



# DASAR PEMROGRAMAN

- **Basic Input/Output**
- **Operator**

# Standard Output (cout)

- `cout<<"Hello";`
- Akan menampilkan tulisan Hello ke layar

Hello

- `cout<<5+3;`
- Akan menampilkan angka 8 ke layar

8

- `cout<<110;`
- Akan menampilkan angka 110 ke layar

110

- `cout<<hello;`
- Akan menampilkan **isi variabel hello** ke layar
- `int hello = 9;`  
`cout<<hello;`

9

# Whitespace and basic formatting

**Whitespace** dapat berupa spasi, tab, ataupun terkadang newline. Kompiler C++ biasanya mengabaikan whitespace (dengan pengecualian tertentu)

Pernyataan-pernyataan berikut menghasilkan output yang sama :

- `cout << "Hello world!";`
- `cout                    <<                    "Hello world!";`
- `cout <<                    "Hello world!";`
- `cout            << "Hello world!";`

# Basic Formatting

- Newline tidak diperbolehkan untuk teks yang diapit tanda “”. Jika hendak membuat newline, gunakan ‘\n’ atau endl.

```
cout << "Hello  
world!" << endl;
```

Not allowed!

```
cout << "Hello \n world!" << endl;
```

Allowed

```
cout << "Hello " <<  
"world!" << endl;
```

Allowed

# Standard Output (cout)

```
cout << "Hello, " << "I am " << "a C++ statement";
```

- Statement tersebut akan menampilkan ke layar

```
Hello, I am a C++
```

```
cout << "Hello, I am " << age << " years old and my zipcode is " << zipcode;
```

- Bila diasumsikan isi variabel age adalah 25 dan zipcode adalah 90011 maka outputnya adalah :

```
Hello, I am 25 years old and my zipcode is 90011
```

# New Line (\n dan endl)

```
cout << "First sentence.\n ";  
cout << "Second sentence.\nThird sentence.";
```

- Output :

```
First sentence.  
Second sentence.  
Third sentence.
```

```
cout << "First sentence." << endl;  
cout << "Second sentence." << endl;
```

- Output :

```
First sentence.  
Second sentence.
```

# Standard Input (cin)

```
int umur;  
cin >> umur;
```

- Statement pertama mendeklarasikan suatu variabel bernama umur yang bertipe data integer.
- Statement kedua menanti input dari cin(keyboard) untuk menyimpan input tersebut ke dalam variabel umur.

# Operator

Relational operators	&&,   , !
Arithmetic operators	+, -, *, /, %
Comparison operators	<, >, ==, <=, >=, !=
Assignment operators	=, +=, -=, *=, /=, %=
Increment / Decrement operators	--, ++



# Operator Aritmatika

- Merupakan tanda untuk operasi aritmatika antara dua nilai

Opr	Fungsi	Contoh	Deskripsi
+	Penjumlahan	$a + b$	Penjumlahan a dan b
-	Pengurangan	$a - b$	Pengurangan a dan b
*	Perkalian	$a * b$	Perkalian a dan b
/	Pembagian	$a / b$	Pembagian a dan b
%	Modulus	$a \% b$	Sisa pembagian a dan b

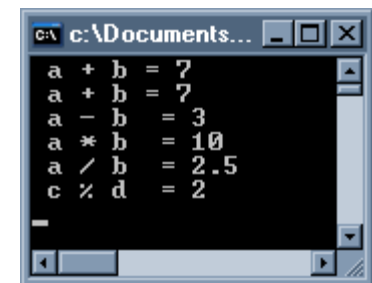
# Operator Aritmatika

```
#include <iostream>
#include <conio.h>
using namespace std;
void main()
{
    double a = 5.0;
    double b = 2.0;
    double tambah = a + b;
    double kurang = a - b;
    double kali    = a * b;
    double bagi    = a / b;
    int c = 10, d = 4;
    int sisa= c % d;
```

```
        cout<< " a + b = " << tambah <<endl;
//atau :
        cout<< " a + b = " << (a + b)<<endl;

        cout<< " a - b = " << kurang<<endl;
        cout<< " a * b = " << kali  <<endl;
        cout<< " a / b = " << bagi  <<endl;
        cout<< " c % d = " << sisa <<endl;

        _getch();
}
```



```
c:\Documents...
a + b = 7
a - b = 3
a * b = 10
a / b = 2.5
c % d = 2
```

# Assignment Operator

Opr	Fungsi	Contoh	Deskripsi
=	Sama dengan	<code>a = b</code>	a akan bernilai sama dengan b
+=	Penjumlahan & sama dengan	<code>a += b</code>	<code>a = a + b</code>
-=	Pengurangan & sama dengan	<code>a -= b</code>	<code>a = a - b</code>
*=	Perkalian & sama dengan	<code>a *= b</code>	<code>a = a * b</code>
/=	Pembagian & sama dengan	<code>a /= b</code>	<code>a = a / b</code>
%=	Modulus & sama dengan	<code>a %= b</code>	<code>a = a % b</code>

# Assignment Operator

```
// assignment operator
#include <iostream>
using namespace std;
int main ()
{
    int a, b;           // a:?, b:?
    a = 10;             // a:10, b:?
    b = 4;              // a:10, b:4
    a = b;              // a:4, b:4
    b = 7;              // a:4, b:7
    cout << "a:"; cout << a;
    cout << " b:"; cout << b;
    return 0;
}
```

# Assignment Operator

```
a = 2 + (b = 5);
```

Serupa dengan

```
b = 5; a = 2 + b;
```

```
a = b = c = 5;
```

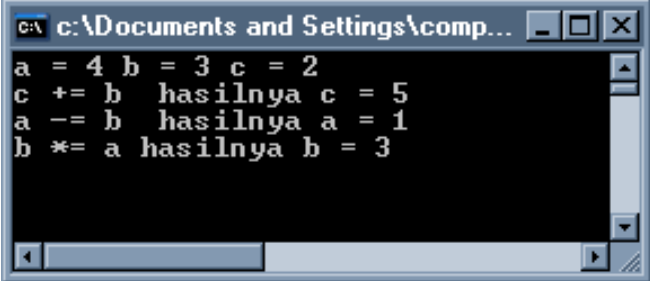
➔ Memberikan nilai 5 ke dalam tiga variabel : a, b dan c.

expression	is equivalent to
value += increase;	value = value + increase;
a -= 5;	a = a - 5;
a /= b;	a = a / b;
price *= units + 1;	price = price * (units + 1);

# Assignment Operator

```
#include <iostream>
#include <conio.h>
using namespace std;
void main()
{
    int a = 4;
    int b = 3;
    int c = 2;
    cout << "a = " << a << " b = " << b << " c = " << c<<endl;
    c += b;    // c = c + b
    cout<<"c += b  hasilnya c = " << c<<endl;
    a -= b;    // c = c - b
    cout<<"a -= b  hasilnya a = " << a<<endl;
    b *= a;    // b = b * a
    cout<<"b *= a  hasilnya b = " << b<<endl;

    _getch();
}
```



```
C:\Documents and Settings\comp...
a = 4 b = 3 c = 2
c += b  hasilnya c = 5
a -= b  hasilnya a = 1
b *= a  hasilnya b = 3
```

# Increment / Decrement Operator

Opr	Fungsi	Contoh	Deskripsi
++	Increment Operator	nilai++	Nilai ditambah satu setelah dioperasikan
		++nilai	Nilai ditambah satu sebelum nilai dioperasikan
--	Decrement Operator	nilai--	Nilai berkurang satu setelah dioperasikan
		--nilai	Nilai berkurang satu sebelum nilai dioperasikan

# Increment / Decrement Operator

- Penulisan operator sebagai akhiran ( $y = x++$  atau  $y = x--$ ) menunjukkan bahwa  $y$  memperoleh nilai  $x$  sebelum  $x$  berubah
- Jika operator ditulis sebagai awalan ( $y = ++x$  atau  $y = --x$ ), nilai  $x$  diberikan ke  $y$  setelah terjadinya perubahan



# Increment / Decrement Operator

```
#include <iostream>
#include <conio.h>
using namespace std;
void main()
{
    int x = 0;          int y = 0;
    cout<< "x dan y bernilai " << x << " dan " << y <<endl;
    x++;
    cout<< "x++ menghasilkan " << x <<endl;
    ++x;
    cout<<"++x menghasilkan " << x <<endl;

    cout<<"\nx dinolkan lagi!\n";
    x = 0;

    y = x++;
    cout<<"y = x++ (akhir) menghasilkan : \n";
    cout<<"x is " << x <<endl;
    cout<<"y is " << y <<endl;
```

```
cout<<"\nx dinolkan lagi!\n";
x = 0;
```

```
y = ++x;
cout<<"y = ++x (awalan) menghasilkan : \n";
cout<<"x is " << x <<endl;
cout<<"y is " << y <<endl;
```

```
_getch();
```

```
}
```

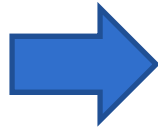
```
c:\Documents and Settings\compaq\My...
x dan y bernilai 0 dan 0
x++ menghasilkan 1
++x menghasilkan 2

x dinolkan lagi!
y = x++ (akhir) menghasilkan :
x is 1
y is 0

x dinolkan lagi!
y = ++x (awalan) menghasilkan :
x is 1
y is 1
-
```

# Increment / Decrement Operator

```
C=C+1;  
C+=1;  
C++;
```



mempunyai arti yang sama,  
yaitu menambahkan nilai c dengan 1

## Example 1

```
B=3;  
A=++B;  
// A contains 4, B contains 4
```

## Example 2

```
B=3;  
A=B++;  
// A contains 3, B contains 4
```

# Comparison Operator

- Bernilai **True** atau **False**

Opr	Fungsi	Contoh	Deskripsi
==	Equality operator	<b>a == b</b>	Bernilai true jika a sama dengan b
!=	Inequality operator	<b>a != b</b>	Bernilai true jika a tidak sama dengan b
<	Less than operator	<b>a &lt; b</b>	Bernilai true jika a lebih kecil dari b
>	Greater than operator	<b>a &gt; b</b>	Bernilai true jika a lebih besar b
<=	Less than or equal operator	<b>a &lt;= b</b>	Bernilai true jika a lebih kecil atau sama dengan b
>=	Greater than or equal operator	<b>a &gt;= b</b>	Bernilai true jika a lebih besar atau sama dengan b

# Operator Relasional

Opr	Fungsi	Contoh	Deskripsi
&&	Logical AND operator	<b>a &amp;&amp; b</b>	Bernilai true jika kedua nilai benar
	Logical OR operator	<b>a    b</b>	Bernilai true jika salah satu dari keduanya benar
!	Logical NOT operator	<b>! b</b>	Bernilai true jika nilai b adalah false

## Operator Not

Not True : False

Not False: True

## Operator AND

Arg 1	Arg 2	Arg 1 <b>AND</b> Arg 2
True	True	True
True	False	False
False	True	False
False	False	False

## Operator OR

Arg 1	Arg 2	Arg 1 <b>OR</b> Arg 2
True	True	True
True	False	True
False	True	True
False	False	False

# Ternary if-then-else operators

Expression ? Statement1 : statement2

- expression menyatakan besaran boolean yang akan diperiksa.
- Jika expression bernilai benar (**true**) maka **statement1** dieksekusi atau dijalankan,
- sedangkan statement2 dijalankan jika expression bernilai salah (**false**).
- Selain itu kedua statement harus memiliki tipe yang sama.
- Ternary operator dapat diistilahkan :  
If Expression ? **Then** statement1 **else** : statement2

## Expression ? Statement1 : statement2

```
// conditional operator
#include <iostream>
using namespace std;
int main ()
{
    int a,b,c;
    a=2; b=7;
    c = (a>b) ? a : b;
    cout << c;
    return 0;
}
```

- $7 == 5 ? 4 : 3$   
// akan mengembalikan nilai 3, karena 7 tidak sama dengan 5.
- $7 == 5 + 2 ? 4 : 3$   
// akan mengembalikan nilai 4, karena 7 bernilai sama dgn  $5 + 2$ .
- $5 > 3 ? a : b$   
// akan mengembalikan isi variabel a, karena 5 lebih besar dari 3.
- $a > b ? a : b$   
// akan mengembalikan nilai yang lebih tinggi, a atau b.

# Comma operator ( , )

- Digunakan untuk memisahkan dua atau lebih ekspresi.

- Contoh berikut ini :

```
a = (b=3, b+2);
```

Akan memberikan nilai 3 ke dalam variabel b, kemudian memberikan nilai (b+2) ke dalam variabel a, sehingga variabel a akan mempunyai nilai 5 dan variabel b akan mempunyai nilai 3.