OVERLOADING FUNCTION
Bab 4

4.1 PENGERTIAN

Overloading function dalah beberapa fungsi dapat memiliki argument berbeda tetapi namanya sama.

Contoh program :

```
# include <iostream.h>
Void tulis (char c)
{
Cout << “karakter : “ << c << endl ;
}
Void tulis (long x)
{
Cout << “bilangan bulat =” << x << endl ;
}
Void main ( )
{
Tulis ( ‘A’ ) ;
Tulis (1805) ;
}
```

Keterangan :

- walaupun 2 fungsi diatas mempunyai nama yang sama, tetapi compiler tahu fungsi manakah yang harus dipanggil.

C++ hanya dapat mengkompilasi overloading function jika argument fungsinya berbeda. Bila argument fungsinya sama tetapi tipe nilai kembaliannya berbeda, compiler akan melaporkan adanya kesalahan.

Contoh :

Fungsi : **double coba (int) ;** *dengan* Fungsi : **int coba (int) ;**

4.2 DEFAULT ARGUMEN

Contoh :

Void func (int x = 0, int y = 0) ;

Maksudnya adalah :

Contoh fungsi diatas dapat dipanggil dengan 3 (tiga) cara berbeda :
1. func ( ) *// nilai x dan y otomatis 0*
2. func (7) *// nilai x = 7 dan y = 0*
3. func (7, 33) *// nilai x = 7 dan y = 33*

Contoh deklarasi fungsi diatas akan terjadi kesalahan, jika variable x diberi nilai, sedangkan variable y tidak beri nilai.

Contoh :

Void func (**int x = 2, int y**) ;

4.3 AMBIGUITY PADA OVERLOADING FUNCTION

Beberapa hal yang menimbulkan ketidakjelasan pada program :
1. type conversion
Contoh :

```
Void f ( double nilai )
Main ( )
{
Int I = 12 ;
F ( I ) ;
}
```

2. reference parameter
Parameter yang memiliki perbedaan hanya pada argument, dimana fungsi yang satu menggunakan reference parameter dan yang lainnya menggunakan value parameter.
Contoh :

Void func **(int x , int y);** *dengan* void func func **(int x , int& y) ;**

3. default argumen

4.4 TATA CARA PENULISAN PROGRAM

1. Jangan menggunakan nama identifier yang terlalu singkat.
2. Identifier yang tidak melambangkan sesuatu dapat ditulis dengan satu huruf saja.
3. hindari pendeklarasian variable yang bertipe sama dalam satu baris, contoh : **int a, b, c** ;
4. Tambahkan satu spasi setelah tanda koma atau titik dua atau titik koma, jangan sebelumnya.

5. Jangan tambahkan spasi sesudah operator unary : ++ , - - , dsb.
   Contoh : **a ++**
6. Jangan tambahkan spasi setelah tanda kurung buka dan kurung tutup.
7. Jangan gunakan spasi dipenulisan index, contoh : **x[10]** *atau* **x [10]**
8. Hindari penulisan secara horizontal.
   Contoh :

```
Void main ( )
{
Tulis ( 'A' ) ; Tulis (1805) ;
}
```