Praktikum 1

Array, Pointer dan Struktur

A. TUJUAN PEMBELAJARAN

Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu:

1. Memahami konsep struktur data array dalam Bahasa C
2. Memahami konsep pointer dalam Bahasa C
3. Memahami konsep struktur dalam Bahasa C
4. Mengerti perbedaan penggunaan array dan pointer
5. Mengerti perbedaan array dan struktur

B. DASAR TEORI

B.1 Array

Suatu array berdimensi satu dideklarasikan dalam bentuk umum berupa :

```
tipe_data nama_var[ukuran];
```

dengan :

- tipe_data : untuk menyatakan tipe dari elemen array, misalnya *int, char, float*.
- nama_var : nama variabel array
- ukuran : untuk menyatakan jumlah maksimal elemen array.

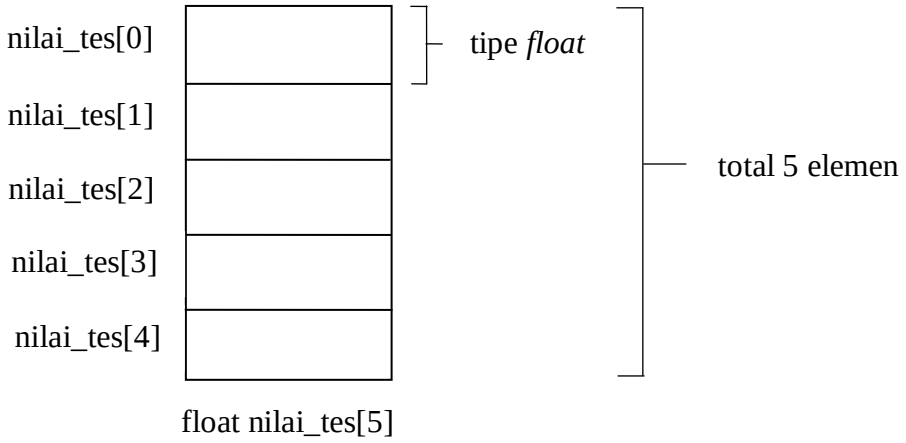
Contoh pendeklarasian array :

```
float nilai_tes[5];
```

menyatakan bahwa array **nilai_tes** mengandung 5 elemen bertipe *float.*

Pada C, data array akan disimpan dalam memori yang berurutan. Elemen pertama mempunyai indeks bernilai 0. Jika variabel **nilai_tes** dideklarasikan sebagai array dengan 5 elemen, maka elemen pertama memiliki indeks sama dengan 0, dan

elemen terakhir memiliki indeks 4. Gambar 1.1 di bawah ini menjelaskan urutan komponen dalam array.



Gambar 1.1 Array berdimensi satu

Bentuk umum pengaksesan array adalah sbb :

```
nama_var[indek
```

sehingga, untuk array **nilai_tes**, maka :

`nilai_tes[0]` → elemen pertama dari **nilai_tes**

`nilai_tes[4]` → elemen ke-5 dari **nilai_tes**

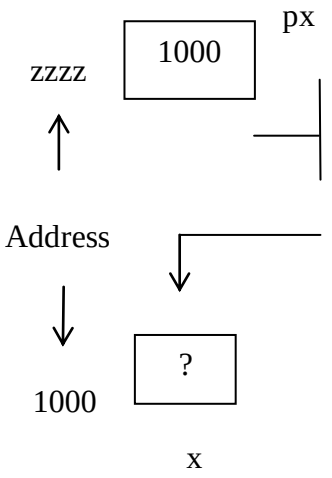
Contoh :

```
nilai_tes[0] = 70;      /* contoh 1 */
scanf("%f", &nilai_tes[2]);/* contoh 2 */
```

Contoh pertama merupakan pemberian nilai 70 ke **nilai_tes[0]**. Sedangkan contoh 2 merupakan perintah untuk membaca data bilangan dari keyboard dan diberikan ke **nilai_tes[2]**.

B.2 Pointer

Variabel pointer sering dikatakan sebagai variabel yang menunjuk ke obyek lain. Pada kenyataan yang sebenarnya, variabel pointer berisi alamat dari suatu obyek lain (yaitu obyek yang dikatakan ditunjuk oleh pointer). Sebagai contoh, **px** adalah variabel pointer dan **x** adalah variabel yang ditunjuk oleh **px**. Kalau **x** berada pada alamat memori (alamat awal) 1000, maka **px** akan berisi 1000. Sebagaimana diilustrasikan pada gambar 8.1 di bawah ini



Gambar 1.2 Variabel pointer px menunjuk ke variabel x

B.2.1 Mendeklarasikan Variabel Pointer

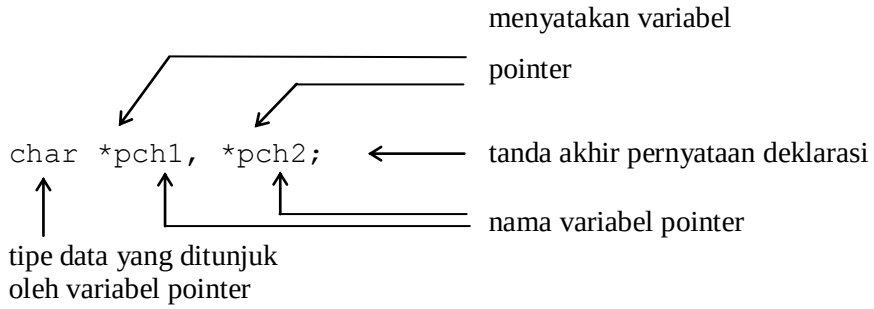
Suatu variabel pointer dideklarasikan dengan bentuk sebagai berikut :

```
tipe *nama_variabel
```

dengan **tipe** dapat berupa sembarang tipe data dalam bahasa C. Adapun **nama_variabel** adalah nama dari variabel pointer. Sebagai contoh :

```
int *px;                    / *contoh 1 */
char *pch1, *pch2;          / *contoh 2 */
```

Contoh pertama menyatakan bahwa **px** adalah variabel pointer yang menunjuk ke suatu data bertipe *int*, sedangkan contoh kedua masing **pch1** dan **pch2** adalah variabel pointer yang menunjuk ke data bertipe *char*.



Gambar 1.3 Ilustrasi pendeklarasian variabel pointer

2.2 Mengatur Pointer agar Menunjuk ke Variabel Lain

Agar suatu pointer menunjuk ke variabel lain, mula-mula pointer harus diisi dengan alamat dari variabel yang akan ditunjuk. Untuk menyatakan alamat dari suatu variabel, operator **&** (operator alamat, bersifat *unary*) bisa dipergunakan, dengan menempatkannya di depan nama variabel. Sebagai contoh, bila **x** dideklarasikan sebagai variabel bertipe *int*, maka

```
&x
```

berarti "alamat dari variabel **x**". Adapun contoh pemberian alamat **x** ke suatu variabel pointer px (yang dideklarasikan sebagai pointer yang menunjuk ke data bertipe *int*) yaitu :

```
px = &x;
```

Pernyataan di atas berarti bahwa **px** diberi nilai berupa alamat dari variabel **x**. Setelah pernyataan tersebut dieksekusi barulah dapat dikatakan bahwa **px** menunjuk ke variabel **x**.

B.2.2 Mengakses Isi Suatu Variabel Melalui Pointer

Jika suatu variabel sudah ditunjuk oleh pointer, variabel yang ditunjuk oleh pointer tersebut dapat diakses melalui variabel itu sendiri (pengaksesan langsung) ataupun melalui pointer (pengaksesan tak langsung). Pengaksesan tak langsung dilakukan dengan menggunakan operator *indirection* (tak langsung) berupa simbol ***** (bersifat *unary*). Contoh penerapan operator * yaitu :

```
*px
```

yang menyatakan "isi atau nilai variabel/data yang ditunjuk oleh pointer **px**" . Sebagai contoh jika **y** bertipe *int*, maka sesudah dua pernyataan berikut

```
px = &x;
y = *px;
```

y akan berisi nilai yang sama dengan nilai **x**.

B.2.3 Pointer dan 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];
```

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.**

B.3 Struktur

Struktur adalah koleksi dari variabel yang dinyatakan dengan sebuah nama, dengan sifat setiap variabel dapat memiliki tipe yang berlainan. Struktur dapat digunakan untuk mengelompokkan beberapa informasi yang berkaitan menjadi sebuah satu kesatuan.

Bentuk umum deklarasi struktur adalah sebagai berikut:

```
struct nama_tipe_struktur
{
      tipe field1;
      .
      .
      tipe fieldN;
} variabel_struktur1, …, variabel_strukturM;
```

Elemen dari struktur dapat diakses dengan menggunakan bentuk:

```
variabel_struktur.nama_field
```

B.3.1 Array dan Struktur

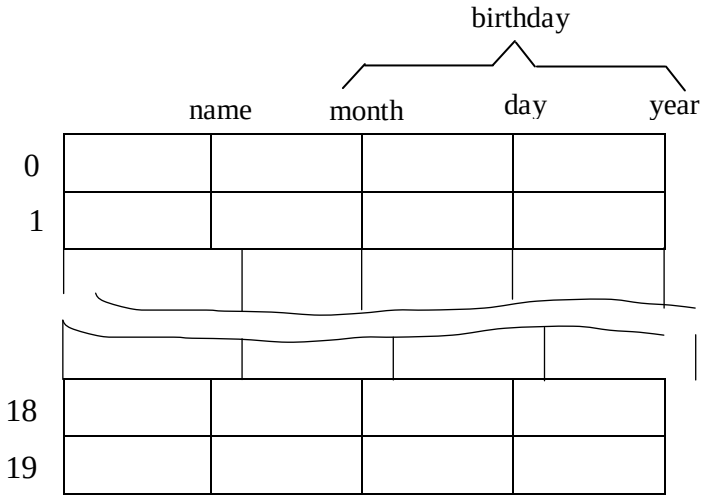
Elemen-elemen dari suatu array juga dapat berbentuk sebuah struktur. Misalnya array yang dipakai untuk menyimpan sejumlah data siswa *(struct student).* Array struktur berdimensi satu ini membentuk suatu tabel, dengan barisnya menunjukkan elemen dari array-nya dan kolomnya menunjukkan elemen dari struktur. Dalam hal ini maka deklarasi yang dibutuhkan adalah sebagai berikut :

```
#define MAKS 20
        .
        .
        .
struct date {       /* definisi dari tipe date */
     int month;
     int day;
     int year;
};

struct person {     /* definisi dari tipe person */
     char name[30];
     struct date birthday;
};

/* deklarasi dari variabel array student */
struct person student[MAKS];
```

yang artinya, mendeklarasikan array **student** yang memiliki elemen yang bertipe *struct* **person** sebanyak **MAKS.** Setelah array **student** dideklarasikan, maka ruang yang disediakan ditunjukkan dalam Gambar 1.4 di bawah ini.



Gambar 1.4 Array dari struktur

Elemen-elemen dari array stuktur tersebut bisa diakses dengan cara sebagai berikut :

```
for (i=0; i<MAKS; i++)
{
     printf("Name            : ");
     fgets(student[i].name, sizeof student[i].name,
          stdin);
     printf("Birthday (mm-dd-yyyy): ");
     scanf("%d-%d-%d", &student[i].birthday.month,
          &student[i].birthday.day,
          &student[i].birthday.year);
     printf("\n");

     /* hapus sisa data dalam penampung keyboard */
     fflush(stdin);
};
```

C. TUGAS PENDAHULUAN

Buatlah flowchart untuk tiap permasalahan yang diberikan pada latihan soal no 1 & 3 di bawah ini sebagai Tugas Pendahuluan.

D. PERCOBAAN

1. Buatlah workspace untuk praktikum Struktur Data dengan menggunakan Visual C++.
2. Buatlah project untuk praktikum pertama.
3. Cobalah untuk masing-masing percobaan di bawah ini.
4. Selesaikan soal-soal yang ada dengan mengimplementasikan flowchart yang anda buat pada Tugas Pendahuluan.

Percobaan 1 : Cara memberikan nilai pada array dan mengakses array

```
#include <stdio.h>

void main ()
{
   int n[ 10 ];
   int i,j;


   for ( i = 0; i < 10; i++ )
   {
      n[ i ] = i + 100;
   }

   for (j = 0; j < 10; j++ )
   {
      printf("Element[%d] = %d\n", j, n[j] );
   }

}
```

Percobaan 2 : Menghitung total dari nilai yang terdapat pada array

```
#include <stdio.h>
#define SIZE 12


void main()
{

    int a[ SIZE ] = { 1, 3, 5, 4, 7, 2, 99,16, 45, 67, 89, 45 };
    int i = 0;
    int total = 0;
    for(i = 0; i<SIZE; ++i)
    {
          total+=a[i];
    }

    printf( "Total elemen yang terdapat di array : %d\n", total );
}
```

Percobaan 3 : Penggunaan Array pada Bilangan Fibonaci

```
#include<stdio.h>
#defineMAX 20

int fibo[MAX];

void main()
{
     int i;

     fibo[1] = 1;
     fibo[2] = 1;

     for (i=3;i<=MAX;i++)
       fibo[i]=fibo[i-2]+fibo[i-1];

     printf("%d Bilangan Fibonaci Pertama adalah : \n",MAX);
     for (i=1;i<MAX;i++)
       printf("%d",fibo[i]);
}
```

Percobaan 4 : Cara mengakses array dua dimensi

```
#include <stdio.h>

void main ()
{

   int a[5][2] = { {0,0}, {1,2}, {2,4}, {3,6},{4,8}};
   int i, j;

      for ( i = 0; i < 5; i++ )
   {
      for ( j = 0; j < 2; j++ )
      {
         printf("a[%d][%d] = %d\n", i,j, a[i][j] );
      }
   }
}
```

Percobaan 5 : Program Merubah Isi Variabel melalui Pointer

```
#include <stdio.h>

main()
{
      int y, x = 87;                        /* x & y bertipe int */
      int *px;
      /* var pointer yang menunjuk ke data yang bertipe int */

      x = 87;

      px = &x;    /* px diisi dengan alamat dari variabel x */
      y = *px;    /* y diisi dengan nilai yg 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);
}
```

Percobaan 6 : Program Mengakses & Mengubah Isi Suatu Variabel Pointer

```
#include <stdio.h>

main()
{
     float d = 54.5f, *pd;

     printf("Isi d mula-mula = %g\n", d);

     pd = &d;
     *pd += 10;

     printf("Isi d sekarang  = %g\n", d);
}
```

Percobaan 7 : Mengakses elemen array dengan pointer.

```
#include <stdio.h>

void main()
{
    int my_array[] = {1,23,17,4,-5,100};
    int *ptr;

    int i;
    ptr = &my_array[0];     /* var pointer ptr menunjuk ke elemen ke-
0 dari myarray */
    printf("\n\n");
    for (i = 0; i < 6; i++)
    {
      printf("my_array[%d] = %d   ",i,my_array[i]);
      printf("ptr + %d = %d\n",i, *(ptr + i));
    }
}
```

Percobaan 8 : Mengkopi string menggunakan pointer.

```
#include <stdio.h>

char strA[80] = "ABCDE";
char strB[80];

void main()
{

    char *pA;
    char *pB;
    puts(strA);
```

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

Percobaan 9 : Penggunaan Pointer untuk Bilangan Fibonaci

```
#include<stdio.h>
#include<stdlib.h>
#define     MAX 20

void main()
{
     int *fibo;
     int i;

     fibo = malloc(MAX * sizeof(int));

     *(fibo + 1) = 1;
     *(fibo + 2) = 1;

     for (i=3;i<=MAX;i++)
      *(fibo + i)= (*(fibo + i - 2) + *(fibo + i - 1));
     printf("%d Bilangan Fibonaci Pertama adalah : \n",MAX);
     for (i=1;i<MAX;i++)
            printf("%d",*(fibo+i));
}
```

Percobaan 10 : Penggunaan Struktur pada Konversi Koordinat Polar ke Koordinat Cartesian

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

struct polar {
      double r;
      double alpha;
};

struct kartesian {
      double x;
      double y;
};


void main()
{
      struct polar p1;
      struct kartesian k1;
```

```
      printf("Masukkan nilai r untuk koordinat polar : ");
      scanf("%lf",&p1.r);

      printf("Masukkan nilai alpha untuk koordinat polar : ");
      scanf("%lf",&p1.alpha);

      k1.x = p1.r * cos(p1.alpha);
      k1.y = p1.r * sin(p1.alpha);

      printf("Nilai  koordinat  kartesian  untuk  koordinat  polar  r=
%2.2lf alpha= %2.2lf adalah:\n",p1.r,p1.alpha);
      printf("x = %2.2lf  y = %2.2lf",k1.x,k1.y);
}
```

Percobaan 11 : Program Struktur dalam Array

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

struct dtnilai
{
      char nrp[10];
      char nama[20];
      double nilai;
};

struct dtnilai data[10];
int j=0;

void tambah_data()
{
      char jawab[2];
      while(1)
      {
            fflush(stdin);
            printf("NRP :");scanf("%s",&data[j].nrp);
            printf("Nama      :");scanf("%s",&data[j].nama);
            printf("Nilai Test :");scanf("%lf",&data[j].nilai);

            printf("Ada data lagi(y/t):"); scanf("%s",&jawab);

            if((strcmp(jawab,"Y")==0)||(strcmp(jawab,"y")==0))
            {
                  j++;continue;
            }
            else if ((strcmp(jawab,"T")==0)||(strcmp(jawab,"t")==0))
                  break;
      }
}

void tampil()
{
      int i;
      printf("Data Mahasiswa yang telah diinputkan :\n");
      printf("NRP\tNama\tNilai\n");

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

```
      {
            printf("%s\t%s\t%6.2f\n",data[i].nrp,data[i].nama,
data[i].nilai);
      }
}

void main()
{
      tambah_data();
      tampil();
}
```

Percobaan 12 : Mengakses Struktur dengan Pointer

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

void tampil(struct tag *p)

struct dtnilai
{
      char nrp[10];
      char nama[20];
      double nilai;
};

struct dtnilai my_struct;

void main()
{
    struct dtnilai *st_ptr;
    st_ptr = &my_struct;
    strcpy(my_struct.nrp,"01");
    printf("\n%s ",my_struct.nrp);

    strcpy(my_struct.nama,"Arini");
    printf("\n%s ",my_struct.nama);
    my_struct.nilai = 63.6;
    tampil(st_ptr);

}

void tampil(struct tag *p)
{
    printf("\n%s ", p->nrp);
    printf("%s ", p->nama);
    printf("%d\n", p->age);
}
```

E. LATIHAN

1. Terdapat array dengan tipe char yang berisi {'p', 'e', 'n', 's', 'i', 't'}, baliklah array tersebut menjadi tisnep.
2. Terdapat dua buah matrik dengan ordo n yang direpresentasikan dengan array dimensi dua, lakukan operasi penjumlahan, pengurangan dan perkalian.
3. Bagaimana output program di bawah ini ?

```
main() {
      int  count = 10, *temp, sum = 7;

      temp = &count;
      *temp = 32;
      temp = &sum;
      *temp = count;
      sum = *temp * 4;

      printf("count=%d, *temp=%d, sum=%d\n", count,*temp, sum );
   }
```

4. Masalah aritmatika polinom adalah membuat sekumpulan subrutin manipulasi terhadap polinom simbolis (symbolic Polynomial). Misalnya: P1 = $6x^8 + 8x^7 + 5x^5 + x^3 + 15$

   P2 = $3x^9 + 4x^7 + 3x^4 + 2x^3 + 2x^2 + 10$

   P3 = $x^2 + 5$

   Terdapat empat operasi aritmatika polinom dasar antara lain:

   a. Penambahan (P1 + P2 = $3x^9 + 6x^8 + 12x^7 + 5x^5 + 3x^4 + 3x^3 + 2x^2 + 25$)
   b. Pengurangan (P1 - P2 = $- 3x^9 + 6x^8 + 4x^7 + 5x^5 - 3x^4 - x^3 - 2x^2 + 5$)
   c. Perkalian (P1 * P3 = $6x^{10} + 8x^9 + 5x^7 + x^5 + 15x^2 + 30x^8 + 40x^7 + 25x^5 + 5x^3 + 75 = 6x^{10} + 8x^9 + 30x^8 + 45x^7 + 26x^5 + 5x^3 + 15x^2 + 75$)
   d. Turunan (P2' = $27x^8 + 28x^6 + 12x^3 + 6x^2 + 4x$)

   Representasikan bilangan polinom dengan array dan buatlah prosedur-prosedur yang melakukan kelima operasi aritmatika di atas.
5. Bilangan kompleks berbentuk a + bi, dimana a dan b adalah bilangan nyata dan $i^2 = -1$. Terdapat empat operasi aritmatika dasar untuk bilangan kompleks, yaitu:
   - Penambahan : (a+bi) + (c+di) = (a+c) + (b+d)i
   - Pengurangan : (a+bi) - (c+di) = (a-c) + (b-d)i
   - Perkalian : (a+bi) * (c+di) = (ac-bd) + (ad+bc)i
   - Pembagian : (a+bi) / (c+di) = [(ac+bd) / (a^2+b^2)] + [(bc-ad)/(c^2+d^2)]i

Tulis program yang membaca dua bilangan kompleks dan simbol operasi yang perlu dilakukan, kemudian lakukan operasi yang diminta. Gunakan struktur untuk merepresentasikan bilangan kompleks dan gunakan prosedur untuk implementasi tiap operasi.

F. LAPORAN RESMI

1. Untuk setiap listing program dari percobaan-percobaan di atas, ambil *capture* outputnya.
2. Tuliskan kesimpulan dari percobaan yang telah anda lakukan.