

BAB 6

ARRAY & SORTING

A. ARRAY

Variabel Larik atau lebih dikenal dengan ARRAY adalah Tipe terstruktur yang terdiri dari sejumlah komponen-komponen yang mempunyai tipe yang sama. Suatu Array mempunyai jumlah komponen yang banyaknya tetap. Banyaknya komponen dalam suatu larik ditunjukkan oleh suatu indek untuk membedakan variabel yang satu dengan variabel yang lainnya.

Array adalah kumpulan dari nilai-nilai data bertipe sama dalam urutan tertentu yang menggunakan sebuah nama yang sama. Nilai-nilai data di suatu larik disebut dengan elemen-elemen larik.

Letak urutan dari suatu elemen larik ditunjukkan oleh suatu subscript atau suatu index.

Menurut dimensinya, array dapat dibedakan menjadi 2, yaitu:

1. Array satu dimensi

Sebelum digunakan, variabel array perlu dideklarasikan terlebih dahulu.

Cara mendeklarasikan variabel array sama seperti deklarasi variabel yang lainnya, hanya saja diikuti oleh suatu indek yang menunjukkan jumlah maksimum data yang disediakan.

Bentuk Umum pendeklarasian array:

Index array secara default dimulai dari 0

Deklarasi array :

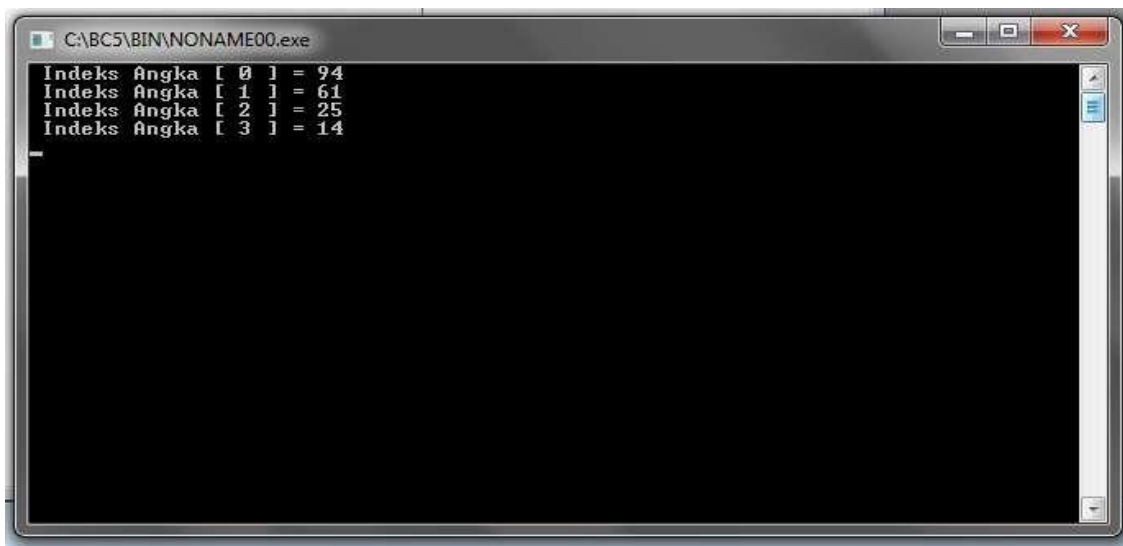
Tipe array nama array[ukuran]

Contoh-1 :

```
#include <iostream.h>
#include <conio.h>
main()
{
int angka[4]={94,61,25,14};

for(int a=0;a<4;a++)
{
cout<<" Indeks Angka [ "<<a<<" ] = "<<angka[a];
cout<<endl;
}
getch();
}
```

Hasilnya :



Gambar 6.1

Contoh-2 :

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

main()
{ char
mem[9][20]={ "Xiumin", "Lay", "Suho", "Baekhyun", "Chen", "Chanyeol", "D.o", "Kai", "S
ehun"};

cout<<" Nama Personel "<<endl<<endl;
for(int m=0;m<9;m++)
{
cout<<m<<". "<<mem[m]<<endl;
}

getch();
}
```

Hasilnya



Gambar 6.2

2. Array dua dimensi

Array dimensi dua tersusun dalam bentuk baris dan kolom, dimana indeks pertama menunjukkan baris dan indeks kedua menunjukkan kolom. Array dimensi dua dapat digunakan seperti pendataan penjualan, pendataan nilai dan lain sebagainya.

Array dua dimensi merupakan array yang terdiri dari m buah baris dan n buah buah kolom bentuknya dapat berupa matriks atau tabel.

Deklarasi array : Bentuk Umum pendeklarasian array

Tipe_array nama_array [baris][kolom]

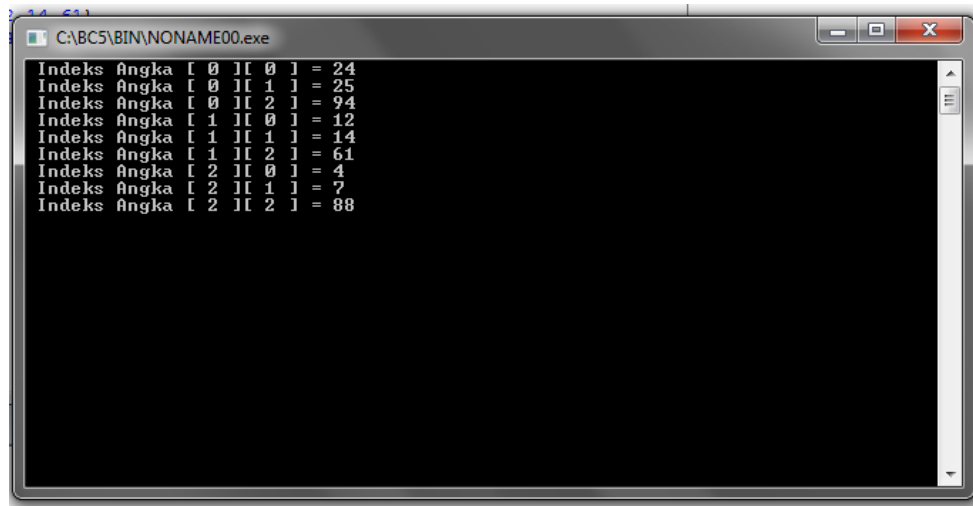
Contoh-1 :

```
#include <iostream.h>
#include <conio.h>
main()
{

int a,b;
int angka[3][3]={ { 24,25,94},
                  { 12,14,61},
                  { 4, 7,88}, };

for(a=0;a<3;a++)
{
for(b=0;b<3;b++)
{
cout<<" Indeks Angka [ "<<a<<" ][ "<<b<<" ] = "<<angka[a][b]<<endl;
}
}
getch();}
```

Hasilnya :



Gambar 6.3

Contoh-2 :

```
#include <iostream.h>
#include <conio.h>
main()
{
char nama[2][3][20]={ { "Park","Ooh","Kim"},
                        { "Chanyeol","Sehun","JongIn"}, };
cout<<endl<<nama[0][0]<<" "<<nama[1][0]<<endl;
cout<<nama[0][1]<<" "<<nama[1][1]<<endl;
cout<<nama[0][2]<<" "<<nama[1][2]<<endl;
getch();}
```

Hasilnya :



Gambar 6.4

B. SORTING

Sorting adalah suatu proses pengurutan data yang sebelumnya disusun secara acak atau tidak teratur menjadi urut dan teratur menurut suatu aturan tertentu.

Biasanya pengurutan terbagi menjadi dua yaitu :

1. Ascending (pengurutan dari karakter/angka kecil ke karakter/angka besar)
2. Descending (pengurutan dari karakter/angka besar ke karakter/angka kecil)

Contoh Ascending :

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

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

```
#include <iomanip.h>
```

```
int main(){
```

```
int A[5];
```

```
int j,k,i,temp;
```

```
int jmax,u=4;
```

```
cout<<"Masukkan nilai pada elemen array : "<<endl;
```

```
for(i=0;i<5;i++)
```

```
{
```

```
cout<<" A [ "<<i<<" ] = ";cin>>A[i];
```

```
}
```

```
for(j=0;j<5;j++)
```

```
{ jmax=0;
```

```
for(k=1;k<=u;k++)
```

```
if (A[k]>A[jmax])
```

```
jmax=k;
```

```
temp=A[u];
```

```

A[u]=A[jmax];
A[jmax]=temp;
u--;
}

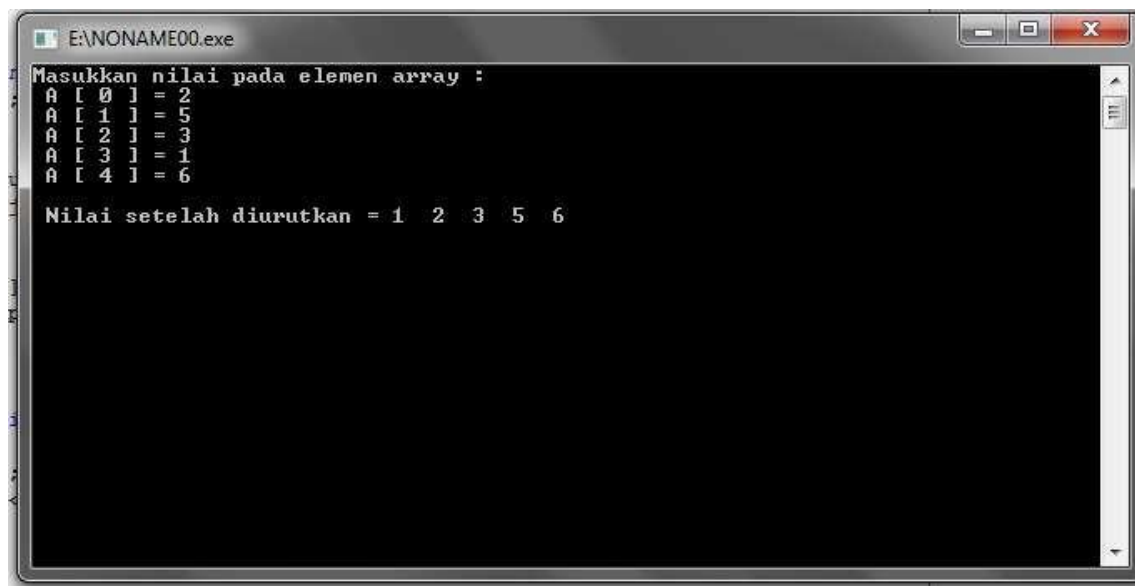
cout<<"\n Nilai setelah diurutkan = ";

for(i=0;i<5;i++)
cout<<A[i]<<" ";

getch();
}

```

Hasilnya :



Gambar 6.5

Contoh Descending :

```

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

int main(){

```

```

int A[5];
int j,k,i,temp;
int jmin,u=4;

cout<<"Masukkan nilai pada elemen array : "<<endl;
for(i=0;i<5;i++)
{
cout<<" A [ "<<i<<" ] = ";cin>>A[i];
}

for(j=0;j<5;j++)
{ jmin=0;
for(k=1;k<=u;k++)
if (A[k]<A[jmin])
jmin=k;
temp=A[u];
A[u]=A[jmin];
A[jmin]=temp;
u--;
}

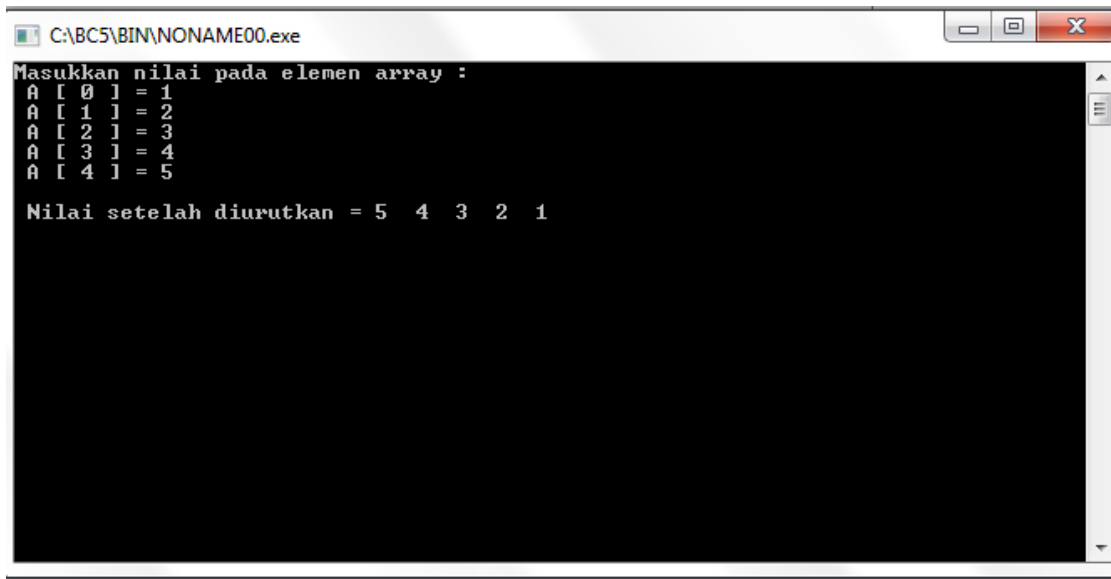
cout<<"\n Nilai setelah diurutkan = ";

for(i=0;i<5;i++)
cout<<A[i]<<" ";

getch();
}

```


Hasilnya :



```
C:\BC5\BIN\NONAME00.exe
Masukkan nilai pada elemen array :
A [ 0 ] = 1
A [ 1 ] = 2
A [ 2 ] = 3
A [ 3 ] = 4
A [ 4 ] = 5

Nilai setelah diurutkan = 5 4 3 2 1
```

Gambar 6.6

Jenis-jenis Sorting

a. Selection Sort

Teknik pengurutan dengan cara pemilihan elemen atau proses kerja dengan memilih elemen data terkecil untuk kemudian dibandingkan & ditukarkan dengan elemen pada data awal, dan seterusnya sampai seluruh elemen shg akan menghasilkan pola data yg telah *disort*.

Prinsip kerja dari teknik ini adalah :

1. Pengecekan dimulai data ke-1 sampai dengan data ke-n
2. Tentukan bilangan dengan Index terkecil dari data bilangan tersebut
3. Tukar bilangan dengan Index terkecil tersebut dengan bilangan pertama ($I = 1$) dari data bilangan tersebut
4. Lakukan langkah 2 dan 3 untuk bilangan berikutnya ($I = I + 1$) sampai didapatkan urutan yg optimal.

Contoh :

```
#include <iostream.h>
#include <conio.h>
#include <stdio.h>
#include <iomanip.h>
main()
{

int a[10],i,tmp,minindex,j;

cout<<"Selection Sort"<<endl<<endl;
cout<<" Masukkan data A :\n";
for(i=0;i<5;i++)
{
cout<<" A [ "<<i<<" ] = ";cin>>a[i];
}
cout<<"\n Data yang akan disorting : ";
for(i=0;i<5;i++)
{
cout<<setw(4)<<a[i];
}
for(i=0;i<5-1;i++) //perulangan iterasi
{
minindex=i;
for(j=i+1;j<5;j++) //perulangan membandingkan data
{
if(a[minindex]>a[j])
{
minindex=j;
}
}
tmp=a[i];
```

```

a[i]=a[minindex];
a[minindex]=tmp;
}
cout<<"\n\n Hasil Sorting      : ";
for(i=0;i<5;i++)
{
cout<<setw(4)<<a[i];
}
getch();
}

```

Hasilnya :



Gambar 6.7

b. Bubble Sort

Prinsip Kerja dari *Bubble Sort* adalah :

1. Pengecekan mulai dari data ke-1 sampai data ke-n
2. Bandingkan data ke-n dengan data sebelumnya (n-1)
3. Jika lebih kecil maka pindahkan bilangan tersebut dengan bilangan yg ada didepannya (sebelumnya) satu persatu (n-1,n-2,n-3,...dst)
4. Jika lebih besar maka tidak terjadi pemindahan
5. Ulangi langkah 2 dan 3 s/d sort optimal.

Contoh :

```
#include <iostream>
```

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

```
int data[10],data2[10];
```

```
int d;
```

```
void tukar(int a,int b)
```

```
{
```

```
int t; t=data[b];
```

```
data[b]=data[a];
```

```
data[a]=t;
```

```
}
```

```
void Input()
```

```
{
```

```
cout<<"Masukkan jumlah data = ";cin>>d;
```

```
cout<<endl;
```

```
for(int i=0;i<d;i++)
```

```
{
```

```
cout<<"Masukkan data ke - "<<(i+1)<<" = ";cin>>data[i];
```

```
data2[i]=data[i];
```

```
}
```

```
cout<<endl;
```

```
}
```

```
void Tampil()
```

```
{
```

```
for(int i=0;i<d;i++)
```

```
{
```

```
cout<<data[i]<<" ";
```

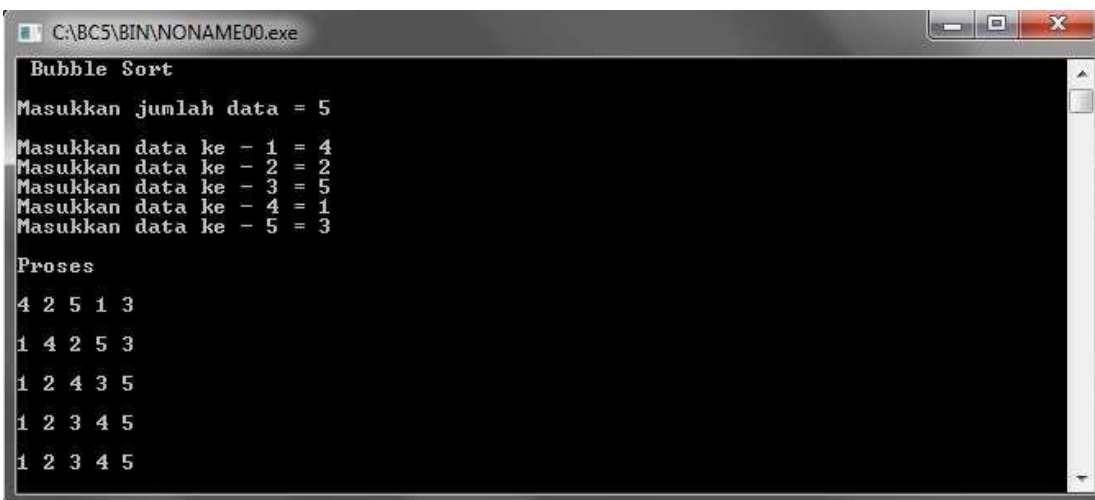
```
}
```

```

cout<<endl<<endl;
}
void bubble_sort()
{
for(int i=1;i<d;i++)
{ for(int j=d-1;j>=i;j--)
{
if(data[j]<data[j-1]) tukar(j,j-1); }
Tampil();
}
cout<<endl; }
main()
{
cout<<" Bubble Sort"<<endl<<endl;
Input();
cout<<"Proses"<<endl<<endl;
Tampil();
bubble_sort();
getch(); }

```

Hasilnya :



```

Bubble Sort
Masukkan jumlah data = 5
Masukkan data ke - 1 = 4
Masukkan data ke - 2 = 2
Masukkan data ke - 3 = 5
Masukkan data ke - 4 = 1
Masukkan data ke - 5 = 3

Proses
4 2 5 1 3
1 4 2 5 3
1 2 4 3 5
1 2 3 4 5
1 2 3 4 5

```

Gambar 6.8

c. Insertion Sort

Prinsip Kerja *Insertion Sort* adalah

1. Pengecekan mulai dari data ke-1 sampai data ke-n
2. Bandingkan data ke-I (I = data ke-2 s/d data ke-n)
3. Bandingkan data ke-I tersebut dengan data sebelumnya (I-1), Jika lebih kecil maka data tersebut dapat disisipkan ke data awal sesuai dgn posisi yg seharusnya
4. Lakukan langkah 2 dan 3 untuk bilangan berikutnya (I= I+1) sampai didapatkan urutan yg optimal.

Contoh :

```
#include <iostream.h>
#include <conio.h>
int data[10],data2[10];
int n;
void tukar(int a, int b)
{
    int t;
    t = data[b];
    data[b] = data[a];
    data[a] = t;
}
void insertion_sort()
{
    int temp,i,j;
    for(i=1;i<=n;i++)
    {
        temp = data[i];
        j = i -1;
        while(data[j]>temp && j>=0)
        {
            data[j+1] = data[j];
            j--;
        }
    }
}
```

```

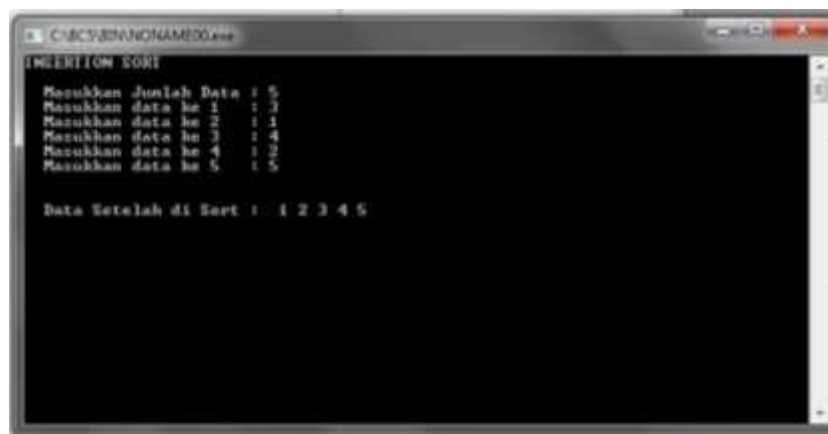
    }
    data[j+1] = temp;
    }
    }

    void main()
    {
        cout<<"INSERTION SORT"<<endl;
        cout<<endl;
        cout<<" Masukkan Jumlah Data : ";cin>>n;

        for(int i=1;i<=n;i++)
        {
            cout<<" Masukkan data ke "<<i<<" : ";
            cin>>data[i];
            data2[i]=data[i];}
        insertion_sort();
        cout<<endl<<endl;
        cout<<" Data Setelah di Sort : ";
        for(int i=1; i<=n; i++)
        {
            cout<<" "<<data[i];}
        getch();}

```

Hasilnya :



Gambar 6.9

LATIHAN 6

Buatlah sebuah program berupa memasukkan nilai dari 10 orang mahasiswa dengan menggunakan array.

Masukkan jumlah Mahasiswa : < input >

Masukkan Nama Mahasiswa : < input >

Masukkan Nilai Mahasiswa : < input >

Nama Mahasiwa : < output >

Nilai : < output >

Nama Mahasiwa : < output >

Nilai : < output >

Nama Mahasiwa : < output >

Nilai : < output >