALGORITMA DAN PEMROGRAMAN 1



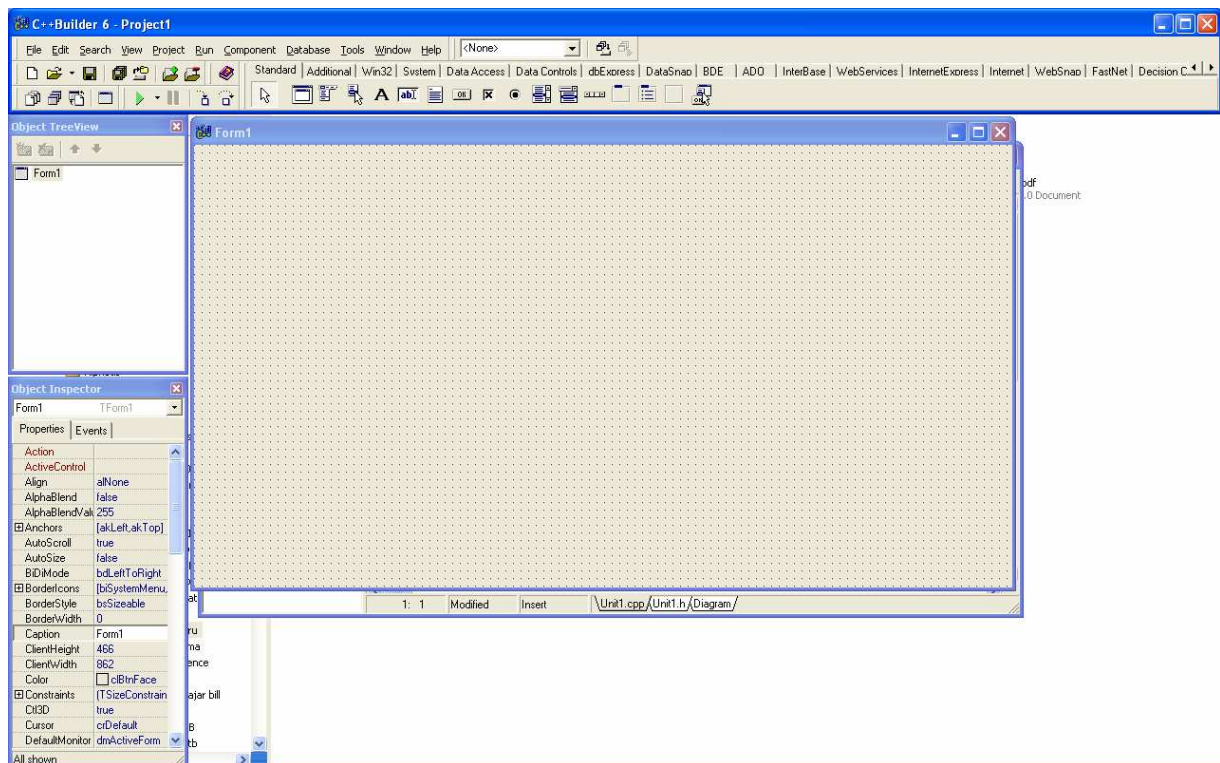
**Teks asli oleh Ferry Gustiawan,
Diperbaharui untuk Mata Kuliah
Praktikum Algoritma dan Pemrograman 1
Fakultas Ilmu Komputer Universitas Sriwijaya
Oleh Qurhanul Rizqie**

**Laboratorium Dasar
Fakultas Ilmu Komputer Universitas Sriwijaya
2010/2011**

1. Pendahuluan

Lingkungan Borland C++ Builder 6.0

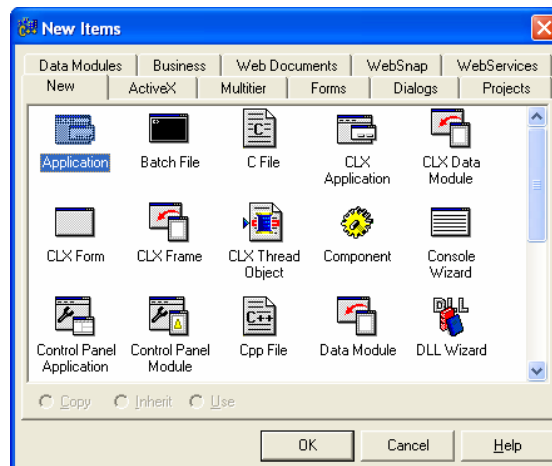
Turbo C++ 4.5 adalah tool yang dipakai untuk membuat code program dalam bahasa C ataupun C++. Berikut adalah jendela utama Borland C++ Builder 6.0.



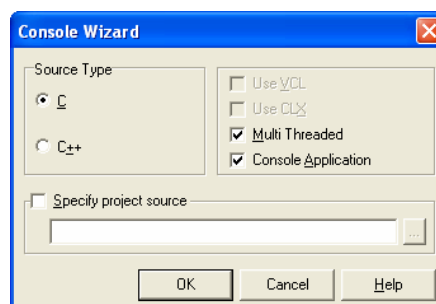
Create new, Open, Save, Save As File

Untuk memulai membuat kode program di Borland C++ Builder 6.0 langkah-langkahnya adalah sebagai berikut :

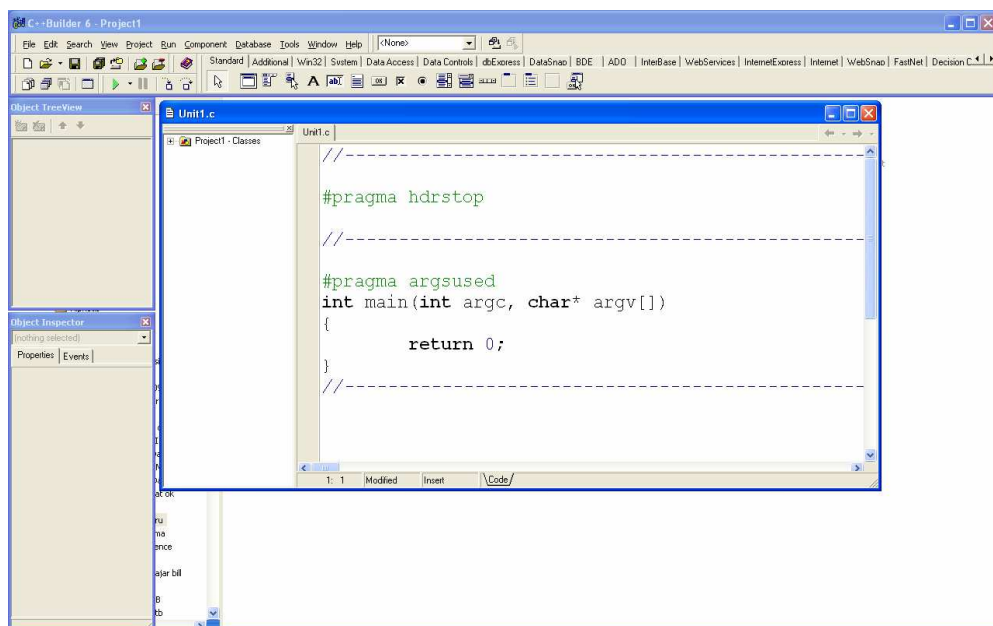
1. Buka Borland C++ Builder 6.0 dari menu program sehingga akan keluar jendela Borland C++ Builder 6.0 seperti gambar di atas.
2. Secara default Borland C++ Builder 6.0, disediakan untuk membuat program berbasis GUI, namun dalam mata kuliah Praktikum Algoritma dan Pemrograman 1, kita hanya akan mempelajari cara membuat program berbasis konsol. Agar Borland C++ Builder 6.0, dapat digunakan untuk membuat program berbasis konsol pilih menu **File > Close All** untuk menutup seluruh project, kemudian pilih **File > New > Other** , maka akan tampil, jendela sebagai berikut :



3. Kemudian, pilihlah opsi Console Wizard, kemudian pada jendela yang muncul berikutnya, pilihlah opsi bahasa C, lalu klik OK.



4. Tampilan jendela utama pada Borland C++ Builder 6.0, akan tampak sebagai berikut

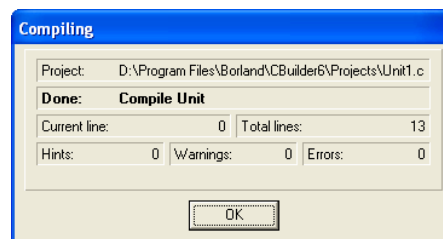


5. Ketikkan kode pada jendela code sesuai sintaks bahasa C, apabila telah selesai kode yang telah diketikkan dapat disimpan sebagai file bahasa C dengan ekstensi "*.c" dan projectnya akan disimpan dengan ekstensi "*.bpr"
6. Untuk menyimpan file pertama kali, pilih **File > Save Project As**, atur lokasi, dan beri nama sesuai keinginan.
7. Untuk membuka file atau kode program yang sudah pernah dibuat, pilih menu **File > Open**, pilih direktori yang diinginkan, kemudian pilih file dengan ekstensi "*.c" atau "*.bpr"

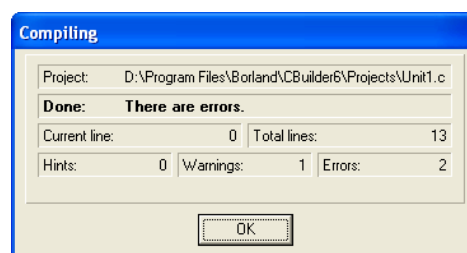
Compile Program, Pendeteksian Error dan Warning, Run Program

Setelah menuliskan kode program, maka berikutnya adalah compile program dengan tujuan untuk mendeteksi kesalahan-kesalahan dalam penulisan kode program. Adapun langkah-langkahnya adalah sebagai berikut :

1. Pilih menu **Project > Compile Unit**, atau kombinasi tombol **ALT+F9**, akan tampil jendela status compile seperti berikut :



Dari status di atas maka tidak ditemukan **error** atau **warning** pada program.



Dari status di atas dapat dilihat bahwa terdapat **error** pada program. Untuk melihat pesan error tersebut klik **OK** maka akan pesan error akan tampil di bawah jendela code.

2. Langkah berikutnya adalah membuat executable dari program, dengan cara pilih menu **Project > Create Project** atau kombinasi tombol **Ctrl + F9**
3. Setelah kode program di-create maka langkah berikutnya adalah menjalankannya, yaitu dengan memilih menu **Run > Run** atau dengan tombol **F9**.

Program Pertama “Hello world”

Langkah-langkah :

1. Buka aplikasi Borland C++ Builder 6.0.
2. Siapkan sebuah project console dengan cara yang telah diuraikan sebelumnya, kemudian ketikkan kode berikut :

```
Unit1.c |
#include<stdio.h>
#include<conio.h>

void main()
{
    printf("halo");
    getch();
}
```

3. Simpan program tersebut dengan nama hello.c.
4. Kompilasi program tersebut, lalu run. Akan tampil jendela hasil run program



5. Simpan program tersebut.

Catatan :

Baris program ‘#include<conio.h>’ dan ‘getch()’ adalah baris yang ditambahkan agar hasil program dapat terlihat di layar, tanpa kedua baris program inipun program sudah dapat berjalan, namun agar hasilnya dapat dilihat, harus dibuka melalui Command Prompt.

Contoh – contoh program selain program pertama ini tidak akan memasukkan kedua baris program tersebut, namun pada prakteknya, diharapkan mahasiswa untuk menambahkan sendiri baris program tersebut agar lebih mudah saat dilakukan evaluasi.

2. Tipe Dasar dan Bentukan

Tipe data dapat dikelompokkan menjadi dua macam :

1. Tipe Dasar
2. Tipe Bentukan

Tipe Dasar

Adalah tipe yang dapat langsung dipakai.

Tipe Dasar	Ukuran Memori (byte)	Jangkauan Nilai	Jumlah Digit Presisi
char	1	-128 hingga +127	-
int	2	-32768 hingga +32767	-
long	4	-2.147.438.648 hingga 2.147.438.647	-
float	4	3,4E-38 hingga 3,4E38	6-7
double	8	1.7E-308 hingga 1.7E308	15-16
long double	10	3.4E-4932 hingga 1.1E4932	19

Tipe data dapat diubah (type cast), misalkan:

```
float x = 3.345;
```

```
int p = int(x);
```

maka nilai p adalah 3 (terjadi truncating).

Tipe data yang berhubungan dengan bilangan bulat adalah char, int, long. Sedangkan lainnya berhubungan dengan bilangan pecahan.

Contoh :

```
#include <stdio.h>

void main()
{
    int a ;
    printf("%i",a);
}
```

Output dari program adalah :

19125

Darimana angka 19125 ?

Jika variable tidak diinisialisai, namun nilai keluarannya diminta, maka compiler dengan bijak akan menampilkan nilai acak yang nilainya tergantung dari jenis compilernya.

Bilangan logika/Boolean

Contoh :

```
#include <stdio.h>
#include "boolean.h"

void main()
{
    boolean bool;

    bool=true;      //atau bool=1
    if (bool)
    {
        printf("true\n");
    }
    else
    {
        printf("false\n");
    }
    if (!bool)
    {
        printf("salah\n");
    }
    else
    {
        printf("benar\n");
    }
}
```

Bilangan bulat (Integer)

Yang dimaksud dengan integer adalah bilangan bulat ...-3,-2,-1, 0,1,2,3...

Sebuah Integer N biasanya disajikan sebagai sebuah string di dalam memori komputer.

Beberapa format yang dipakai untuk mencetak nilai numerik integer :

Kode Format	Hasil yang dicetak
%i atau %d	Numerik integer bertanda
%u	Numerik integer tak bertanda (unsigned integer)
%li	Numerik Long integer bertanda
%lu	Numerik Unsigned Long Integer
%c	Karakter ASCII
%x	Hexadecimal Integer
%o	Octal Integer

Contoh 1 :

```
#include <stdio.h>

void main()
{
    printf("%i", 25);
}
```

Contoh 2 :

```
#include <stdio.h>

void main()
{
    printf("%d", 125);
}
```


Bilangan riil

Bilangan Riil adalah bilangan yang mengandung pecahan desimal, misalnya 3.65, 0.003 dan sebagainya.

Contoh 1 :

```
#include <stdio.h>

void main()
{
    printf("%f",25.0);
}
```

Contoh 2 :

```
#include <stdio.h>

void main()
{
    printf("%10.2f",25.0);
}
```

Contoh 3 :

```
#include <stdio.h>

void main()
{
    printf("%10.5f",25.0);
}
```

Karakter

Contoh 1 :

```
#include <stdio.h>

void main()
{
    putchar('A');
}
```

Contoh 2 :

```
#include <stdio.h>

void main()
{
    printf("%c", 'A');
}
```

String

String adalah gabungan dari karakter.

Contoh : " Belajar " → Literal String
 " B " → Karakter

Contoh 1 :

```
#include <stdio.h>

void main()
{
    puts("Palembang");
}
```

Contoh 2 :

```
#include <stdio.h>

void main()
{
    printf("Palembang\n");
    printf("P");
}
```

Contoh 3 :

```
#include <stdio.h>

void main()
{
    printf("%15s", "Palembang");
    printf("%s", "Bandung");
}
```

Contoh 4 :

```
#include <stdio.h>

void main()
{
    printf("%-15s", "Jakarta");
    printf("%s", "Samarinda");
}
```

Contoh 5 :

```
#include <stdio.h>

void main()
{
    printf("%15.4s", "Jakarta");
    printf("%s", "Samarinda");
}
```

Contoh 6 :

```
#include <stdio.h>

void main()
{
    printf("%-15.4s", "Jakarta");
    printf("%s", "Samarinda");
}
```

Enumerasi

Enumerasi adalah daftar nilai konstanta integer.

Contoh :

```
#include <stdio.h>

void main()
{
    enum hari
    {
        senin, Selasa, Rabu, Kamis, Jumat, Sabtu
    }
    hariku;

    enum
    {
        satu, dua, tiga
    }
    angka;

    enum
    {
        KEYWORD = 01, EXTERNAL = 03, STATIC = 04
    };

    typedef enum
    {
        merah, putih, kuning
    }
    warna;

    unsigned int flags;
    warna w = kuning;

    angka = tiga;
    printf("Angka %d \n ", angka);

    hariku = 0;
    printf ("Hari %d \n", hariku );

    printf("Masukkan sebuah angka [0..2]");
    scanf ("%d", &angka);
    printf ("Angka %d \n", angka);

    flags = EXTERNAL;
    printf ("flags %d \n ", flags);

    printf ("Warna = %d \n", w);
}
```

Tipe Bentuk

Tipe bentuk adalah tipe yang dibentuk dari tipe dasar atau dari tipe bentuk lain yang sudah didefinisikan, contohnya tipe struktur. Struktur terdiri dari data yang disebut field. Field-field tersebut digabungkan menjadi satu tujuan untuk kemudahan dalam operasi.

Bentuk umumnya :

```
typedef struct{ tipe nama_field1;
                tipe nama_field2;
                tipe nama_field3;
                ....
            }nama_variabel;
```

Contoh :

```
typedef struct { int tahun;
                int bulan;
                int tanggal;
            } data_tgl;

data_tgl tgl_lahir;

void main()
{
    tgl_lahir.tanggal=19;
    tgl_lahir.bulan=8;
    tgl_lahir.tahun=1978;

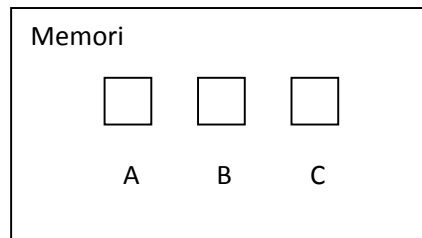
    printf("%i", tgl_lahir.tanggal);
    printf("%s", "/");
    printf("%i", tgl_lahir.bulan);
    printf("%s", "/");
    printf("%i", tgl_lahir.tahun);

}
```

3. Variabel

Deklarasi variable

Variable adalah suatu simbol atau lambang yang mempunyai nilai. Dalam pemrograman disebut juga area atau tempat di dalam memory komputer yang isinya bisa diubah-ubah.



Syarat penamaan variabel :

1. Tidak boleh sama dengan nama atau kata yang sudah disiapkan oleh komputer (reserved word) seperti *keyword* dan *functions*. Juga harus berbeda dengan nama label atau konstanta yang dibuat oleh programmer.
2. Maksimum 32 karakter, bila lebih 32 karakter maka karakter selebihnya tidak diperhatikan oleh komputer. Huruf besar dan huruf kecil berbeda.
3. Karakter pertama harus huruf atau karakter garis bawah (underscore) dan karakter berikutnya boleh huruf atau angka atau karakter garis bawah.
4. Tidak boleh ada spasi atau blank.

Contoh penamaan variable yang benar :

A, P5, Nilai, NILAI, nilai, HargaSatuan, Harga_Satuan, _Harga, dsb.

Bentuk umum deklarasi variable :

Nama_tipe Nama_variable

Contoh :

`int x; // Deklarasi x bertipe integer`

`char y, huruf, nim[10]; // Deklarasi variable bertipe char`

`float nilai; // Deklarasi variable bertipe float`

`double beta; // Deklarasi variable bertipe double`

`int array[5][4]; // Deklarasi array bertipe integer`

Contoh :

```
#include <stdio.h>

void main()
{
    int a, b, c;
    a = 3;
    b = 5;
    c = a * b;

    printf("%i", c);
}
```

Variabel lokal

Variable yang hanya bisa dipakai pada fungsi/prosedur itu sendiri.

Contoh :

```
#include <stdio.h>

void CETAK();

void main()
{
    int A,B,T;
    A=5;
    B=2;
    T=A+B;
    CETAK();
}

void CETAK()
{
    printf("%d", T);
}
```

Variabel A, B, dan T adalah variable lokal karena dideklarasikan di dalam prosedur dan hanya bisa dipakai oleh prosedur itu saja, yaitu prosedur **main()**. Pada program di atas akan terjadi error karena ada prosedur **CETAK()** yang menggunakan variable **T** dan variabel **T** tidak dikenal oleh prosedur **CETAK()**.

Variabel global

Variabel yang bisa dipakai oleh fungsi/prosedur lain karena dideklarasikan secara global.

Contoh :

```
#include <stdio.h>

void CETAK();
int T;

void main()
{
    int A,B;
    A=5;
    B=2;
    T=A+B;
    CETAK();
}

void CETAK()
{
    printf("%d",T);
}
```

Dari program di atas dapat dilihat bahwa variable **T** adalah variabel global karena dideklarasikan di luar prosedur sehingga bisa dipakai oleh prosedur lain, yaitu prosedur **CETAK()**. Sedangkan variable A dan B adalah variabel lokal.

4. Nilai, Konstanta, Ekspresi, Input Ouput dan Operator

Assignment

Pemberian nilai atas suatu variabel disebut *assignment*.

Contoh :

```
int A;
```

```
A = 5;
```

Untuk lebih jelas perhatikan program perkalian dua bilangan berikut.

```
#include <stdio.h>

void main()
{
    int a, b, c;
    a = 3;
    b = 5;
    c = a * b;

    printf("%i", c);
}
```

Terdapat 3 buah *assignment statement*.

Konstanta

Perhatikan kembali program untuk mengalikan dua bilangan di bawah ini :

```
#include <stdio.h>

void main()
{
    int a, b, c;
    a = 3;
    b = 5;
    c = a * b;

    printf("%i", c);
}
```

a, b, dan c adalah nama variabel sedangkan 3 dan 5 adalah nilai konstanta numerik.

Dalam bahasa C, konstanta dapat diberi nama. Lihat contoh berikut yang menghasilkan hasil yang sama dengan program di atas, dengan menggunakan **define**.

```

#include <stdio.h>
#define D 5
void main()
{
    int a,b,c;
    a = 3;
    b = D;
    c = a * b;
    printf("%i",c);
}

```

D adalah nama sebuah konstanta, nilai D tidak dapat dirubah.

Cara lain dengan menggunakan **const**.

```

#include <stdio.h>
void main()
{
    const int D = 5;
    int a, b, c;
    a = 3;
    b = D;
    c = a * b;
    printf("hasil = %i",c);
}

```

Ekspresi

Perhatikan assignment statement berikut ini :

X = A + B * C;

A + B * C disebut **arithmetic expression** atau pernyataan aritmatika dimana A, B, dan C disebut **operand** (yang dioperasikan) sedangkan + dan * disebut **arithmetic operator** (operator aritmatika).

Operator Aritmatika

Dalam bahasa C dan C++ operator aritmatika yang digunakan (disusun berdasarkan tingkat hirarki tertinggi sampai dengan yang terendah) adalah sebagai berikut :

Operator	Maksud dan tingkat hirarki
* dan / dan %	Kali dan Bagi dan Modulus

	* dan / dan % tingkt hirarkinya sama * dan / dan % tingkt hirarkinya lebih tinggi dari + dan -
+ dan -	Tambah dan Kurang + dan – tingkat hirarkinya sama

Pembagian

Contoh 1 :

```
int N = 25;
N = N / 2;
printf("%i", N);
```

akan tercetak : 12

Contoh 2 :

```
float F = 25.0;
F = F / 2;
printf("%f", F);
```

akan tercetak : 12.500000

Contoh 3:

```
int N;
float F;
N = 25;
F = N / 2;
printf ("%f", F);
```

Akan tercetak : 12.000000

Contoh 4 :

```
int N;
float F;
F = 25.5;
N = F / 2;
printf ("%i", N);
```

Akan tercetak : 12

Modulus

Operator % biasa disebut Modulus, hanya berlaku untuk bilangan integer, sehingga % disebut sisa pembagian bilangan integer.

Contoh :

```
int N = 25;
int K;
K = N % 2;
printf ("%i", K);
```

Akan tercetak : 1

Aritmatika dalam instruksi printf()

Contoh :

```
int A = 25;
printf ("%i", A);

printf ("%i", A+1); //Nilai A tidak berubah
printf ("%i", A+=1); //A ditambah 1 dulu baru dicetak
printf ("%i", A++); //A dicetak dulu kemudian ditambah 1
printf ("%i", ++A); //A ditambah 1 dulu baru dicetak
printf ("%i", A--); //A dicetak dulu kemudian dikurang 1
printf ("%i", --A); //A dikurang 1 dulu kemudian dicetak
```

Pangkat

Bahasa C, C++ tidak menyiapkan operator untuk pangkat tapi menyiapkan fungsi untuk perasi pangkat yang masuk dalam library file **math.h**.

Contoh :

```
#include <stdio.h>
#include <math.h>

void main()
{
    int N = 7;
    int K;
    K = pow (N,2);
    printf ("%i", K);
}
```

Akar

C atau C++ menyiapkan fungsi hanya untuk akar kwadrat yaitu **sqrt()**;

Contoh :

```
#include<stdio.h>
#include<math.h>
void main()
{
    float F = 27;
    printf("%f", sqrt(F));
}
```

Operator Relational

Relational merupakan ungkapan dengan menghubungkan atau merelasikan dua buah nilai atau lebih tepatnya membandingkan dua buah nilai seperti dicontohkan berikut ini :

- a. $5 > 3$ suatu pernyataan yang pasti bernilai TRUE
- b. $N > 30$ suatu pernyataan yang bisa bernilai TRUE dapat juga FALSE

Operator relational yang digunakan dalam bahasa C, C++ adalah :

Operator	Maksud
==	Sama dengan (Equal to)
>	Lebih besar dari (Greater than)
<	Lebih kecil dari (Less than)
>=	Lebih besar atau sama dengan (Greater than or equal to)
<=	Lebih kecil atau sama dengan (Less than or equal to)
!=	Tidak sama dengan (Not equal to)

Biasanya digunakan dalam oleh control statement **if**, **or** , atau **while**.

Dalam bahasa C, suatu ungkapan yang menyatakan kondisi, bila bernilai TRUE maka secara numerik (binary) ungkapan tersebut bernilai 1(satu), sedangkan yang bernilai FALSE bernilai 0(nol).

Contoh :

```
#include <stdio.h>

void main()
{
    printf("%i", 3 < 7);
}
```

Pembacaan Nilai

Instruksi untuk menginput data melalui keyboard pada program C dapat dibedakan menjadi dua macam :

1. Instruksi input tanpa format, yaitu :

gets (var) untuk karakter string

var = getche()

var = getchar()

var = getch()

instruksi ini hanya digunakan untuk menginput sebuah karakter dengan tipe **char**.

Contoh 1 :

```
#include <stdio.h>
void main()
{
    char S[10];
    printf("Masukkan string : ");
    gets(S);
    printf("%s", S);
}
```

Contoh 2 :

```
#include <stdio.h>
#include <conio.h>

void main()
{
    char C;
    printf("Masukkan sebuah karakter : ");
    C = getche();
    printf("\n%c", C);
}
```

2. Instruksi input menggunakan format.
Instruksinya **scanf()**.

Contoh 1 :

Input satu karakter

```
#include <stdio.h>
#include <conio.h>

void main()
{
    char C;
    printf("Masukkan sebuah karakter : ");
    scanf("%c", &C);
    printf("%c", C);
}
```

Contoh 2 :

Input dua karakter tanpa koma

```
#include <stdio.h>
#include <conio.h>

void main()
{
    char C1, C2;
    printf("Masukkan dua karakter : ");
    scanf("%c%c", &C1, &C2);
    printf("%c %c", C1, C2);
    printf("Selesai...");
}
```

Contoh 3 :

Input dua karakter disisipkan koma

```
#include <stdio.h>
#include <conio.h>

void main()
{
    char C1, C2;
    printf("Masukkan dua karakter : ");
    scanf("%c, %c", &C1, &C2);
    printf("%c %c", C1, C2);
    printf("Selesai...");
}
```

Contoh 4 :

Input dua karakter dipisah spasi

```
#include <stdio.h>
#include <conio.h>

void main()
{
    char C1, C2;
    printf("Masukkan dua karakter : ");
    scanf("%c %c", &C1, &C2);
    printf("%c %c", C1, C2);
    printf("Selesai...");

}
```

Contoh 5 :

Input berupa string

```
#include <stdio.h>

void main()
{
    char S[10];
    printf("Masukkan string : ");
    scanf("%s", S);
    printf("%s", S);

}
```

Contoh 6 :

Input berupa integer

```
#include <stdio.h>

void main()
{
    int a;
    printf("Masukkan sembarang nilai :");
    scanf("%d", &a);
    printf("Nilai yang diinput : %d \n", a);

}
```


5. Pernyataan Kondisi

IF Statement

Melakukan seleksi dapat dilakukan dengan control statement yaitu *if* dan *switch*. Berikut ini adalah contoh penggunaan if.

5.1.1 Satu Kasus

bentuk umum :

```
if (kondisi)
{
    Statement
}
```

Contoh :

```
#include <stdio.h>

void main()
{
    int a;

    printf("IF dengan Satu Kasus \n");
    printf("Ketikkan nilai integer : ");
    scanf ("%d", &a);
    if (a>=0)
    {
        printf("Nilai a positif %d \n", a);
    }
}
```

5.1.2 Lebih dari satu kasus

bentuk umum :

```
if (kondisi)
{
    Statement 1
}
else
{
    Statement 2
}
```

```
if (kondisi 1)
{
    Statement 1
}
else if (kondisi 2)
{
    Statement 2
}
else
{
    Statement 3
}
```

Contoh 1 :

```
#include <stdio.h>

void main()
{
    int a;

    printf("IF dengan Dua Kasus \n");
    printf("Ketikkan nilai integer : ");
    scanf ("%d", &a);
    if (a>=0)
    {
        printf("Nilai a positif %d \n", a);
    }
    else
    {
        printf("Nilai a negatif %d \n", a);
    }
}
```

Contoh 2 :

```
#include <stdio.h>

void main()
{
    int a;

    printf("IF dengan Tiga Kasus \n");
    printf("Ketikkan nilai integer : ");
    scanf ("%d", &a);
    if (a>0)
    {
        printf("Nilai a positif %d \n", a);
    }
    else if (a == 0)
    {
        printf("Nilai Nol %d \n", a);
    }
    else
    {
        printf("Nilai a negatif %d \n", a);
    }
}
```

Switch Case

Selain menggunakan *if* seleksi dapat juga menggunakan ***switch***.

Contoh :

```
#include <stdio.h>

void main()
{
    int n;

    printf("Ketik angka 1 - 3 :");
    scanf("%i", &n);

    switch (n)
    {
        case 1:
        {
            printf("Yang anda ketik angka 1");
            break;
        }
        case 2:
        {
            printf("Yang anda ketik angka 2");
            break;
        }
        case 3:
        {
            printf("Yang anda ketik angka 3");
            break;
        }
        default:
            printf("Salah ketik..");
    }
}
```

6. Pengulangan (Looping)

Pengulangan dalam bahasa C, C++ bisa menggunakan instruksi **for**, **while** dan **do while**.

Pengulangan berdasarkan banyaknya pengulangan

Contoh 1 :

```
#include <stdio.h>

void main()
{
    int I;
    for (I=1; I<=5; I=I+1)
    {
        printf("Palembang\n");
    }
    printf("\nBandung");
}
```

Contoh 2 :

```
#include <stdio.h>

void main()
{
    int I;
    for (I=1; I<=3; I++)
    {
        printf("Palembang\n");
    }
    printf("\nBandung");
}
```

Contoh 3 :

```
#include <stdio.h>

void main()
{
    int i;
    int N;

    printf("Baca N, print 1 s/d N\n");
    printf("N= ");
    scanf("%d", &N);
    for (i=1; i<=N; i++)
    {
        printf("%d \n", i);
    }
    printf("Selesai..\n");
}
```

Pengulangan berdasarkan kondisi berhenti

Contoh :

```
#include <stdio.h>

void main()
{
    int i;
    int N;

    printf("Nilai N > 0 = ");
    scanf("%d", &N);
    i=1;
    printf("print i dengan REPEAT : \n");
    do
    {
        printf("%d \n", i);
        i++;
    }
    while (i<=N);
}
```

Pengulangan berdasarkan kondisi pengulangan

Contoh 1 :

```

#include <stdio.h>

void main()
{
    int I;
    I = 1;
    while (I<=5)
    {
        printf("Bengkulu\n");
        I = I+1;
    }
    printf("\nLampung");
}

```

Pengulangan berdasarkan 2 aksi

Contoh :

```

#include <stdio.h>

void main()
{
    int N;
    int i;

    printf("Nilai N > 0 =");
    scanf("%d", &N);
    i = 1;
    printf("print i dengan ITERATE :");
    for (;;)
    {
        printf("%d \n", i);
        if (i == N)
            break;
        else
        {
            i++;
        }
    }
}

```

7. Fungsi

Fungsi adalah sejumlah instruksi yang dikelompokkan menjadi satu, berdiri sendiri, yang berfungsi untuk menyelesaikan suatu pekerjaan. Bahasa C minimal mempunyai satu buah fungsi yang disebut Fungsi **main()** yaitu fungsi induk/utama.

Contoh :

```
#include <stdio.h>

int HITUNG(int A, int B);

void main()
{
    int A,B,T;
    A=5; B=2; T=0;
    T=HITUNG(A,B);
    printf("\n %d", T);
}

int HITUNG(int A, int B)
{
    int T;
    A = A * 2;
    B = B * 2;
    T = A + B;
    return (T);
}
```

Dari contoh di atas fungsi **HITUNG** adalah fungsi yang dibuat sendiri. Fungsi **HITUNG** dipanggil di **main**. Fungsi yang kita buat sendiri bisa diletakkan diatas atau pun di bawah **main**. Bila fungsi yang kita buat diletakkan di atas **main** maka tidak perlu dideklarasikan lagi.

8. Prosedur

Prosedur sama dengan fungsi hanya saja tidak mengembalikan nilai return.

Contoh :

```
#include <stdio.h>

void tukar (int *a, int *b);

void main()
{
    int a, b;

    printf("ketikkan dua bilangan, pisahkan dengan return : \n");
    scanf("%d %d", &a, &b);
    printf("Tukar kedua bilangan... \n");
    tukar(&a, &b);
    printf("kedua bilangan setelah ditukar : a=%d, b=%d \n", a, b);
}

void tukar(int *a, int *b)
{
    int temp;
    temp=*a;
    *a=*b;
    *b=temp;
}
```


9. Array

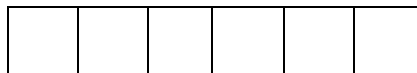
Array satu dimensi

Dalam bahasa pemrograman, array adalah variabel yang sejenis yang berderet sedemikian rupa sehingga alamatnya saling bersambung/kontigu atau dengan kata lain variabel berindeks.

Bentuk umum :

tipe_array nama_array

Ilustrasi array satu dimensi :



Array di atas mempunyai enam element.

Contoh 1 :

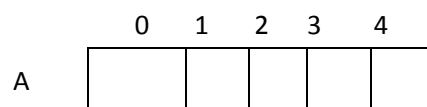
```
#include <stdio.h>

void main()
{
    int A[5];

    printf("\n%x", &A[0]);
    printf("\n%x", &A[1]);
    printf("\n%x", &A[2]);
    printf("\n%x", &A[3]);
    printf("\n%x", &A[4]);
}
```

int A[5] artinya menyiapkan array satu dimensi bertipe int dengan 5 elemen yang diberi index dari 0 sampai dengan 4. Program di atas akan mengalokasi 5 alamat elemen array dan mencetak alamat array dalam hexadesimal.

Ilustrasinya :



Contoh 2 :

Menyiapkan array numeric dengan nilainya.

```
#include <stdio.h>

void main()
{
    int A[5]={3, 5, 12, 7, 9};
    int i;

    for (i=0;i<=4;i++)
    {
        printf("%i", A[i]);
    }
}
```

Akan tercetak : 3 5 12 7 9

Contoh 3 :

```
#include <stdio.h>

void main()
{
    int A[5]={3, 5};
    int i;

    for (i=0;i<=4;i++)
    {
        printf("%i", A[i]);
    }
}
```

Akan tercetak : 3 5 0 0 0

Contoh 4 :

```
#include <stdio.h>

void main()
{
    char A[5]={'A', 'B', 'C'};
    int i;

    for (i=0;i<=4;i++)
    {
        printf("%c", A[i]);
    }
}
```

Akan tercetak : ABC

Array Multidimensi

Array multidimensi adalah array yang mempunyai lebih dari satu dimensi. Misal : A[3][5] artinya array tersebut mempunyai 3 baris 5 kolom.

Contoh 1 :

```
#include <stdio.h>

void main()
{
    int A[3][4] = { {1,2,3,4},
                    {2,3,4,5},
                    {3,4,5,6},
                    } ;

    int i;        // indeks baris
    int j;        // indeks kolom

    printf("Print Matriks 3 x 4 \n");
    for (i=0;i<3;i++)
    {
        for (j=0;j<4;j++)
        {
            printf("i,j=%d,%d A[i,j]=%d \n", i, j, A[i][j]);
        }
    }
}
```

Contoh 2 :

```

#include <stdio.h>

void main()
{
    //definisi tabel 3 dimensi
    int A[3][4][2];
    //definisi indeks
    int i;           //indeks baris
    int j;           //indeks kolom
    int k;           //indeks ???

    printf("Isi dan print tabel 3 dimensi, kemudian print \n");
    for (i=0;i<3;i++)
    {
        for (j=0;j<4;j++)
        {
            for (k=0;k<2;k++)
            {
                if ((i==j) && (j==k) && (i==k))
                {
                    A[i][j][k]=1;
                }
                else
                {
                    A[i][j][k]=0;
                }
                printf("i,j,k=%d,%d,%d A[i,j,k]=%d \n", i, j, k,
                    A[i][j][k]);
            }
        }
    }
}

```