

Pengantar Struktur Data

S1 Informatika

Nafiatun Sholihah, S.Kom., M.Cs.

Kenapa struktur data penting
untuk dipahami
oleh programmer?



- Cara menyimpan dan mengatur data secara terstruktur pada system computer atau database sehingga lebih mudah diakses
- Secara teknis data disimpan dalam bentuk :
 - Angka
 - Huruf
 - Simbol

Dan diletakkan dalam kolom-kolom atau susunan tertentu.

- Penyusunan Data :
 - Node
 - Elemen dalam struktur data, setiap node berisi pointer ke node selanjutnya
 - Indeks
 - Objek dalam struktur data untuk mempercepat proses penyimpanan dan pencarian data

Perbedaan Tipe Data, Obyek Data & Struktur Data (1)

- **Tipe data** adalah jenis data yang mampu ditangani oleh suatu bahasa pemrograman pada komputer.
- Tiap-tiap bahasa pemrograman memiliki tipe data yang memungkinkan:
 - Deklarasi terhadap variabel tipe data tersebut
 - Menyediakan kumpulan operasi yang mungkin terhadap variabel bertipe data tersebut
 - Jenis obyek data yang mungkin
 - Contoh tipe data di C? Java? Pascal? .NET?



Perbedaan Tipe Data, Obyek Data & Struktur Data (2)

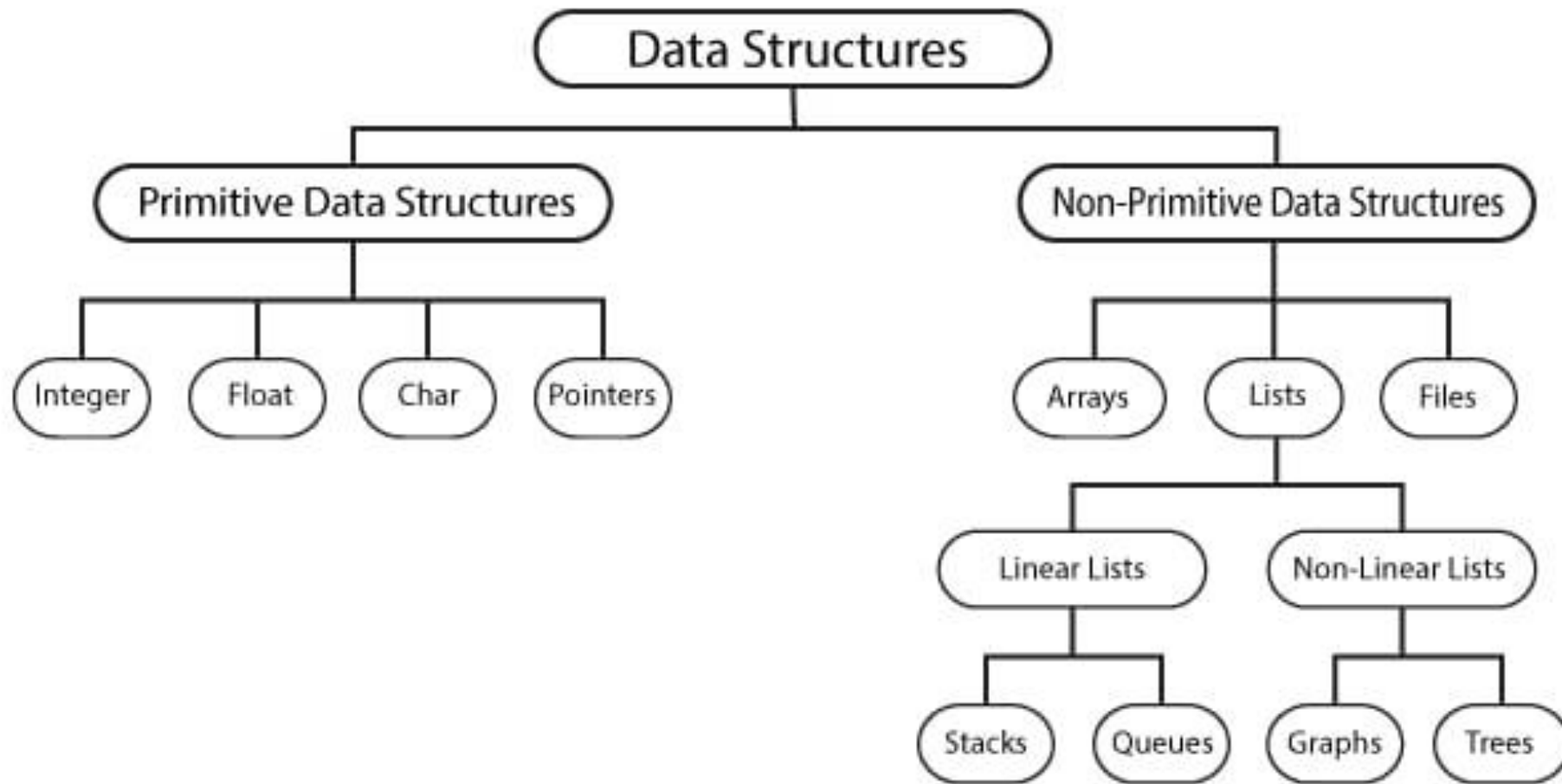
- **Obyek Data** adalah kumpulan elemen yang mungkin untuk suatu tipe data tertentu.
 - Mis: integer mengacu pada obyek data -32768 s/d 32767, byte 0 s/d 255, string adalah kumpulan karakter maks 255 huruf
- **Struktur Data** adalah cara penyimpanan dan pengorganisasian data-data pada memori komputer maupun file secara **efektif** sehingga dapat digunakan secara **efisien**, termasuk operasi-operasi di dalamnya.

- Di dalam struktur data kita berhubungan dengan 2 aktivitas:
 - Mendeskripsikan kumpulan obyek data yang sah sesuai dengan tipe data yang ada
 - Menunjukkan mekanisme kerja operasi-operasinya
 - Contoh: integer (-32768 s/d 32767) dan jenis operasi yang diperbolehkan adalah +, -, *, /, mod, ceil, floor, <, >, != dsb.
- Struktur data = obyek data + [operasi manipulasi data]

Hubungan SD dan Algoritma

- Dengan pemilihan struktur data yang baik, maka problem yang kompleks dapat diselesaikan sehingga **algoritma dapat digunakan secara efisien**, operasi-operasi penting dapat dieksekusi dengan **sumber daya yang lebih kecil**, memori **lebih kecil**, dan **waktu eksekusi yang lebih cepat**.
- Tidak semua struktur data baik dan sesuai. Contoh untuk problem data bank: pengupdate-an harus cepat, sedangkan penambahan/penghapusan data boleh lebih lambat.





- Set instruksi yang terbatas waktu dan harus berhenti setelah instruksi selesai dikerjakan.
- Ciri algoritma yang baik menurut **Donald E.Knuth**:
 - Input: ada minimal 0 input atau lebih
 - Output: ada minimal 1 output atau lebih
 - Definite: ada kejelasan apa yang dilakukan (tidak ambigu)
 - Effective: langkah yang dikerjakan harus efektif (bisa diimplementasikan)
 - Terminate: langkah harus dapat berhenti (stop) secara jelas

- Program harus jalan terus, tapi algoritma suatu saat harus berhenti
- Program dirancang untuk :
 - Menghemat waktu (*Time Complexity*)
 - Menghemat memory (*Space Complexity*)
 - Kebenaran dari program (tidak terjadi error saat di-run)

ADT (Abstract Data Type) atau Tipe Data Bentukan

- Bahasa pemrograman bisa memiliki tipe data:
 - Built-in : sudah tersedia oleh bahasa pemrograman tersebut
 - Tidak berorientasi pada persoalan yang dihadapi.
 - UDT : User Defined Type, dibuat oleh pemrogram.
 - Mendekati penyelesaian persoalan yang dihadapi
 - Contoh: record pada Pascal, struct pada C, class pada Java
 - ADT : Abstract Data Type
 - memperluas konsep UDT dengan menambahkan pengkapsulan atau enkapsulasi, berisi sifat-sifat dan operasi-operasi yang bisa dilakukan terhadap kelas tersebut.
 - Contoh: class pada Java

- Bahasa C memiliki tipe data numerik dan karakter (seperti int, float, char dan lain-lain). Disamping itu juga memiliki tipe data enumerasi dan structure. Bagaimana jika kita ingin membuat tipe data baru?
- Untuk pembuatan tipe data baru digunakan keyword **typedef**
- Bentuk umum:

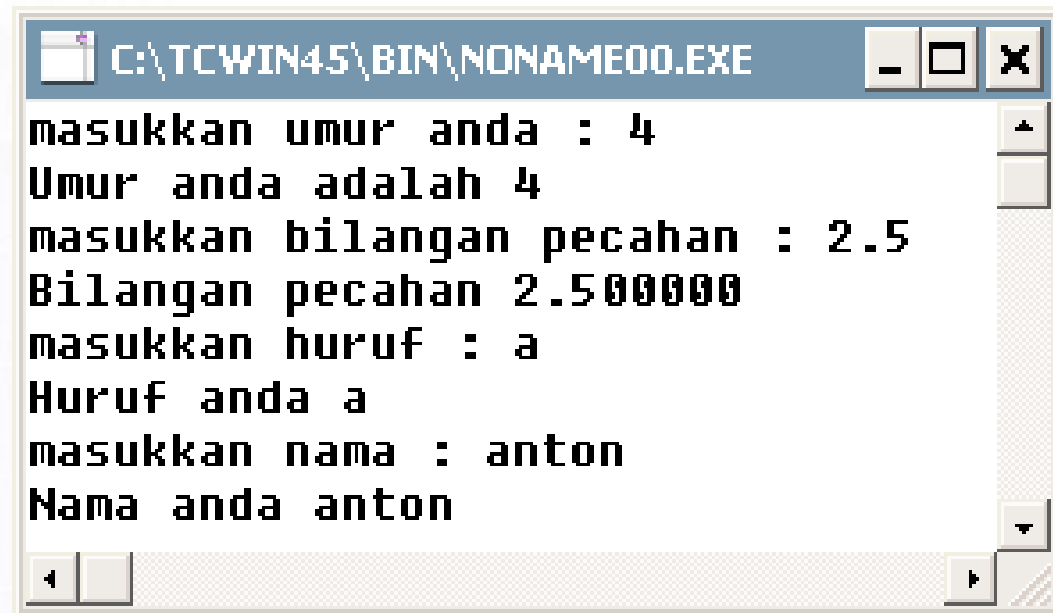
```
typedef <tipe_data_lama> <nama_tipe_data_baru>
```

- Contoh:

```
#include <stdio.h>
#include <conio.h>
typedef int angka;
typedef float pecahan;
typedef char huruf;
void main(){
    clrscr();
    angka umur;
    pecahan pecah;
    huruf h;

    huruf nama[10];
    printf("masukkan umur anda : ");scanf("%d",&umur);
    printf("Umur anda adalah %d",umur);
    printf("\nmasukkan bilangan pecahan : ");scanf("%f",&pecah);
    printf("Bilangan pecahan %f",pecah);
    printf("\nmasukkan huruf : ");h=getche();
    printf("\nHuruf anda %c",h);
    printf("\nmasukkan nama : ");scanf("%s",nama);
    printf("Nama anda %s",nama);
    getch();
}
```

Hasil Program



```
C:\TCWIN45\BIN\NONAME00.EXE
masukkan umur anda : 4
Umur anda adalah 4
masukkan bilangan pecahan : 2.5
Bilangan pecahan 2.500000
masukkan huruf : a
Huruf anda a
masukkan nama : anton
Nama anda anton
```



Buatlah Contoh Algoritma untuk :

1. Mencari Bilangan Terbesar dari 3 buah bilangan
2. Menentukan Bilangan Ganjil & Genap
3. Menghitung Luas Persegi
4. Menghitung akar-akar persamaan kuadrat
5. Mencari Nilai FPB (Euclidean)