PRAKTIKUM 12

POINTER 2

A. Tujuan

1. Menjelaskan tentang Array of Pointer
2. Menjelaskan tentang Pointer to Pointer
3. Menjelaskan tentang Pointer dalam Fungsi
4. Menjelaskan tentang Pointer sebagai Parameter Fungsi
5. Menjelaskan tentang Pointer sebagai Keluaran Fungsi

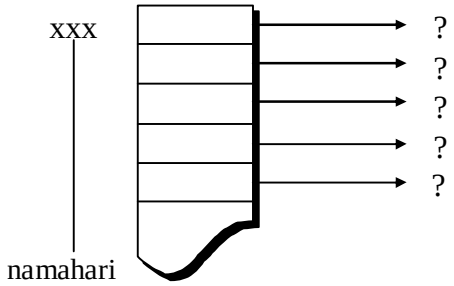
B. Dasar Teori

Array dari Pointer (Array of Pointer)

- Suatu array bisa digunakan untuk menyimpan sejumlah pointer. Sebagai contoh:

```
char *namahari[10];
```

merupakan pernyataan untuk mendeklarasikan array pointer. Array **namahari** terdiri dari 10 elemen berupa pointer yang menunjuk ke data bertipe *char*.



Gambar 8.5 Array pointer

- Array pointer bisa diinisialisasi sewaktu pendeklarasian. Sebagai contoh:

```
static char *namahari[] =

{“Senin”,
“Selasa”,
“Rabu”,
“Kamis”,
“Jumat”,
“Sabtu”,
“Minggu”};
```

Pada contoh ini,

namahari[0] berisi alamat yang menunjuk ke string “Senin”.

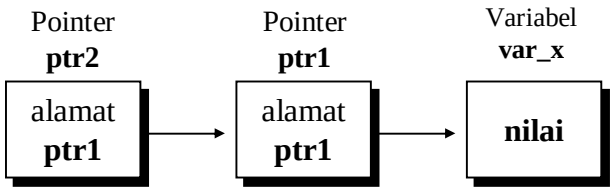
namahari[1] berisi alamat yang menunjuk ke string “Selasa”.

namahari[2] berisi alamat yang menunjuk ke string “Rabu”.

dan sebagainya.

Pointer menunjuk Pointer (Pointer to Pointer)

Suatu pointer bisa saja menunjuk ke pointer lain. Gambar berikut memberikan contoh mengenai pointer menunjuk pointer.



Gambar 8.6 Pointer yang menunjuk pointer

- Untuk membentuk rantai pointer seperti pada gambar di atas, pendeklarasian yang diperlukan berupa

```
int var_x;
int *ptr1;
int **ptr2;
```

Perhatikan pada deklarasi di depan:

- **var_x** adalah variabel bertipe int.
- **ptr1** adalah variabel pointer yang menunjuk ke data bertipe *int.*
- **ptr2** adalah variabel pointer yang menunjuk ke pointer *int.*
  (itulah sebabnya deklarasinya berupa *int **ptr2;*)
- Agar **ptr1** menunjuk ke variabel **var_x**, perintah yang diperlukan berupa

```
ptr1 = &var_x;
```

- Sedangkan supaya **ptr2** menunjuk ke **ptr1**, instruksi yang diperlukan adalah

```
ptr2 = &ptr1;
```

Pointer dalam Fungsi

Pointer dan kaitannya dengan fungsi yang akan dibahas berikut meliputi :

- Pointer sebagai parameter fungsi
- Pointer sebagai keluaran fungsi

Pointer Sebagai Parameter Fungsi

- Penerapan pointer sebagai parameter yaitu jika diinginkan agar nilai suatu variabel internal dapat diubah oleh fungsi yang dipanggil.
- Sebagai contoh dapat dilihat pada fungsi berikut.

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

- Fungsi di atas dimaksudkan agar kalau dipanggil, variabel yang berkenaan dengan parameter aktual dapat diubah nilainya, masing-masing dinaikkan sebesar 2. Contoh pemanggilan :

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

- Perhatikan, dalam hal ini variabel **a** dan **b** harus ditulis diawali operator alamat (**&**) yang berarti menyatakan alamat variabel, sebab parameter fungsi dalam pendefinisian berupa pointer.

Pointer Sebagai Keluaran Fungsi *(return value)*

- Suatu fungsi dapat dibuat agar keluarannya berupa pointer. Misalnya, suatu fungsi menghasilkan keluaran berupa pointer yang menunjuk ke string nama_bulan, seperti pada contoh berikut.

```
char *nama_bulan(int n)
{
    static char *bulan[]=
    {“Kode bulan salah”, “Januari”, “Februari”, Maret”,
          “April”, “Mei”, “Juni”, “Juli”, "Agustus”,
          “September”, “Oktober”, “Nopember”,“Desember”
    };
    return ( (n<1  | |  n>12) ? bulan[0] : bulan[n] );
}
```

- Pada definisi fungsi di atas,

```
char *nama_bulan()
```

menyatakan bahwa keluaran fungsi **nama_bulan()** berupa pointer yang menunjuk ke obyek char (atau string).

- Dalam fungsi **nama_bulan()**, mula-mula array bernama **bulan** dideklarasikan dan sekaligus diinisialisasi agar menunjuk sejumlah string yang menyatakan nama bulan. Di bagian akhir fungsi, pernyataan

```
return ( (n<1 || n>12) ? bulan[0] : bulan[n] );
```

menyatakan bahwa hasil fungsi berupa pointer yang menunjuk ke

→ string "Kode bulan salah" (**bulan[0]**) jika masukan fungsi n<1 atau n>12

→ **bulan[n]** untuk n yang terletak antara 1 sampai dengan 12.

C. TUGAS PENDAHULUAN

1. Buat program dengan menggunakan fungsi! Serta gambarkan ilustrasi alokasi memori dari setiap baris pernyataan!

```
#include <stdio.h>
main()
{
int a = 4;
int b = 7;
int *x, *y;
x=&a;
y=&b;
printf("SEMULA : a = %d  b = %d\n", a, b);
*x = *x * 4;
*y = *y + *x;
printf("KINI   : a = %d  b = %d\n", a, b);
}
```

D. PERCOBAAN

1. Tuliskan alokasi memori dari setiap baris pernyataan berikut serta tampilkan hasilnya!

```
#include <stdio.h>

main()
{
int var_x = 20;
```

```
int *ptr1;
int **ptr2;

ptr1 = &var_x;
ptr2 = &ptr1;
*ptr1=var_x +**ptr2;
printf("Nilai var_x = *ptr1  = %d\n", *ptr1);
printf("Nilai var_x = **ptr2 = %d\n\n", **ptr2);

printf("ptr1 = &var_x = %p\n", ptr1);
printf("ptr2 = &ptr1  = %p\n", ptr2);
printf("       &ptr2  = %p\n", &ptr2);

}
```

2. Berikan ilustrasi dan jelaskan apa yang dilakukan oleh program di bawah ini dan tampilkan hasil eksekusinya.

```
#include <stdio.h>

int r, q = 10;

int go_crazy(int *, int *);

main()
{
   int *ptr1 = &q;
   int *ptr2 = &q;

   r = go_crazy(ptr2, ptr1);
   printf("q = %d, r = %d, *ptr1 = %d, *ptr2 = %d\n",
        q, r,*ptr1,*ptr2);

   ptr1 = &r;

   q = go_crazy(ptr1, ptr2);
   r=r*5;
   q = q + r;
   printf("q = %d, r = %d, *ptr1 = %d, *ptr2 = %d\n",
             q, r,*ptr1,*ptr2);
   }

   int go_crazy(int *p1, int *p2)
   {
     int x = 5;

     r = 12;
     *p2 = *p1 * 2;
     p1 = &x;
     return *p1 * 3;
   }
}
```

3. Berikan ilustrasi dan jelaskan apa yang dilakukan oleh program di bawah ini dan tampilkan hasil eksekusinya.

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

```
char strA[80] = "halo apa kabar";
char strB[80];

main()
{

 char *pA, *pB;

 puts(strA);
 pA = strA;
 puts(pA);
 pB = strB;
 putchar('\n');

 while(*pA != '\0')
 {
     *pB++ = *pA++;
 }
 *pB = '\0';
 puts(strB);
 }
```

Proses apakah yang sebenarnya dilakukan pada program tersebut ?

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

```
 #include <stdio.h>
char *my_strcpy(char *, char *);

main()
 {

     char strA[80]="A string to be used for demonstration";
     char strB[80];

     my_strcpy(strB, strA);
      puts(strB);
 }

 char *my_strcpy(char *destination, char *source)
 {
    char *p = destination;
    while (*source != '\0')
    {
        *p++ = *source++;
    }
    *p = '\0';
    return destination;
 }
```

5. Bandingkan fungsi my_strcpy di atas dengan fungsi my_strcpy di bawah ini. Berikan penjelasan terhadap perbedaan proses dari kedua fungsi tersebut

```
  char *my_strcpy(char dest[], char source[])
 {
     int i = 0;
     while (source[i] != '\0')
```

```
    {
        dest[i] = source[i];
        i++;
    }
    dest[i] = '\0';
    return dest;
}
```

E. LAPORAN RESMI

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