

Bab 12. Pointer 2

Konsep Pemrograman
Politeknik Elektronika Negeri Surabaya
2006

Overview

- *Pointer to array*
- *Pointer to string*
- *Array of pointer*
- *Pointer to pointer*

Pointer to Array

- Hubungan antara pointer dan array pada C sangatlah erat.
- Ingat bahwa sesungguhnya array secara internal akan diterjemahkan dalam bentuk pointer

array yang dituliskan tanpa kurung sikunya \Leftrightarrow alamat dr elemen pertama (indeks ke-0) dr array tsb.

Pointer to Array

```
#include <stdio.h>
```

```
main()
```

```
{
```

```
    int tgl_lahir[] = {16, 8, 2003};
```

```
    int *ptgl;
```

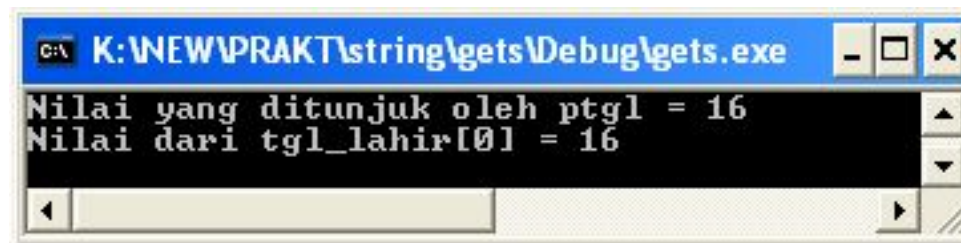
```
    ptgl = tgl_lahir; ←
```

```
    printf("Nilai yang ditunjuk oleh ptgl = %d\n", *ptgl);
```

```
    printf("Nilai dari tgl_lahir[0] = %d\n", tgl_lahir[0]);
```

```
}
```

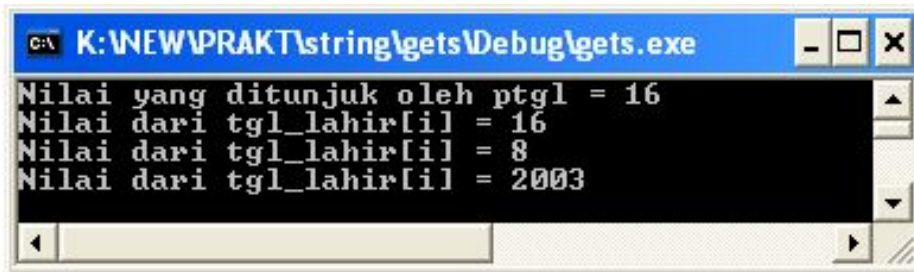
ptgl = tgl_lahir;
artinya sama dengan
ptgl = &tgl_lahir[0];
ptgl adalah pointer to array of integer



Pointer to Array

```
#include <stdio.h>
main()
{
    int tgl_lahir[] = {16, 8, 2003};
    int *ptgl, i;

    ptgl = tgl_lahir;
    printf("Nilai yang ditunjuk oleh ptgl = %d\n", *ptgl);
    for (i=0; i<3; i++)
        printf("Nilai dari tgl_lahir[i] = %d\n", *(ptgl+i));
}
```



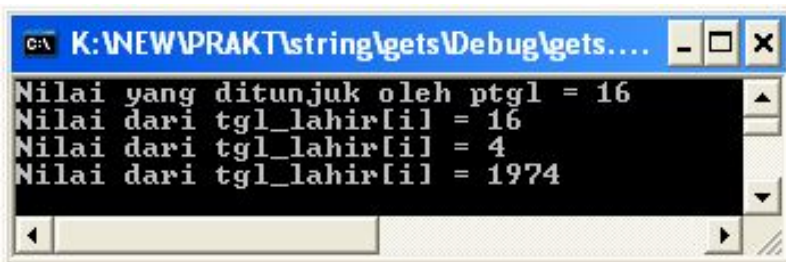
Untuk setiap iterasi, indeks i dinaikkan sebesar : **1 x size** dari satu elemen array

Pointer to Array

```
#include <stdio.h>
main()
{
    int tgl_lahir[] = {16, 4, 1974};
    int I, *ptgl;

    ptgl = tgl_lahir;

    printf("Nilai yang ditunjuk oleh ptgl = %d\n", *ptgl);
    for (i=0; i<3; i++)
        printf("Nilai dari tgl_lahir[i] = %d\n", *ptgl++);
}
```



Post-increment, setiap kali iterasi
address pd `ptgl` dinaikkan sebanyak
1 X size dari satu elemen array

Pointer to Array

- Untuk mengarahkan pointer `ptgl` agar menunjuk ke elemen pertama dari array `tgl_lahir` :

```
ptgl = tgl_lahir;
```

INGAT !!

array `tgl_lahir` dituliskan tanpa kurung sikunya \Leftrightarrow `&tgl_lahir[0]`

- Menampilkan data pada elemen pertama tsb dengan cara :
`*ptgl`
- Untuk mengakses elemen-elemen berikutnya, gunakan looping dengan salah satu cara sbb
 - `*(ptgl+i)`
 - `*ptgl++`

Setiap kali iterasi address pd `ptgl` dinaikkan sebanyak **1 X size** dari satu elemen array

Pointer to String

- String bukanlah sebuah tipe data baru dalam C, melainkan merupakan sekumpulan karakter, sehingga *string = array of char*
- Variabel array yang dituliskan tanpa kurung sikunya \Leftrightarrow alamat/address dari array tsb pada indeks ke-0

Pointer to String

```
#include <stdio.h>
main() {
    char kota[] = "Surabaya", *pkota;

    pkota = kota;                //pkota = &kota[0]
    printf("String yang ditunjuk oleh pkota = ");
    printf("%s\n", pkota);        //printf("%s\n", kota);
}
```

```
-----
#include <stdio.h>
main() {
    //pkota adl pointer yg menunjuk konstanta string "Surabaya"
    char *pkota = "Surabaya";

    printf("String yang ditunjuk oleh pkota = ");
    printf("%s\n", pkota);
}
```

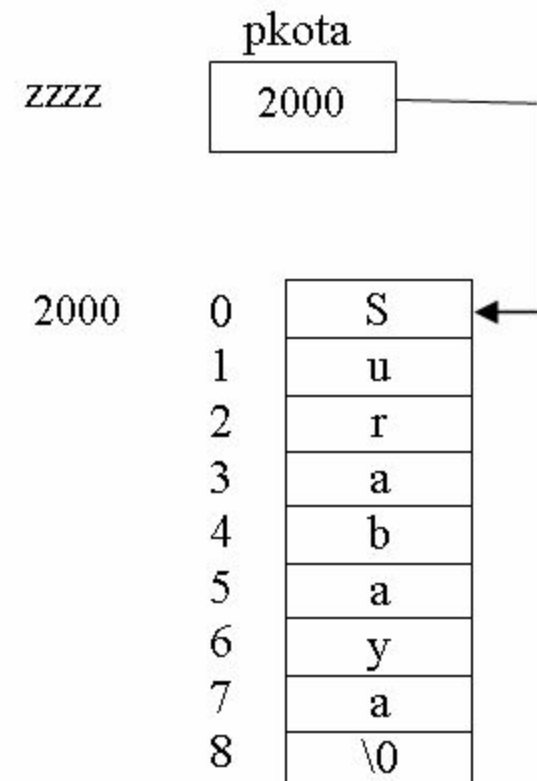
Pointer to String

- Pada program di atas,

```
char *pkota = "Surabaya";
```

akan menyebabkan kompiler mengalokasikan variabel `pkota` sebagai variabel pointer yang menunjuk ke obyek bertipe *char* dan menempatkan konstanta "Surabaya" dalam suatu memori

- kemudian pointer `pkota` akan menunjuk ke lokasi string "Surabaya"

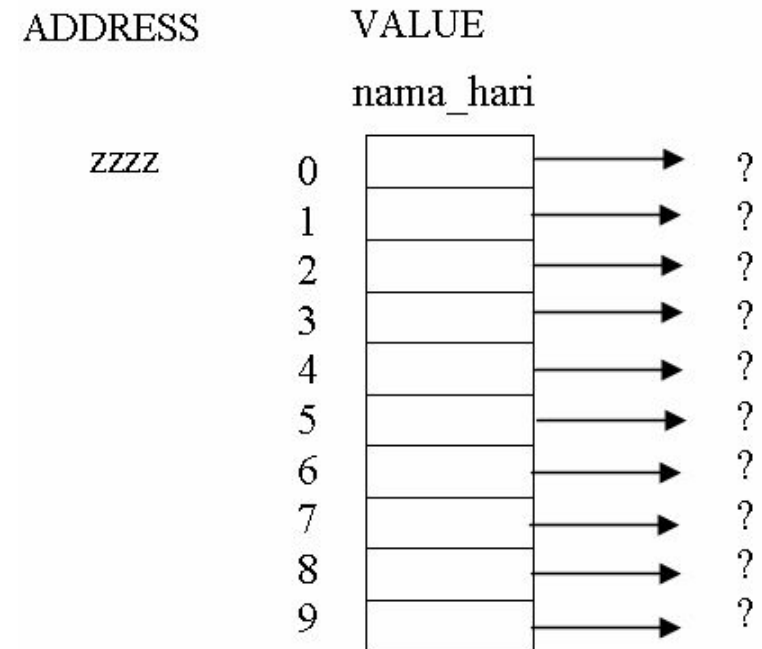


Array of Pointer

- Suatu array bisa digunakan untuk menyimpan sejumlah pointer.
- Jika dideklarasikan :

```
char *nama_hari[10];
```

merupakan pernyataan untuk mendeklarasikan *array of pointer to char*.
- Array `nama_hari` terdiri dari 10 elemen berupa pointer yang menunjuk ke data bertipe *char*.



Array of Pointer

- Array pointer bisa diinisialisasi sewaktu pendeklarasian.
- Jika dideklarasikan:

```
char *namahari[] =  
    {"Senin",  
     "Selasa",  
     "Rabu",  
     "Kamis",  
     "Jumat",  
     "Sabtu",  
     "Minggu"};
```

- Pada contoh ini :
 - namahari[0] berisi alamat/pointer yang menunjuk ke string “Senin”.
 - namahari[1] berisi alamat/pointer yang menunjuk ke string “Selasa”.
 - namahari[2] berisi alamat/pointer yang menunjuk ke string “Rabu”.
 - dan seterusnya

Pointer to Pointer

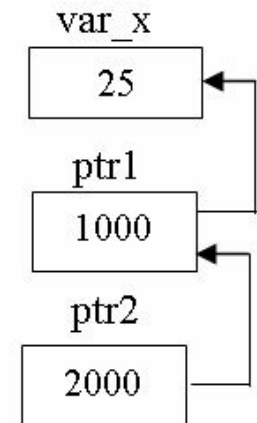
ADDRESS VALUE

- Suatu pointer bisa menunjuk ke pointer yang lain
- Jika dideklarasikan :

```
int var_x = 25, *ptr1, **ptr2;
```

 - `var_x` adalah variabel bertipe *int*.
 - `ptr1` adalah variabel bertipe *pointer to int* → pointer yang menunjuk ke sebuah data bertipe *int*
 - `ptr2` adalah variabel bertipe *pointer to pointer to int* → pointer yang menunjuk ke *pointer to int* (itulah sebabnya deklarasinya berupa `int **ptr2;`)
- Agar `ptr1` menunjuk ke variabel `var_x` dan `ptr2` menunjuk ke `ptr1`, instruksinya sbb : —

```
ptr1 = &var_x;  
ptr2 = &ptr1;
```

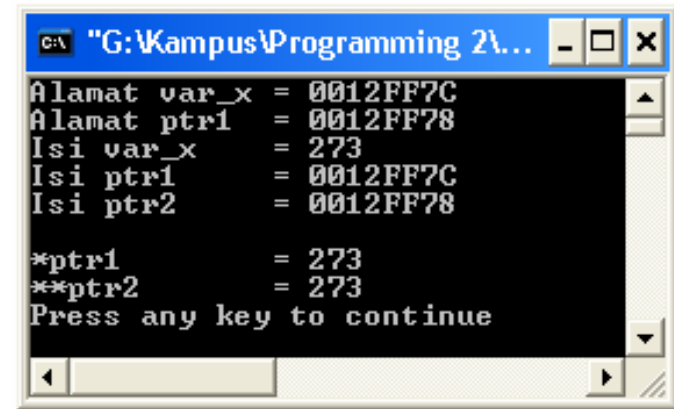


Pointer to Pointer

```
#include <stdio.h>
main() {
    int var_x = 273, *ptr1, **ptr2;

    ptr1 = &var_x;
    ptr2 = &ptr1;

    printf("Alamat var_x = %p\n", &var_x);
    printf("Alamat ptr1 = %p\n", &ptr1);
    printf("Isi var_x = %d\n", var_x);
    printf("Isi ptr1 = %p\n", ptr1);
    printf("Isi ptr2 = %p\n", ptr2);
    printf("\n*ptr1 = %d\n", *ptr1);
    printf("**ptr2 = %d\n", **ptr2);
}
```



```
"G:\Kampus\Programming 2\... - [X]
Alamat var_x = 0012FF7C
Alamat ptr1 = 0012FF78
Isi var_x = 273
Isi ptr1 = 0012FF7C
Isi ptr2 = 0012FF78

*ptr1 = 273
**ptr2 = 273
Press any key to continue
```

Suatu lokasi/address yang telah ditunjuk oleh sebuah pointer, maka lokasi tsb value-nya bisa diakses secara DIRECT maupun INDIRECT melalui pointernya