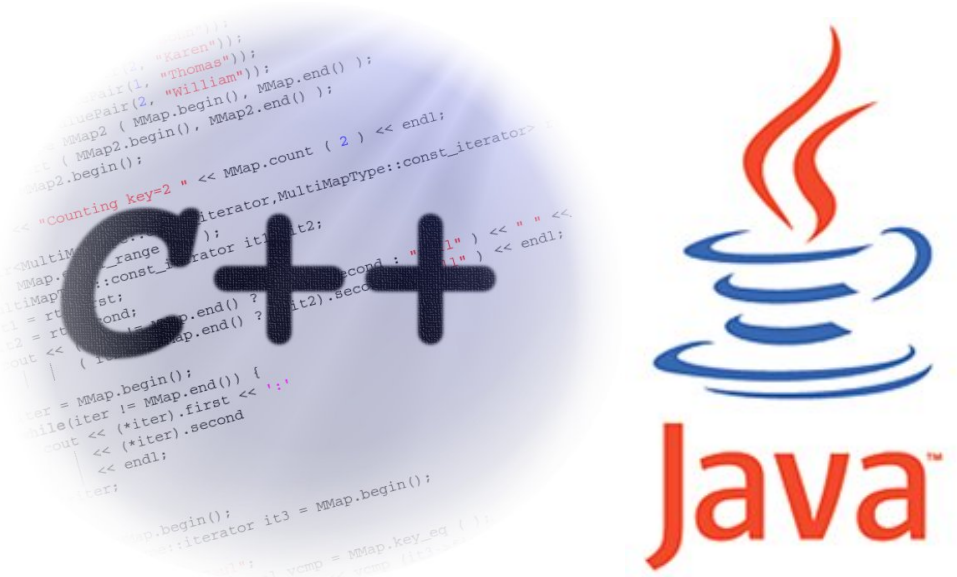
Modul Praktikum

PEMOGRAMAN

BERORIENTASI OBJEK



Disusun oleh

S. Indriani Lestariningati, M.T

2011

MODUL 1
PENGENALAN PEMOGRAMAN BERORIENTASI OBJEK

Cara terbaik untuk belajar bahasa pemograman adalah dengan cara langsung mempraktekkannya.

1. Menggunakan Bahasa C++

```
// program pertamaku
#include<iostream.h>
int main()
{
      cout <<"Selamat belajar C++";
}
```

Program diatas, misalnya dapat disimpan dengan nama *latih.cpp*.
Tekan tombol Alt+F9 untuk mengkompile program (*compile program*), jika status dinyatakan sukses, dan tidak ada pesan kesalahan (error) maka tekan Ctrl+F9 untuk menjalankan program tersebut (*run program*).

- `// program pertamaku`

  Merupakan sebuah baris komentar. Semua baris yang ditandai dengan dua buah tanda slash (//), akan diangkap sebagai baris komentar dan tidak akan berpengaruh pada hasil. Biasanya, baris komentar dipakai oleh programmer untuk memberikan penjelasan tentang program.
  Baris komentar dalam C++, selain ditandai dengan (//) juga dapat ditandai dengan (/*…*/)
  Perbedaan mendasar dari keduanya adalah
  // baris komentar
  /* blok komentar */

- `#include<iostream.h>`

  Pernyataan yang diawali dengan tanda (#) merupakan pernyataan untuk menyertakan pre-processor. Pernyataan ini bukan untuk

dieksekusi. #include<iostream.h> berarti memerintahkan compiler untuk menyertakan file header iostream.h. Dalam file header ini, terdapat beberapa fungsi standar yang dipakai dalam proses input dan output. (yaitu perintah cin dan cout)

- `int main( )`

  Baris ini menandai dimulainya compiler akan mengeksekusi program. Atau dengan kata lain, pernyataan main sebagai penanda program utama. Adalah suatu keharusan, dimana sebuah program yang ditulis pada bahasa C++ memiliki sebuah main.
  Main diikuti dengan sebuah tanda kurung ( ), karena main merupakan sebuah fungsi. Dalam bahasa C++ sebuah fungsi harus diikuti dengan tanda ( ), yang nantinya dapat berisi sebuah argument.

- `{ }`

  Isi dari sebuah fungsi harus diawali dengan kurung kurawal buka ({) dan diakhiri dengan kurung kurawal tutup (})

- `cout<<"Selamat belajar c++";`

  perintah ini merupakan hal yang akan dieksekusi oleh compiler dan merupakan perintah yang akan dikerjakan.
  Perlu diingat bahwa setiap pernyataan dalam C++ harus diakhiri dengan tanda semicolon (;) untuk memisahkan antara satu pernyataan dengan pernyataan yang lain.

- `return 0;`

  Pernyataan Return 0 akan menyebabkan fungsi main() menghentikan program dan mengembalikan nilai kepada main. Dalam hal ini, yang dikembalikan adalah nilai 0. Mengenai pengembalian akan dijelaskan nanti mengenai Fungsi dalam C++.

2. Menggunakan Bahasa Java

```
/*program pertamaku */
package latihan;
public class latihan
{
      public static void main (String[] args)
      {
            System.out.println("Selamat Belajar C++");
      }
}
```

- Package latihan;

  Package dalam bahasa Java merupakan sekumpulan dari berbagai kode yang terangkum dalam satu paket. Untuk memudahkan penulisan dan pembagian logika suatu program, satu paket terbagi menjadi beberapa berkas (file) dimana setiap file memiliki tugas atau tugas yang sangat khusus, misalnya satu file berfungsi untuk mendeklarasikan konstanta dan kelas, sementara file yang lain berisi implementasi kelas dan prosedurnya.
  Pada contoh sebelumnya, paket ini hanya berisi satu buah file yang isinya terdiri dari satu kelas dan satu metode.

- Komentar

  Komentar tidak akan diproses oleh kompiler tetapi berguna bagi programmer lain.
  Bahasa Java memiliki 3 jenis komentar:
  - /* text */ - compiler akan mengabaikan kata-kata antara /* dan */
  - /**documentation*/ - ini merupakan komentar yang dipergunakan khusus untuk dokumentasi. Alat bantu javadoc akan memproses komentar dokumentasi untuk membuat dokumentasi secara otomatis dari sumber program.
  - //text – kompiler akan mengabaikan segala sesuatu dari // hingga akhir baris

- `Public class latihan`

  Kelas merupakan bagian integral dari bahasa Java karena Java merupakan bahasa berorientasi objek. Setiap aplikasi harus terdiri dari satu kelas.
  Dalam hal ini kita definisikan kelas latihan sebagai kelas utama.

- `Public static void main(String[] args)`

  Metoda main( ) mirip dengan fungsi main pada bahasa C/C++ dimana fungsi ini merupakan pintu gerbang dimulanya suatu program. Metoda main dapat dipanggil dengan menyertakan variabel, baik hanya satu variabel, banyak variabel atau bahkan tidak sama sekali

- `System.out.println("Selamat belajar C++");`

  Perintah untuk menampilkan tulisan "Selamat belajar C++" pada layar komputer. Perintah diatas termasuk *compound names* atau nama campuran, yaitu yang merupakan nama biasa yang dihubungkan dengan titik.
  System.out.println artinya System menampung out dan out menampung println

MODUL 2
ELEMEN DASAR

1. Tipe Data

- Bahasa C++

Jenis Data	Deskripsi	Ukuran (bits)	Range
unsigned char	Karakter Unicode	8	0 s/d 255
char atau signed char		8	-128 s/d 127
unsigned int atau unsigned	Bilangan Bulat	16	0 s/d 65,535
int atau signed int atau signed		16	-32,768 s/d 32,767
unsigned long atau unsigned long int		32	0 s/d 4,294,967,295
long atau long int atau signed long atau signed long int		32	-2,147,483,648 s/d 2,147,483,647
Float	Bilangan Riil	32	3.4 E-38 s/d 3.4 E38
double	Bilangan Riil	64	1.7E-308 s/d 1.7E308
long double		80	3.4E-4932 s/d 1.1E4932

- Bahasa Java

Jenis Data	Deskripsi	Ukuran (bits)	Range
Boolean	Hanya bisa berisi benar atau salah	1	
Char	Karakter Unicode	8	0 s/d 65535
Byte	Bilangan Bulat	8	-128 s/d 127
Short	Bilangan Bulat	16	-32768 s/d 32767
Int	Bilangan Bulat	32	-2147483648 s/d 2147483647
Long	Bilangan Bulat	64	-9223372036854775808 s/d 9223372036854775807
Float	Bilangan Riil	32	1.40129846432481707 e-45 s/d 3.4028234663852886 e+38
Double	Bilangan Riil	64	4.94065645841246544 e-324 s/d 1.7976931348623157 e+308

Contoh program:

1. Mencetak string

```
#include<iostream.h>
void main()
{
       char S[8]="Bandung";
       cout<<S;
}
```

```
public class Contoh
{      public static void main (string[]args)
       { string S = “Jakarta”;
          System.out.println(S);
       }
}
```

2. Mencetak char

```
#include<iostream.h>
void main()
{
       char C=’A’;
       cout<<C;
}
```

```
public class Contoh
{      public static void main (string[]args)
       { char C=’A’;
          System.out.println(C);
       }
}
```

3. Mencetak bilangan bulat (int)

```
#include<iostream.h>
void main()
{
       int X=10;
       cout<<X;
}
```

```
public class Contoh
{      public static void main (string[]args)
       {  int X=10;
          System.out.println(X);
       }
}
```

2. Variabel

Variabel adalah sebuah tempat untuk menyimpan data. Deklarasi variabel adalah sebuah perintah agar komputer menyediakan variabel yang akan kita pakai. Satu-satunya cara memasukkan data kedalam variabel adalah dengan menggunakan assignment statement, atau pernyataan pemberian nilai

```
Variable = ekspresi;
```

Aturan penamaan variabel:

1. Nama variabel hanya boleh terdiri dari huruf, angka dan garis bawah (_)
2. Karakter pertama harus huruf

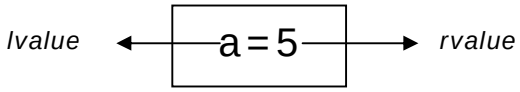
3. Huruf besar dan kecil dianggap berbeda (Case Sensitive)
4. Kata kunci (keyword) yang sudah ada didalam bahasa yang digunakan tidak boleh dipakai sebagai nama variabel.

3. Operator

Operator adalah sebuah simbol yang memerintahkan komputer untuk melakukan suatu operasi/ aksi terhadap satu atau lebih operand

- Operator Assign (=)

  Operator (=), akan memberikan nilai kedalam suatu variabel.



  Sebelah kiri tanda = dalam pernyataan diatas, dikenal dengan *lvalue* (left value) dan disebelah kanan tanda = dikenal dengan *rvalue* (right value).

  lvalue harus selalu berupa variabel, sedangkan *rvalue* dapat berupa variabel, nilai, konstanta, hasil operasi ataupun kombinasinya.

  Contoh program:

```
#include<iostream.h>
void main()
{
       int A, B, T;
       A = 10;
       B = 5;
       T = A+B;
       cout<<T;
}
```

```
public class Contoh
{   public static void main (string[]args)
    {
       int A, B, T;
       A = 10;
       B = 5;
       T= A+B;
       System.out.println(T);
    }
}
```

- Operator Aritmatika (+, -, *, /, %)

Operator	Keterangan
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
%	Modulus / Sisa bagi

Contoh:

a=11%3, maka variabel a akan terisi nilai 2 karena sisa hasil bagi 11 dan 3 adalah 2.

- Operator Penaikan dan Penurunan (++ dan --)

  Operator penaikan/ *increment* (++) akan menaikkan atau menambahkan 1 nilai variabel. Sedangkan operator penurunan/ *decrement* (--) akan menurunkan atau mengurangi 1 nilai variabel.

  Contoh:

  a++;

  a--;

  karakteristik dari operator ini adalah dapat dipakai diawal variabel (++a) atau diakhir variabel (a--). Hal ini akan berpengaruh pada hasilnya.

  Contoh:

  Pengaruh penempatan increment didepan:

```
#include <iostream.h>
void main()
{
	int r=2;
	int s;
	s=2 + ++r;
	cout<<s<<endl<<r;
}
```

  Pengaruh penempatan increment dibelakang:

```
#include <iostream.h>
void main()
{
	int r=2;
	int s;
	s=2 + r++;
	cout<<s<<endl<<r;
}
```

- Operator majemuk (+=, -=, *=, /=, %=, <<=, >>=, &=, |=)

  Dalam C++, operasi aritmatika dapat disederhanakan penulisannya dengan format penulisan operator majemuk.

  Contoh:

  a+=5 sama artinya dengan menuliskan a= a+5

  a*=5 sama artinya dengan menuliskan a=a*5

  a/=5 sama artinya dengan menuliskan a =a/5

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

void main()
{      int x = 2;
       clrscr();
       cout <<"x = "<< x <<endl;
       x+=3;
       cout <<"setelah x+=3, x menjadi " << x <<endl;
       x*=3;
       cout <<"setelah x*=3, x menjadi "<< x <<endl;
}
```

- Operator Relasional (==, !=, >, <, >=, <=)

  Yang dihasilkan dari operator ini bukan berupa sebuah nilai, namun berupa bilangan bool yaitu benar dan salah

Operator	Keterangan
==	Sama dengan
!=	Tidak sama dengan
>	Lebih besar dari
<	Kurang dari
>=	Lebih besar dari atau sama dengan
<=	Kurang dari atau sama dengan

  Contoh:

  (7==5) hasilnya adalah false

  (5 > 4) hasilnya adalah true

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

void main()
{      int nilai;

       clrscr();

       nilai = 3>2; //hasil ungkapan: benar
       cout << "nilai =   " << nilai<<endl;

       nilai = 2>3; //hasil ungkapan: salah
       cout << "nilai =   " <<nilai<<endl;
}
```

- Operator Logika (!, &&, | |)

  Operator logika juga digunakan untuk memberikan nilai atau kondisi true dan false. Biasanya operator logika dipakai untuk membandingkan dua buah kondisi. Misalnya:
  ((5==5) && (3>6) hasilnya akan bernilai false, karena (true && false)

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

void main()
{      int x = 200;

       clrscr();

       cout<<"(x>=1)&&(x<=50) ->"<<((x>=1)&&(x<=50))<<endl;
       cout<<"(x>=1)||(x<=50) ->"<<((x>=1)||(x<=50))<<endl;
}
```

- Operator kondisional (?)

  Format penulisan operator kondisional adalah:

  ```
  kondisi ? hasil1:hasil2
  ```

  Jika kondisi benar maka yang dijalankan adalah hasil1 dan jika kondisi salah maka akan dijalankan hasil2

Contoh:

7==5 ? 4 : 3 hasilnya adalah 3, karena 7 tidak sama dengan 5

5>3 ? a : b hasilnya adalah a, karena 5 lebih besar dari 3

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

void main()
{
	int a,b, minim;

	clrscr();

	a = 53;
	b = 6;
	minim=a<b?a:b;

	cout<<"Bilangan terkecil="<<minim<<endl;
}
```

MODUL 3
MENGINPUT DATA MELALUI KEYBOARD

1. Menginput Data Melalui Keyboard Pada Program C++

```
#include <iostream.h>
void main()
{
	int N;
	cin>>N;
	cout<<N;
}
```

a. Menginput sebuah karakter

- Menggunakan cin>>var;

```
#include <iostream.h>
void main()
{
	char C;
	cin>>C;
	cout<<C;
}
```

- Menggunakan var=getch();

```
#include <iostream.h>
void main()
{
	char C;
	C=getch();
	cout<<C;
}
```

- Menggunakan var=getche();

```
#include <iostream.h>
void main()
{
	char C;
	C=getche();
	cout<<N;
}
```

b. Menginput String

- Menggunakan cin>>var;

```
#include <iostream.h>
void main()
{
	char S[7];
	cin>>S;
	cout<<S;
}
```

- Menggunakan cin.getline(var, sizeof(var));

```
#include <iostream.h>
void main()
{
	char S[7];
	cin.getline(S,7);
	cout<<S;
}
```

c. Menginput nilai numerik

- Menggunakan cin>>var;

```
#include <iostream.h>
void main()
{
	int N;
	cin>>N;
	cout<<N;
}
```

- Menggunakan variabel lain

```
#include <iostream.h>
void main()
{
	Signed long int N;
	cin>>N;
	cout<<N;
}
```

2. Menginput Data Melalui Keyboard Pada Program Java

Scanner digunakan untuk meminta nilai yang dimasukkan oleh *user* (disebut juga input) didalam bahasa Java. Berikut adalah langkah-langkah kita menggunakan scanner

Kita harus membuat *scanner* terlebih dahulu sebelum kita meminta input dari layar.

Gunakan fungsi nextInt untuk meminta bilangan bulat (integer), nextFloat untuk meminta bilangan pecahan (float)

Scanner disimpan di library file/package yang bernama java.util.s=Scanner, sebelum kalian menggunakan Scanner, jangan lupa menyertakan (import) java.util.Scanner.

```
Scanner namaScanner = new Scanner(System.in);
```

Agar kalian tidak bingung, berikan nama scanner sesuai dengan aturan penamaan variabel

a. Menginput sebuah karakter

```
public class contoh
{
    public static void main(String[] args) throws Exception
       {      char c;
              System.out.println("Enter a character");
              c=(char)System.in.read();
              System.in.read();
              System.out.println("You entered=\n"+c);
       }
}
```

b. Menginput sebuah string

```
import java.util.Scanner;
public class contoh
{
       public static void main(String[] args)
       {      Scanner sc = new Scanner (System.in);
              System.out.print("Masukkan Nama Anda : ");
              String nama = sc.next ();
              System.out.println("Hello "+ nama);
       }
}
```

Sebuah objek scanner dapat mengurai input yang dimasukkan melalui keyboard atau dari sebuah file. Scanner memisahkan inputnya menjadi menjadi token terpisah (yang biasanya dipisahkan dengan spasi), dan kemudian mengembalikannya pada satu waktu. Scanner menyediakan metode untuk mengubah token menjadi nilai dari tipe yang berbeda. Kita dapat menggunakan utility scanner dalam dua cara, yaitu :

1. Untuk membaca dari keyboard, menggunakan perintah di bawah ini
   `Scanner input = new Scanner(System.in);`
2. Untuk membaca dari sebuah file, menggunakan perintah di bawah ini
   `Scanner input = new Scanner(new FileStream(“filename.txt”));`

c. Menginput sebuah bilangan

```
import java.util.Scanner;
public class contoh
{
	public static void main(String[] args)
	{
	Scanner sc = new Scanner(System.in);
	int luas;
	System.out.print("Masukkan Panjang : ");
	Integer panjang = sc.nextInt();
	System.out.print("Masukkan Lebar   : ");
	Integer lebar = sc.nextInt();

	luas = panjang * lebar;

	System.out.println("Luas Persegipanjang : "+ luas);
	}
}
```

d. Menginput string dan bilangan bulat

```
import java.util.Scanner;
public class contoh
{
	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);

		double Nilai_Akhir;

		System.out.print("Masukkan NIM Anda  : ");
		String NIM = sc.next();
		System.out.print("Masukkan Nama Anda : ");
		String Nama = sc.next();
		System.out.print("Masukkan UTS       : ");
		Integer UTS = sc.nextInt();
		System.out.print("Masukkan UAS       : ");
		Integer UAS = sc.nextInt();

		Nilai_Akhir = 0.4*UTS + 0.6*UAS;

		System.out.println("");
		System.out.println("");
		System.out.println("NIM Anda    : "+ NIM);
		System.out.println("Nama Anda   : "+ Nama);
		System.out.println("UTS         : "+ UTS);
		System.out.println("UAS         : "+ UAS);
		System.out.println("Nilai Akhir:"+ Nilai_Akhir);
		}
}
```

e. Menginput bilangan pecahan

```
import java.util.Scanner;
public class contoh
{
	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);
		double luas;
		System.out.print("Masukkan Panjang : ");
		Double panjang = sc.nextDouble();
		System.out.print("Masukkan Lebar   : ");
		Double lebar = sc.nextDouble();

		luas = panjang * lebar;

		System.out.println("Luas Persegipanjang : "+ luas);
			}
}
```

MODUL 4
SELEKSI KONDISIONAL DAN PERULANGAN

Dalam sebuah proses program, biasanya terdapat kode penyeleksian kondisi, kode pengulangan program, atau kode untuk pengambilan keputusan

1. SELEKSI KONDISIONAL

1.1. if

Kondisi adalah ekspresi yang akan dibandingkan. Jika kondisi bernilai benar, maka pernyataan akan dijalankan. Namun, jika kondisi bernilai salah, maka pernyataan akan diabaikan.

Format penulisannya:

```
if (kondisi)
{      pernyataan;
}
```

```
//C++
#include<iostream.h>
#include<conio.h>
void main()
{
       int a=5;
       clrscr();
       if (a<7)
       {
              cout<<“nilai a lebih kecil dari 7”<<endl;
       }
}
```

```
//Java
public class contoh
{
public static void main(String[] args)
       {
       int a=5;
       if(a<7)
              {
              System.out.println("nilai  a  lebih  kecil  dari  7");

              }
       }
}
```

1.2. if_else

Percabangan if_else dipakai untuk mengeksekusi salah satu dari 2 pernyataan dari syarat tertentu yang pada pada if yang dapat bernilai benar atau salah. Pernyataan1 akan dilakukan kalau ekspresi_boolean bernilai true. Kalau ekspresi_boolean bernilai false, maka Pernyataan2 akan dikerjakan. Sintaks dari if-else adalah sebagai berikut:

```
if(ekspresi_boolean)
{      Pernyataan1; }
else
{      Pernyataan2;}
```

```
//C++
#include<iostream.h>
void main()
{
       int a=3;
       int b=17;
       if (a<7)
              cout<<a<<" lebih kecil dari 7"<<endl;
       else
              cout<<a<<" lebih besar dari 7"<<endl;
       if (b<7)
              cout<<b<<" lebih kecil dari 7";
       else
              cout<<b<<" lebih besar dari 7";
}
```

```
//Java
public class contoh
{
public static void main(String[] args)
   {
       int a=3;
       int b=17;
       if (a<7)
       {
              System.out.println(a+ " lebih kecil dari 7");}
       else
       {
              System.out.println(a+ " lebih besar dari 7"); }
       if (b<7)
       {
              System.out.println(b+ " lebih kecil dari 7");}
       else
       {
              System.out.println(b+ " lebih besar dari 7"); }
    }
}
```

1.3. if_elseif (lebih dari dua kondisi)

Terkadang satu kondisi saja tidak cukup untuk menentukan satu syarat sehingga diperlukan dia atau lebih kondisi.

```
//Java
public class contoh
{
public static void main(String[] args)
   {
       char nilaiIndeks;
       double nilaiUTS, nilaiUAS, nilaiAkhir;
       // contoh data yang dimasukkan
       nilaiUTS = 75.0;
       nilaiUAS = 60.0;
       // menghitung nilai akhir menggunakan rumus di atas
       nilaiAkhir = (0.4 * nilaiUTS) + (0.6 * nilaiUAS);
       if (nilaiAkhir >= 80)
              {
                     nilaiIndeks = 'A';
              }
              else if (nilaiAkhir >= 70)
              {
                     nilaiIndeks = 'B';
              }
              else if (nilaiAkhir >= 50)
              {
                     nilaiIndeks = 'C';
              }
              else if (nilaiAkhir >= 30)
              {
                     nilaiIndeks = 'D';
              }
              else
              {    // (nilaiAkhir < 30)
                     nilaiIndeks = 'E';
              }
       System.out.println("Nilai Akhir\t: " +nilaiAkhir);
       System.out.println("Nilai Indeks\t: " +nilaiIndeks);
   }
}
```

1.4. switch_case

Logika menggunakan switch sama dengan menggunakan perintah if yang telah dijelaskan sebelumnya.

Sintaksnya adalah:

```
switch (option)
{
       case nilai 1:
              blok pernyataan 1
              break;
       case nilai 2:
              blok pernyataan 2
              break;
   -
   -
   default:
              blok pernyataan default
}
```

```
//C++
#include<iostream.h>
#include<conio.h>
void main()
{
       int angka;
       clrscr();
       cout<<"masukkan angka menurut keterangan dibawah ini:"<<endl;
       cout<<"-----------------------------------------------"<<endl;
       cout<<" 1 = jika anda berusia dibawah 17 tahun"<<endl;
       cout<<" 2 = jika anda berusia diatas 17 tahun"<<endl;
       cout<<"-----------------------------------------------"<<endl;
       cin>>angka;
       switch (angka)
       {
              case 1:
                     cout<<"Anda tidak diperkenankan menonton!";
                     break;
              case 2:
                     cout<<"Selamat menonton!";
                     break;
              default:
                     cout<<"tidak terdefinisi";
       }
}
```

```
//Java
import java.util.Scanner;
public class helloworld
{
public static void main(String[] args)
	{
	Scanner sc = new Scanner(System.in);
	System.out.println("Masukkan Angka menurut keterangan dibawah ini: ");
	System.out.println("-----------------------------------------------");
	System.out.println("1 : Jika usia anda dibawah 17 tahun: ");
	System.out.println("2 : Jika usia anda diatas 17 tahun: ");
	System.out.println("-----------------------------------------------");
	Integer angka = sc.nextInt();
	switch (angka)
		{
		case 1:
			System.out.println("Anda tidak diperkenankan menonton!");
			break;
		case 2:
			System.out.println("Selamat menonton!");
			break;
		default:
			System.out.println("tidak terdefinisi");

		}
	}
}
```

2. PERULANGAN

Sebuah atau beberapa pernyataan akan dijalankan secara berulang ulang, selama kondisi terpenuhi.

Loop statement digunakan agar kita tidak perlu menuliskan satu atau sekumpulan statement berulang-ulang. Dalam bahasa C++ dan Java setidaknya ada 3 macam perintah perulangan umum yang digunakan yaitu

1. Perintah while()
2. Perintah do while()
3. Perintah for()

2.1. while ()

Sintaksnya adalah:

```
while (kondisi) pernyataan;
```

Pernyataan akan dijalankan selama ekspresi bernilai true.

```
#include<iostream.h>
#include<conio.h>
void main()
{      int n;
       cout<<"Masukkan angka untuk mulai: ";
       cin>>n;
       while (n>0)
       {      cout<<n<<" , ";
              --n;
       } cout<<"selesai";
}
```

```
import java.util.Scanner;
public class contoh
{   public static void main(String[] args)
       {   int n;
           Scanner s = new Scanner(System.in);
           System.out.println("Masukkan nilai untuk memulai");
           n = s.nextInt();
           while(n>0)
              {      System.out.println(n);
                     --n;
              }
           System.out.println("Selesai");
       }
}
```

Algoritma untuk pengulangan diatas adalah sebagai berikut:

a. User menginputkan sebuah nilai ke variabel n.
b. Pernyataan while akan melakukan pengecekan apakah (n=0)?
c. Dalam kondisi ini, terdapat dua kemungkinan:
    1. True : lakukan pernyataan (langkah 3)
    2. False: lompati pernyataan (langkah 5)
d. Lakukan perintah:

```
cout<<n<<" , ";
--n;
```

e. Akhiri blok, kembali lagi ke langkah 2.
f. Lanjutkan program setelah blok while. Cetak SELESAI, dan akhiri program.

2.2. do while()

Sintaksnya adalah:

```
do pernyataan while (kondisi);
```

Konsep do_while mirip dengan while. Namun pernyataan akan dijalankan terlebih dahulu sebelum pengecekan kondisi.

```
#include<iostream.h>
#include<conio.h>
void main()
{      int n;
       cout<<"Masukkan angka untuk mulai: ";
       cin>>n;
       do
       {      cout<<n<<" , ";
              --n;
       }
       while (n>0);
  cout<<"selesai";
}
```

```
import java.util.Scanner;
public class contoh
{   public static void main(String[] args)
      {   int n;
          Scanner s = new Scanner(System.in);
          System.out.println("Masukkan nilai untuk memulai");
          n = s.nextInt();
          do
             {     System.out.println(n);
                   --n;
             }
          while(n>0);
          System.out.println("Selesai");
      }
}
```

2.3. for()

Pernyataan akan diulangi jika kondisi bernilai true (sama seperti while). Namun for() menetapkan inisialisasi dan counter berada dalam tanda kurung . Penjelasannya adalah sebagai berikut:

1. `inisialisasi;` akan dieksekusi. Biasanya merupakan variabel yang akan dipakai sebagai counter atau pencacah. Bagian ini akan dieksekusi hanya sekali.
2. `kondisi;` akan diperiksa, jika bernilai true maka perulangan akan dilanjutkan dan jika bernilai false maka perulangan akan dilewati.
3. `counter;` akan dieksekusi. Biasanya dapat terdiri dari sebuah instruksi atau blok instruksi yang berada diantara { dan }.

```
//C++
#include<iostream.h>
#include<conio.h>
void main()
{
      int i;
      clrscr();
      for(i=0;i<10;i++)
            {
                  cout<<"C++"<<endl;
            }
}
```

```
//Java
public class Contoh
{
public static void main(String[] args)
	{
		int n;
		for(n=0; n<5;n++)
		{
			System.out.println("Java");
		}
	}
}
```

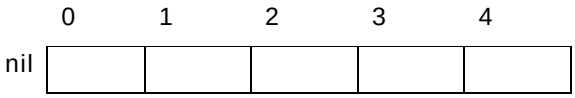
Contoh:

```
//C++
#include<iostream.h>
#include<conio.h>
void main()
{
	int i, x;
	clrscr();
	cout<<"Masukkan nilai x= ";
	cin>>x;
	for(i=x;i<5;i++)
		{
			cout<<i<<" , ";
		}
	cout<<"selesai";
}
```

MODUL 5
ARRAY

Pada program yang dibahas terdahulu, banyak menggunakan variabel tunggal, artinya sebuah variabel hanya digunakan untuk menyimpan satu nilai.

Array merupakan koleksi data dimana setiap elemen memakai nama yang sama dan bertipe sama, setiap elemen diakses dengan membedakan indeks array-nya.



Dari ilustrasi diatas, terlihat sebuah array satu dimensi, yang digambarkan dengan 5 buah kotak.

Berikut adalah cara mendeklarasikan sebuah array 1 dimensi didalam program:

- C++

```
tipe namaArray[kapasitas];
```

- Java

```
tipe namaArray[]=new tipe[kapasitas];
```

Array merupakan sekumpulan tempat penyimpanan data. Sebuah array dapat menyimpan lebih dari satu buah nilai (tergantung dari besarnya/kapasitas array). Meskipun begitu, nilai-nilai yang disimpan didalam sebuah array harus bertipe sama, yaitu sesuai dengan tipe dari array tersebut.

Contoh:

- C++

```
float A[3];
```

- Java

```
float A[]=new float[3];
```

Untuk mengakses (mengisi atau membaca) sebuah elemen dari array, kita hanya perlu menuliskan nama dari array tersebut, lalu diikuti dengan index yang dituju dan diapit dengan tanda kurung siku ([ ]).
Berikut adalah contoh potongan program yang mendeklarasikan, mengisi, lalu menampilkan sebuah array ke layar.

```
//C++
#include<iostream.h>
#include<conio.h>
void main()
{
       int bilangan[4];

       bilangan[0]=7;
       bilangan[1]=5;
       bilangan[2]=9;
       bilangan[3]=99;

       cout<<bilangan[0]<<endl;
       cout<<bilangan[1]<<endl;
       cout<<bilangan[2]<<endl;
       cout<<bilangan[3]<<endl;
}
```

```
//Java
public class Contoh
{
public static void main(String[] args)
   {   int bilangan []=new int[4];

       bilangan[0]=7;
       bilangan[1]=5;
       bilangan[2]=9;
       bilangan[3]=99;

       System.out.println(bilangan[0]);
       System.out.println(bilangan[1]);
       System.out.println(bilangan[2]);
       System.out.println(bilangan[3]);
    }
}
```

Dalam mengakses (mengisi atau mengambil nilai) sebuah array, kita harus dapat menggunakan perintah perulangan. Dengan demikian kita tidak akan repot karena harus mengakses elemen array tersebut satu persatu.
Contoh

```
#include<iostream.h>
#include<conio.h>
void main()
{
   int bilangan[4], index;

   bilangan[0]=7;
   bilangan[1]=5;
   bilangan[2]=9;
   bilangan[3]=99;

   for(index=0; index<4; index++)
       cout<<bilangan[index]<<endl;
}
```

```
//Java
public class Contoh{
public static void main(String[] args)
{  int bilangan[]=new int[4], index;

   bilangan[0]=7;
   bilangan[1]=5;
   bilangan[2]=9;
   bilangan[3]=99;

   for(index=0; index<4; index++)
       System.out.println(bilangan[index]);
    }
}
```

Array Multi Dimensi

Array multidimensi adalah array yang memiliki lebih dari satu index. Array multidimensi juga dapat dikatakan sebagai array dari array (sekumpulan array)

Sebagai contoh, sebuah matriks B berukuran 2x3 dapat dideklarasikan sebagai berikut:

```
tipe namaArray[kapasitas][kapasitas];
```

Contoh:

```
int n[2][3]={{2,4,1},{3,5,7}};
```

yang akan menempati lokasi memori dengan susunan sebagai berikut:

	0	1	2
0	2	4	1
1	3	5	7

Contoh program dengan dua dimensi

```
#include<iostream.h>
void main()
{
	int i,j;
	int matriks[2][3]={{2,4,1},{5,3,7}};
	for (i=0;i<2;i++)
	{for (j=0;j<3;j++)
		{
			cout<<matriks[i][j];
		}
	cout<<endl;
	}
}
```

Beberapa Operasi dengan Array

- Memperoleh bilangan terbesar
- Mencari suatu data pada array
- Mengurutkan data

1. Memperoleh Bilangan terbesar

Pada program berikut, mula-mula array diisi dengan bilangan acak. Kemudian program menampilkan isi array dan sekaligus memperoleh bilangan yang terbesar. Setelah menampilkan isi seluruh array, nilai terbesar ditampilkan

```
#include<iostream.h>
#include<conio.h>
#include<stdlib.h>
#include<time.h>
#include<math.h>

void main()
{
	clrscr();

	const MAKS = 10;
	int data[MAKS];
	int maks;

	//memperoleh data secara acak
	randomize();
	for(int i=0; i<MAKS; i++)
	data[i]=rand();

	//menampilan data dan mencari
	//data terbesar

	cout<<"DATA : "<<endl;
	cout<<data[0]<<endl;
	maks = data[0]; // isi dengan data pertama

	for(i=1 ; i<MAKS ; i++)
	{
		cout<<data[i]<<endl;

		if (data[i]>maks)
		{
			maks = data[i];
		}
	}
	cout<<"Data terbesar="<<maks<<endl;
}
```

2. Mencari Suatu Data

Program berikut memberikan gambaran cara untuk mencari suatu data didalam array. Mula-mula data yang akan dicari perlu dimasukkan dari keyboard. Kemudian, data ini dibandingkan dengan elemen-elemen array. Jika ada yang sama, program melaporkan posisi elemen array yang cocok dengan data yang dicari.

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

void main()
{
	int i, x, ketemu;

	clrscr();

	int data[]={5,100,20,31,77,88,99,20,55,1};

	cout<<"Data yang anda cari: ";
	cin>>x;

	ketemu = 0;
	for (i=0; i<sizeof(data)/sizeof(int); i++)
	{
		if (data[i] == x)
		{
			ketemu=!ketemu;	//ubah menjadi benar
			break;		//keluar dari for
		}
	}
	if (ketemu)
		cout<<"Data tersebut ada pada posisi ke”<<
		(i+1)<<endl;
		else
		cout<<"Data tersebut tidak ada!"<<endl;
}
```

3. Mengurutkan data

- Ada berbagai teknik untuk mengurutkan data, salah satu diantaranya adalah metoda *bubble sort*. (metoda ini terkenal karena kesederhanaannya).
- Pengurutan dilakukan dengan membandingkan setiap elemen dengan seluruh elemen yang terletak sesudah posisinya.

```
#include<iostream.h>
#include<conio.h>
#include<iomanip.h>
void main()
{
	int i, j, tmp, jumdata;
	clrscr();
	int data[]={5,100,20,31,77,88,99,20,55,1};
	jumdata=sizeof(data)/sizeof(int);

	//menampilkan data
	cout<<"data semula:"<<endl;
	for (i=0;i<jumdata;i++)
		cout<<setw(4)<<data[i];
		cout<<endl;

	//mengurutkan data
	for (i=0;i<jumdata-1;i++)
		for(j=i+1;j<jumdata;j++)
		if (data[i]>data[j])
		{
			tmp=data[i];
			data[i]=data[j];
			data[j]=tmp;
		}

	//menampilkan data
	cout<<"data setelah diurutkan:"<<endl;
	for (i=0;i<jumdata;i++)
		cout<<setw(4)<<data[i];
	cout<<endl;
}
```

MODUL 5
STRING

Menyiapkan array character sring satu dimensi lengkap dengan isinya
Disiapkan array satu dimensi bertipe char dan diisi dengan nilai awalan “ABC” dengan ilustrasi sbb:

A	B	C	\0	\0

Bila jumlah elemen yang disiapkan lebih banyak dari karakter yang diisi, maka sisa elemen selebihnya akan diisi dengan karakter NULL

Manipulasi String Pada Bahasa C++

```
#include<iostream.h>
void main()
{
    char C[5]="ABC";
    int i;

    for(i=0;i<=4;i++)
    {       cout<<C[i];
            cout<<"selesai";
    }
}
```

Terlihat ada dua spasi (“ “) antara “ABC” dan “selesai”. Spasi tersebut adalah hasil cetakan karakter “\0”

```
#include<iostream.h>
void main()
{
        char C[5]="ABC";
        cout<<C;
        cout<<"selesai";
}
```

Tercetak: ABCselesai
‘0’ tidak ikut tercetak

Bagaimana cara agar spasi dapat terbaca??

Fungsi anggota get() pada obyek cin (cin.get()) dapat dipakai untuk keperluan ini.

Contoh:

```
#include<iostream.h>
#include<conio.h>
void main()
{
char teks[13];
clrscr();
cout<<"Masukkan sebuah kata"<<endl;
cin.get(teks,13);
cout<<"yang anda masukkan:"<<teks<<endl;
}
```

Suatu masalah akan timbul kalau cin.get() digunakan dua kali, maka get() diganti dengan getline()

Contoh:

```
#include<iostream.h>
#include<conio.h>
void main()
{
        char nama[25];
        char alamat[35];
        clrscr();
        cout<<"Nama Anda      :";
        cin.getline(nama,sizeof(nama));
        cout<<"Alamat  :"<<alamat;
        cin.getline(alamat,sizeof(alamat));
        cout<<"NAMA    ="<<nama<<endl;
        cout<<"ALAMAT  ="<<alamat<<endl;
}
```

Membaca sejumlah baris

Fungsi anggota getline() juga bisa dipakai untuk membaca sejumlah baris hingga suatu karakter yang telah ditentukan dijumpai.

Hal ini dapat diperoleh dengan menyertakan argumen ketiga pada fungsi tersebut.

Argumen ini berupa karakter pengakhir.

Contoh:

```
cin.getline(teks, sizeof(teks), '$');
```

```
#include<iostream.h>
#include<conio.h>
void main()
{
char teks[128];
clrscr();
cout<<"Masukkan data (boleh beberapa baris)"<<endl;
cout<<"dan akhiri dengan $ dan Enter"<<endl;
cin.getline(teks, sizeof(teks),'$');
cout<<"yang anda ketikkan:"<<endl<<teks;
}
```

Mengisi array satu dimensi dengan nilai string

Contoh

1. Sudah ada array satu dimensi yang dibuat dengan char A[11], belum ada isinya. Susun algoritma untuk mengisi array A diatas dengan sebuah nilai string sehingga isinya menjadi sebagai berikut:

0	1	2	3	4	5	6	7	8	9	10
J	a	k	a	r	t	a	\0			

Jawab:
Cukup dengan satu instruksi :
Strcpy (A, “Jakarta”)

```
#include<iostream.h>
#include<string.h>
void main()
{
char A[10];
int i, n;
strcpy(A, "Jakarta");
n=strlen(A);
for(i=0;i<n;i++)
       {
              cout<<A[i];
       }
}
```

2. Sudah ada dua buah array satu dimensi masing-masing dibuat dengan char A[7] dan B[5]. Sudah ada isinya. Susun algoritma untuk membandingkan apakah isi kedua buah array tersebut sama. Bila sama, maka cetak “A==B” bila isi array A lebih kecil dari isi array B cetak perkataan “A<B”, selain cetak perkataan “A>B”

   Gunakan string compare : strcmp

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
void main()
{       char st[12];
        char cpp[]="string";
        clrscr();
        cout<<"Masukkan sembarang string "<<endl;
        cin.getline(st, sizeof(st));
        int hasil=strcmp(st, cpp);
        if (hasil==0)
                cout<<st<<"=="<<cpp<<endl;
        else if (hasil<0)
                cout<<st<<"<"<<cpp<<endl;
        else
        cout<<st<<">"<<cpp<<endl;
}
```

Mengetahui panjang string dengan strln()

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
void main()
{       char bunga[25]="mawar";
        char kosong[15]="";
        clrscr();
        cout<<strlen(bunga)<<endl;
        cout<<strlen(kosong)<<endl;
}
```

Menggabungkan string dengan strcat()

```
#include<iostream.h>
#include<conio.h>
#include<string.h>
void main()
{       char st[25]="satu dua";
        clrscr();
        cout<<"Isi st semula  : "<<st<<endl;
        strcat(st," tiga empat lima");
        cout<<"Isi st semula  : "<<st<<endl;
}
```

Perintah	Kegunaan
Strcpy	Menyalin isi string
Strln	Mengetahui panjang string
Strcat	Menggabungkan string
Strcmp	Membandingkan dua buah string

Manipulasi String Pada Bahasa Java

Untuk meminta input pada bahasa Java, kita tetap menggunakan Scanner. Namun fungsi yang digunakan adalah `nextLine()`. Berikut ini adalah fungsi-fungsi yang dapat digunakan untuk memanipulasi string dalam bahasa Java:

1. Fungsi `length()`: untuk memeriksa panjang dari sebuah string
   Contoh:

```
String str;
int panjang;

str = “halo”;
panjang = str.length();
System.out.printf(panjang);
```

   Hasil yang ditampilkan:

   4

2. Fungsi `concat()`: untuk menggabungkan 2 buah string
   Contoh:

```
String str1, str2;

str = “ha”;
str2= str1.concat(“lo”);
System.out.printf(str1, str2);
```

3. Fungsi `compareTo()` : untuk membandingkan apakah 2 buah string sama atau tidak. Jika sama, maka `compareTo()` akan menghasilkan nilai 0

   Contoh:

```
String str1, str2;

str =  “halo”;
str2= str2;
if(str1.compareTo(str2)==0)
       System.out.printf(“sama”);
else
       System.out.printf(“tidak sama”);
```

4. Fungsi `toCharArray()`: untuk mengubah tipe data String menjadi array dari char
   Contoh:

```
String str;

char arr[ ];
str=”halo”;
arr=str.toCharArray();
```

5. Fungsi `toString()`: untuk mengubah array dari char menjadi string
   Contoh:

```
String str1, str2;

str=“halo”;
arr=str.toCharArray();
str2=arr.toString();
```