PRAKTIKUM 25

Pointer 2

A. TUJUAN PEMBELAJARAN

1. Memahami tentang *Pointer to Array*
2. Memahami tentang *Pointer to String*

B. DASAR TEORI

Pointer dan Array *(pointer to array)*

Hubungan antara pointer dan array pada C sangatlah erat. Sebab sesungguhnya array secara internal akan diterjemahkan dalam bentuk pointer. Pembahasan berikut akan memberikan gambaran hubungan antara pointer dan array. Misalnya dideklarasikan di dalam suatu fungsi

```
static int tgl_lahir[3] = { 01, 09, 64 };
```

dan

```
int *ptgl;
```

Kemudian diberikan instruksi

```
ptgl = &tgl_lahir[0]; //pointer to array of integer
```

maka **ptgl** akan berisi alamat dari elemen array **tgl_lahir** yang berindeks nol. Instruksi di atas bisa juga ditulis menjadi

```
ptgl = tgl_lahir;
```

sebab nama array tanpa tanda kurung menyatakan alamat awal dari array. Sesudah penugasan seperti di atas,

```
*ptgl
```

dengan sendirinya menyatakan elemen pertama (berindeks sama dengan nol) dari array **tgl_lahir.**

Pointer dan String *(pointer to string)*

Contoh hubungan pointer dan string ditunjukkan pada program berikut.

```
//Program : ptr4.c
#include <stdio.h>

main() {
    //pkota menunjuk konstanta string "SEMARANG"
    char *pkota = "SEMARANG";

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

Contoh eksekusi :

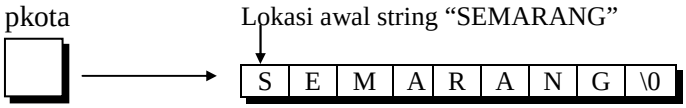
```
String yang ditunjuk oleh pkota = SEMARANG
```

Pada program di atas,

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

akan menyebabkan kompiler

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



Gambar 20.1 Pointer menunjuk data **string**

Pernyataan di atas menyerupai pernyataan

```
char kota[] = "SEMARANG";
```

tetapi sebenarnya kedua pernyataan inisialisasi di depan tidaklah tepat sama. Sebab **`pkota`** adalah pointer (menyatakan alamat) yang dengan mudah dapat diatur agar menunjuk ke string lain (bukan string `"SEMARANG"),` sedangkan **`kota`** adalah array (array menyatakan alamat yang konstan, tak dapat diubah).

Jadi, ada beberapa cara penulisan ekspresi variabel yang menghasilkan sebuah address memory, di antaranya adalah :

1. **Menuliskan ampersand di depan sebuah variabel 'normal'**

   → Misalnya a dideklarasikan sebagai sebuah variabel normal, maka &a akan menghasilkan adress memory dari variabel a tersebut

2. **Menuliskan nama dari sebuah variabel pointer**

   → Misalnya a dideklarasikan sebuah variabel sebagai `int *a, maka a` akan berisi sebuah address memory yang tersimpan pada variabel `a` (setelah ada sebuah address memory yang di-assign ke variabel `a` tersebut)

3. **Menuliskan nama sebuah variabel array tanpa indexnya**

   → Misalnya dideklarasikan

   - `float nilai[5]`, maka penulisan `nilai` artinya sama dengan `&nilai[0]`
   - `int b[3][4], maka b[3]` adalah sama dengan `&b[3][0]` dan `b[0]` adalah sama dengan `&b[0][0].`

Bisa disimpulkan bahwa sebuah variabel array yang dituliskan tanpa kurung siku indexnya adalah sama dengan address memory dari elemen pertama array tersebut

C. PERCOBAAN

Untuk setiap program pada no 1 – 8 di bawah ini,

- gambarkan ilustrasi alokasi memori dari setiap baris pernyataan yang diproses
- perkirakan hasil eksekusinya

Untuk no. 9 buatlah programnya

1. Hubungan antara pointer dan array. Suatu nama array yang ditulis tanpa dengan indeksnya menunjukkan alamat elemen pertama dari array (versi 1).

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

2. Hubungan antara pointer dan array. Suatu nama array yang ditulis tanpa dengan indeksnya menunjukkan alamat elemen pertama dari array (versi 2).

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

3. Hubungan antara pointer dan array. Suatu nama array yang ditulis tanpa dengan indeksnya menunjukkan alamat elemen pertama dari array (versi 3).

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

Analisislah dan jelaskan perbedaan antara aplikasi pada nomor 1, 2 dan 3

4. Menukarkan isi 2 string tanpa pemakaian pointer.

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

#define PANJANG 20
```

```
char nama1[PANJANG] = "AHMAD";
char nama2[PANJANG] = "RIFDA";

main(){
  char namax[PANJANG];

  puts("SEMULA : ");
  printf("nama1 --> %s\n", nama1);
  printf("nama2 --> %s\n", nama2);

  strcpy(namax, nama1);
  strcpy(nama1, nama2);
  strcpy(nama2, namax);

  puts("KINI : ");
  printf("nama1 --> %s\n", nama1);
  printf("nama2 --> %s\n", nama2);
}
```

5. Menukarkan isi 2 string dengan fasilitas pointer.

```
#include <stdio.h>

char *nama1 = "AHMAD";
char *nama2 = "RIFDA";

main(){
  char *namax;

  puts("SEMULA : ");
  printf("nama1 --> %s\n", nama1);
  printf("nama2 --> %s\n", nama2);

  namax = nama1;
  nama1 = nama2;
  nama2 = namax;

  puts("KINI : ");
  printf("nama1 --> %s\n", nama1);
  printf("nama2 --> %s\n", nama2);
 }
```

Analisislah dan jelaskan perbedaan antara aplikasi pada nomor 4 dengan nomor 5

6. Penggunaan *variable index* pada *array* dan *variable index* pada *pointer*, untuk menunjuk suatu nilai data di dalam suatu *variable array*. Berikan analisis dan kesimpulan

```
main(){
   int nilai[10]={86,75,98,66,56,76,80,95,70,60};
   int index, *ip;

   printf("Mencetak menggunakan array\n");
   printf("Daftar nilai siswa\n\n");

   for(index=0; index<10; index++)
      printf("%3d",nilai[index]);

   puts("\n");
   printf("Mencetak menggunakan pointer dan index\n");
   printf("Daftar nilai siswa\n\n");

   for(index=0; index<10; index++)
      printf("%3d",*(nilai+index));

   puts("\n");
   printf("Mencetak menggunakan pointer\n");
   printf("Daftar nilai siswa\n\n");

   ip=nilai;
   for(index=0; index<10; index++)
           printf("%3d",*ip++);
}
```

7. Berikan ilustrasi dan jelaskan apa yang dilakukan oleh potongan program di bawah ini

```
main()    {
    char *text_pointer = "Good morning!";

    for( ; *text_pointer != '\0'; ++text_pointer)
      printf("%c", *text_pointer);
}
```

8. Berikan ilustrasi dan jelaskan apa yang dilakukan oleh potongan program di bawah ini

```
int array1[10], array2[10];
int *ip1, *ip2 = array2;
int *akhir = &array1[10];
```

```
for(ip1 = &array1[0]; ip1 < akhir; ip1++)
      *ip2++ = *ip1;
```

9. Definisikan sebuah fungsi untuk membaca sebuah array (dengan tipe sembarang) menggunakan pointer. Buatlah program untuk membaca array tersebut dalam rangka mencari sebuah nilai tertentu dan laporkan hasilnya berhasil menemukan atau tidak

D. LAPORAN RESMI

1. Kumpulkan listing program, ilustrasi alokasi memorinya beserta hasil eksekusinya.

2. Untuk no 9 : kumpulkan listing, capture output dan analisisnya