MODUL 7

STRUCT DAN POINTER

1) **Tujuan :**

   a) Mahasiswa memahami yang dimaksud dengan struct dan pointer

   b) Mahasiswa mampu mengimplementasikan struct dan pointer dalam struktur data

2) **Pointer**

Misalnya kita ingin membuat beberapa penunjuk ke blok penyimpan yang berisi integer.

Deklarasi pada C adalah:

int *IntegerPointer;

Tanda asterik (*) yang berada sebelum nama variable IntegerPointer menandakan ‘pointer pada suatu int’. Jadi deklarasi diatas berarti ‘definisikan sebuah tipe yang terdiri dari pointer bertipe integer yang bernama IntegerPointer’. Apabila didepannya ditambahkan typedef sebagai berikut :

Typedef int *IntegerPointer;

Berarti IntegerPointer merupakan suatu tipe pointer berbentuk integer. Apabila akan mendeklarasikan dua variable A dan B sebagai penunjuk ke bilangan integer :

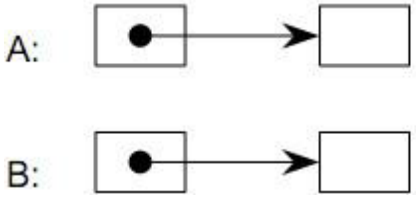
IntegerPointer A, B;

Berarti kompiler C akan berisi nilai dari variable A dan B yang ‘menunjuk ke integer’.

Untuk membuat beberapa penunjuk ke beberapa penyimpan integer yang kosong dan untuk membuat A dan B menunjuk tempat tersebut, digunakan prosedur dinamis untuk alokasi penyimpan yang disebut malloc, yang penulisannya sebagai berikut :

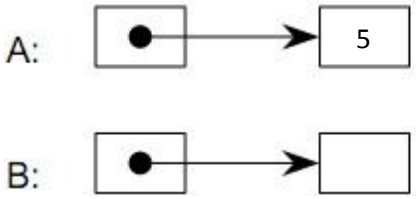
A = (IntegerPointer *) malloc (sizeof(int));

B = (int *) malloc (sizeof(int));



Misalnya kita akan menyimpan integer 5 pada blok penyimpan yang ditunjuk pointer pada variable A. Untuk menyimpan angka 5 pada blok penyimpan integer itu melalui pointer A, digunakan pernyataan :

*A = 5;



Linked list adalah salah satu struktur data yang paling fundamental. Linked list terdiri dari sejumlah kelompok elemen (*linked*) dengan urutan tertentu. Linked list sangat berguna untuk memelihara sekelompok data, semacam array, tetapi linked list lebih menguntungkan dalam beberapa kasus. Linked list lebih efisien dalam proses penyisipan (*insertion*) dan penghapusan (*deletion*). Linked list juga menggunakan pengalokasian penyimpan secara dinamis, dimana penyimpan dialokasikan pada saat waktu berjalan (*runtime*).

3) **Struktur**

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

Contoh sebuah struktur adalah informasi data tanggal, yang berisi: **tanggal, bulan dan tahun**

a) **Mendeklarasikan Struktur**

Contoh pendefinisian tipe struktur adalah sebagai berikut:

struct data_tanggal { int tanggal; int bulan; int tahun; };

yang mendefinisikan tipe struktur bernama data_tanggal, yang terdiri dari tiga buah elemen (field) berupa : **tanggal, bulan dan tahun**. Pendefnisian dan pendeklarasian struktur dapat juga ditulis sebagai berikut:

struct data_tanggal { int tanggal; int bulan; int tahun; } tgl_lahir;

Bentuk umum dalam mendefinisikan dan mendeklarasikan struktur adalah sebagai berikut :

struct nama_tipe_struktur { tipe field1; tipe field2; ... ... tipe fieldn; } variabel_struktur1, ... , variabel_strukturM;

Masing-masing tipe dari elemen struktur dapat berlainan. Adapun variabel_struktur1 sampai dengan variabel_strukturM menyatakan bahwa variabel struktur yang dideklarasikan bisa lebih dari satu. Jika ada lebih dari satu variabel, antara variabel struktur dipisahkan dengan tanda koma.

b) **Mengakses Elemen Struktur**

Elemen dari struktur dapat diakses dengan menggunakan bentuk :

variabel_struktur.nama_field

Antara variabel_struktur dan nama_field dipisahkan dengan operator titik (disebut operator anggota struktur). Contoh berikut merupakan instruksi untuk mengisikan data pada field tanggal

tgl_lahir.tanggal = 30;

LATIHAN PRAKTIKUM

Latihan 1 : Program merubah isi variabel melalui pointer

```
#include <stdio.h>

using namespace std;

int main(){
    int y, x = 87;
    int *px;
    float d = 54.5, *pd;

    px = &x;
    y = *px;

    printf("Alamat x = %p\n", &x);
    printf("Isi px = %p\n", px);
    printf("Isi x = %d\n", x);
    printf("Nilai yg ditunjuk oleh px =  %d\n", *px);
    printf("Nilai y = %d\n", y);

    printf("Isi d mula-mula = %g\n", d);
    pd = &d;
    *pd += 10;
    printf("Isi d sekarang = %f\n", d);

}

```

Latihan 2 : Penggunaan pointer pada bilangan Fibonacci

```
1  #include <bits/stdc++.h>
2  #define MAX 10
3
4  int *fibo;
5
6  using namespace std;
7
8  int main(){
9
10     fibo = (int *) malloc(MAX * sizeof(int));
11
12     *(fibo + 1) = 1;
13     *(fibo + 2) = 1;
14
15     for(int i= 3; i<= MAX; i++){
16         *(fibo + i) = (*(fibo + i - 2) + *(fibo +i - 1));
17     }
18
19     printf("BILANGAN FIBONACCI : ");
20     for(int i= 1; i<= MAX; i++){
21         printf("%d ", *(fibo + i));
22     }
23     printf("\n");
24 }
25
```

malloc adalah fungsi yang di gunakan untuk mengalokasikan memori pada suatu variabel pointer sehingga variabel tersebut menjadi seperti array. (fibo + 1) adalah sama seperti fibo[1] pada array.

Latihan 3 : Penggunaan struktur pada konversi koordinat polar ke koordinat catersian

```
#include <bits/stdc++.h>
#define aku using
#define ganteng namespace

aku ganteng std;

struct polar
{
    double r;
    double alpha;
}p1;

struct kartesian
{
    double x;
    double y;
}k1;


int main(){
    printf("Masukkan nilai r untuk koordinat polar : \n");
    scanf("%lf", &p1.r);
    printf("Masukkan nilai alpha untuk koordinat polar : \n");
    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 = %.2f
        alpha = %.2f adalah : \n", p1.r, p1.alpha);
    printf("x = %.2f y = %.2f\n", k1.x, k1.y);
}

```

Latihan 4 : Struct dalam Array

```
#include <bits/stdc++.h>

using namespace std;

struct dtnilai
{
    string nim;
    string nama;
    int nilai;
};

dtnilai data[10];
int j= 0;

int tambah_nilai(){
    char jawab;
    while(1){
        cout << "NIM : " ; cin >> data[j].nim;
        cout << "Nama : "; cin >> data[j].nama;
        cout << "Nilai : "; cin >> data[j].nilai ;

        cout << "Ada data lagi ? (y/t) : "; cin >> jawab;
        cout << endl;
        if(jawab == 'y' || jawab == 'Y') j++;
        else break;
    }
}

int tampil(){
    cout << "Data Mahasiswa yang diinputkan :\n\n";
    cout << "NIM \t Nama \t Nilai \n";

    for(int i= 0; i<= j; i++){
        cout << data[i].nim << " \t " << data[i].nama << " \t " << data[i].nilai << endl;
    }
}

int main(){
    tambah_nilai();
    tampil();
}
```

TUGAS PRAKTIKUM

1. Buat sebuah program yang dapat menyimpan data pasien rumah sakit yang berisi no pasien (otomatis sesuai dengan urutan dimasukkannya data), nama pasien, jenis penyakit, nama dokter, ruang pasien. Nama dokter yang dimasukkan harus terdaftar, dimana daftar nama dokternya adalah :
    - Michael
    - Carolline
    - John
    - Angga

    Selain nama dokter di atas, maka user diharuskan memasukkan nama dokter hingga valid. Data yang dapat disimpan paling banyak hanya 50 data.

2. Buatlah sebuah program yang mengimplementasikan linked list dengan menggunakan pointer untuk menghubungkan anggota satu dengan yang lainnya.