# BAB XI
# POINTER

## *Tujuan*

1. Menjelaskan tentang konsep dari variabel pointer
2. Menjelaskan tentang pointer array
3. Menjelaskan tentang pointer string
4. Menjelaskan tentang pointer dalam fungsi

## *Pendahuluan*

Pointer adalah sebuah variabel yang isi datanya adalah alamat memori atau variabel lain. Sehingga pointer dapat juga disebut sebagai variabel alamat (address variable).

### 1. Deklarasi Variabel Pointer

Suatu variable pointer dideklarasikan sebagai berikut :

```
tipe_data   *nama_variabel;
```

Tipe_data yg ditunjuk pointer

Nama variabel pointer

Agar suatu pointer menunjuk ke variabel lain, mula-mula pointer harus diisi dengan alamat dari variabel yang ditunjuk. Untuk menyatakan alamat dari suatu variabel, digunaan operator & didepan nama variabel. Contoh, jika x adalah variabel bertipe int, maka pointer px yang berisi alamat dari variabel x ditulis sebagai :

```
px = &x
```

Untuk mengakses isi dari suatu variabel melalui pointer, perlu dinyatakan dulu penunjukkan alamat dari variabel tersebut. Dua pernyataan dibawah ini menunjukkan bahwa nilai y adalah sama dengan nilai x.

`px=&x` → pointer px berisi alamat dari variabel x

`y= *px` → variabel y berisi nilai yang ditunjuk oleh pointer px

Antara tipe pointer dan tipe obyek yang akan ditunjuk oleh pointer haruslah sejenis. Jika tidak sejenis akan menimbulkan kesalahan. Contoh :

```
int x;      →   x adalah obyek
int *px;    →   px adalah pointer yang menunjuk obyek x
```

**2. Pointer dan Array**

Pointer dan array dalam C mempunyai hubungan yang sangat erat. Sebab sesungguhnya array secara internal akan diterjemahkan kedalam bentuk pointer. Contoh :

```
int tgl_lahir[3]={16,10,1993};   →   array tgl_lahir dengan panjang 3
int *ptgl;                       →   pointer ptgl
```

Jika dilanjutkan dengan instruksi

```
ptgl = &tgl_lahir[0];
```

maka pointer ptgl berisi alamat dari array tgl_lahir yang ke 0 (alamat pertama dari array tlg_lahir). Jika ingin menampilkan seluruh elemen array tgl_lahir, instruksi yang biasa dilakukan adalah berupa :

```
for (i=0;i<3;i++)
     printf("%d",tgl_lahir[i]);
```

Instruksi diatas, jika menggunakan pointer dapat dirubah menjadi :

```
ptgl=tgl_lahir;
for(i=0;i<3;i++)
printf("%d",*(ptgl+i));
```

Dimana : `tgl_lahir[i]` dapat diganti dengan variable pointer `*(ptgl+i)`

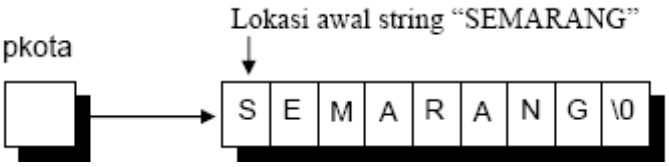
**3. Pointer dan Array**

Seperti halnya array, hubungan pointer dengan string juga dekat. Jika sebuah instruksi dinyatakan sebagai berikut :

```
char *pkota = "SEMARANG";
```

Instruksi di atas diartikan sebagai :

- Kompiler akan mengalokasikan variabel **pkota** sebagai variabel pointer yang akan menunjuk ke obyek bertipe **char** dan menempatkan konstanta "SEMARANG" dalam suatu memori.
- Kemudian pointer pkota akan menujuk ke lokasi string "SEMARANG"




**4. Pointer dan Fungsi**

```
void naikkan_nilai(int *x, int *y)
      {
            *x = *x+2;
            *y= *y+2;
      }
```

Fungsi diatas dimaksudkan agar jika dipanggil, variabel yang berkaitan dengan parameter aktual dapat diubah nilainya. Pemanggilannya dapat berupa :

```
naikkan_nilai(&a, &b);
```

Variabel a dan b harus diawali dengan operator & yang menyatakan sebagai alamat variabel, karena parameter fungsinya berupa pointer.

**Contoh :**

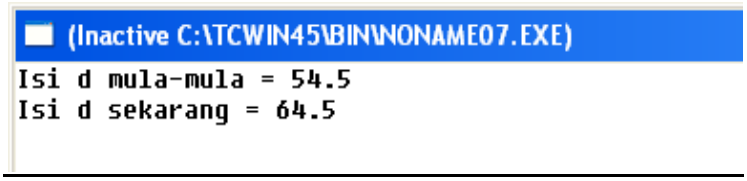
**1. Mengakses isi suatu variabel melalui pointer**

```
#include <stdio.h>
main()
{
   int y, x = 87; /* x & y bertipe int */
   int *px;
   /* var pointer yg menunjuk ke data yang bertipe int */
   px = &x; /* px diisi dgn alamat dari variabel x */
   y = *px; /* y diisi dgn nilai yang ditunjuk oleh px */
   printf("Alamat x = %p\n", &x);
   printf("Isi px = %p\n", px);
   printf("Isi x = %d\n", x);
   printf("Nilai yang ditunjuk oleh px = %d\n", *px);
   printf("Nilai y = %d\n", y);
}
```

```
(Inactive C:\TCWIN45\BIN\NONAME06.EXE)
Alamat x = 208F:2232
Isi px = 208F:2232
Isi x = 87
Nilai yang ditunjuk oleh px = 87
Nilai y = 87
```

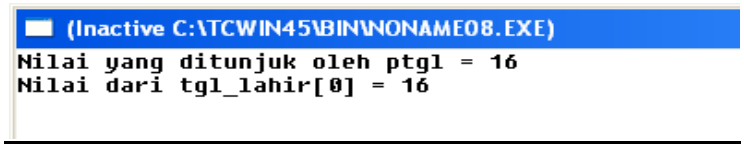
## 2. Mengubah isi suatu variabel melalui pointer

```
#include <stdio.h>
main()
{
   float d, *pd;
   d=54.5;
   printf("Isi d mula-mula = %g\n", d);
   pd = &d;
   *pd = *pd+10;
   printf("Isi d sekarang = %g\n", d);
}
```



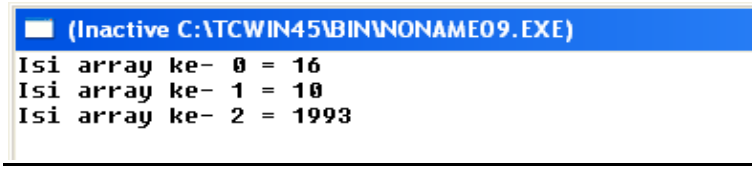

## 3. Pointer dan Array

```
#include<stdio.h>
main()
{
     int tgl_lahir[]={16,10,1993};
     int *ptgl;
     ptgl=tgl_lahir; /*berisi alamat arry */
     printf("Nilai yang ditunjuk oleh ptgl = %d\n",*ptgl);
     printf("Nilai dari tgl_lahir[0] = %d\n",tgl_lahir[0]);
}
```



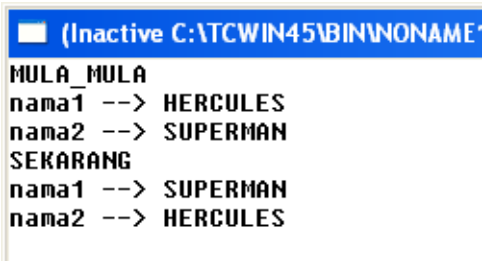

## 4. Menampilkan seluruh elemen array

```
#include<stdio.h>
main()
{
     int tgl_lahir[]={16,10,1993};
     int *ptgl,i;
     ptgl=tgl_lahir; /*berisi alamat arry */
     for(i=0;i<3;i++)
     printf("Isi array ke- %d = %d\n",i,*(ptgl+i));
}
```



## 5. Pointer dan String

```
#include<stdio.h>
main()
{
     char *nama1="HERCULES";
     char *nama2="SUPERMAN";
     char *namax;
     puts("MULA_MULA");
     printf("nama1 --> %s\n",nama1);
     printf("nama2 --> %s\n",nama2);
     namax=nama1;
     nama1=nama2;
     nama2=namax;
     puts("SEKARANG");
     printf("nama1 --> %s\n",nama1);
     printf("nama2 --> %s\n",nama2);
}
```




## 6. Pointer sebagai Parameter Fungsi

```
#include<stdio.h>
void naikkan_nilai(int *x, int *y);
main()
{
     int a=3, b=7;
     printf("Awal a= %d, b= %d\n",a,b);
     naikkan_nilai(&a,&b);
     printf("Akhir a= %d, b= %d\n",a,b);
}

void naikkan_nilai(int *x, int *y)
{
     *x = *x+2;
     *y = *y+2;
}
```

