

# **KONSEP STRUKTUR DATA DAN ADT (ABSTRACT DATA TYPE)**

# TYPE DATA

1. Type data merupakan sekumpulan nilai yang memiliki karakteristik sama dan merupakan jenis data yang ditangani oleh Bahasa pemrograman pada PC atau laptop.
2. Type data yang paling umum digunakan dalam Bahasa pemrograman C++, adalah sebagai berikut:
  - a. int → integer ( bilangan bulat) → ..., -2, -1, 0, 1, 2, ...
  - b. float → Float (nilai desimal) → 3.52, 4.2,....
  - c. double → Double ( sama dengan float tetapi range 2x lipat lebih besar) → ..., -3.121, 3.26, 5.7,...
  - d. char → Character (nilai berupa karakter) → ASCII character
  - e. Bool –Boolean (nilai yang tersimpan 1 atau 0, true atau false) → 0, 1

# PENGUNAAN TYPE DATA

1. Range merupakan nilai yang dapat disimpan
2. Memori: ukuran memori yang digunakan dalam penyimpanan nilai.

Misal:

Type data = 32 bit integer, maka:

→ Range:

a. Integer bertanda:  $(-2^{-31}) - (2^{-31}-1)$  atau  $(-2147483648) - 2147483647$

b. Integer tak bertanda:  $0 - (2^{-32}-1)$  atau  $0 - 4294967295$

→ Memori: 32 bit

# OBJEK DATA

1. Obyek Data merupakan kumpulan elemen untuk suatu tipe data tertentu.

Sebagai contoh:

→ Untuk integer yang mengacu pada obyek data:

-32768 s/d 32767, byte 0- 255.

2. String adalah kumpulan karakter maks 255 huruf.

# STRUKTUR DATA

1. Struktur Data merupakan cara penyimpanan dan pengorganisasian data-data pada memori komputer maupun file pada media penyimpanan secara efektif sehingga dapat digunakan secara efisien, termasuk operasi-operasi didalamnya.
2. Aktivitas yang berhubungan dengan struktur data:
  - a. Mendeskripsikan kumpulan obyek data yang sah sesuai dengan tipe data yang ada.
  - b. Menunjukkan mekanisme kerja operasi-operasinya;

Contoh:

integer (-32768 s/d 32767) dan jenis operasi yang diperbolehkan adalah +, -, \*, /, mod, ceil, floor, <, >, != dan sebagainya.



16 bit

# STRUKTUR DATA

Struktur data = obyek data + [operasi manipulasi]

- Pemilihan struktur data yang baik → masalah kompleks dapat diselesaikan dengan algoritma yang dapat digunakan secara efisien, operasi-operasi penting dapat dieksekusi dengan sumber daya yang lebih kecil, memori lebih kecil, dan waktu eksekusi yang lebih cepat.

# STRUKTUR DATA

- 1. Ciri algoritma yang baik menurut Donald E.Knuth:**
  - a. Input: ada minimal 0 input atau lebih
  - b. Ouput: ada minimal 1 output atau lebih
  - c. Definite: ada kejelasan apa yang dilakukan
  - d. Efective: langkah yang dikerjakan harus efektif
  - e. Terminate: langkah harus dapat berhenti (stop) secara jelas

# ADT

- ADT → abstract data type atau tipe data bentukan
- ADT merupakan sekumpulan objek dengan sekumpulan operasi.
- ADT merupakan abstraksi matematis → tidak ada penjelasan bagaimana sekumpulan operasi diimplementasikan dalam definisi ADT.
- ADT digunakan untuk memodelkan (**abstraksi**) → sekumpulan data yang ditemukan dalam sebuah permasalahan.



# ADT

- ADT merupakan sebuah definisi **TYPE** dan kumpulan operasi dasar (**operasi primitive**) yang beroperasi pada tipe struktur data.
- ADT merupakan sebuah tipe data bentukan yang didefinisikan dan dapat mengandung ADT lainnya.

contoh: ADT waktu → ADT tanggal dan ADT jam

# ADT

- **TYPE** diterjemahkan dalam bentuk **Data Type** yang disesuaikan dengan Bahasa pemrogramannya.

Misal : struct pada Bahasa C/C++, record pada Bahasa Pascal, class pada Bahasa java.

- **Operasi primitive** dalam konteks procedural diterjemahkan dalam bentuk prosedur atau fungsi.

# ADT

## Daftar primitive:

- Konstruktor : pembentuk tipe data
- Destruktor : menghancurkan bersama memorinya
- Selektor: mengakses komponen, umumnya nama method diawali **get()**
- Pengubah : mengubah komponen, umumnya nama method diawali dengan **set()**
- Validator : pemeriksaan apakah dapat membentuk tipe sesuai kriteria.
- Baca/Tulis : Interface dengan input/output device.
- Operator relasional : operator apakah lebih besar, lebih kecil, dan sebagainya
- Aritmatika : operasi aritmatika terhadap komponen atau tipenya.
- Konversi : konversi ke tipe dasar ataupun sebaliknya.

# ADT

Tipe operasi yang dilakukan pada ADT adalah:

## **1. Constructor**

Operasi ini digunakan untuk membuat *instance* ADT baru.

## **2. Mutator**

Operasi ini bersifat mengubah/memodifikasi nilai dari atribut ADT

## **3. Accessor**

Operasi ini bersifat mengembalikan nilai/informasi ADT.

# typedef

- Pembuatan type data baru → syntax typedef

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

Contoh kode:

```
typedef int Counter;  
Counter a;
```

- Kode tersebut mendefinisikan tipe data baru dengan nama 'Counter'.
- Tipe data 'Counter' merupakan nama alternatif dari tipe data integer.

# typedef

## Contoh kode typedef:

```
typedef char Pesan[255];  
Pesan salamPembuka;
```

- Kode tersebut mendefinisikan tipe data baru dengan nama 'Pesan'.
- Tipe data 'Pesan' merupakan *array of char* dengan banyak karakter = 255.

# typedef

## Contoh kode:

```
typedef int *IntPtrter;  
IntPtrter pointer;
```

- Penambahan tanda *asterisk*(\*) membuat *IntPtrter* menjadi *integer pointer*.
- Deklarasi variabel dengan tipe data *IntPtrter* membuat variabel ptr dianggap sebagai *integer pointer*

# typedef

## Contoh kode:

```
typedef struct {  
    int day;  
    int month;  
    int year;  
} Date;  
Date myBirthday;
```

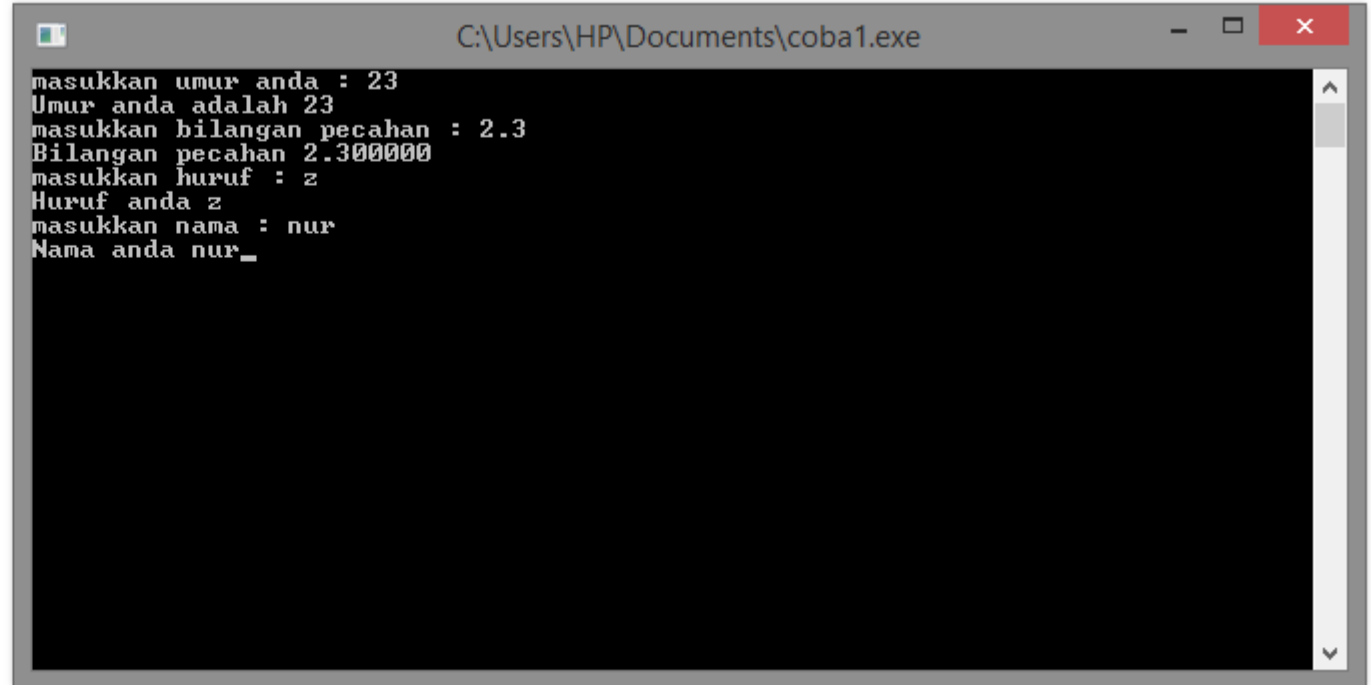
→ Tipe data Date merupakan bentuk abstraksi data dari Date/Tanggal di kehidupan nyata.

→ Anda dapat membuat variabel `myBirthday` yang menyimpan nilai berupa tanggal dengan menggunakan tipe data 'Date'.



# CONTOH PROGRAM

```
1  #include <stdio.h>
2  #include <conio.h>
3  //deklarasi type data
4  typedef int angka;
5  typedef float pecahan;
6  typedef char huruf;
7
8  // deskripsi program
9  int main()
10 {
11     system('clear');
12     angka umur;
13     pecahan pecah;
14     huruf h;
15     huruf nama[10];
16     printf("masukkan umur anda : ");scanf("%d",&umur);
17     printf("Umur anda adalah %d",umur);
18     printf("\nmasukkan bilangan pecahan : ");scanf("%f",&pecah);
19     printf("Bilangan pecahan %f",pecah);
20     printf("\nmasukkan huruf : ");h=getche();
21     printf("\nHuruf anda %c",h);
22     printf("\nmasukkan nama : ");scanf("%s",nama);
23     printf("Nama anda %s",nama);
24     getch();
25 }
```



```
C:\Users\HP\Documents\coba1.exe
masukkan umur anda : 23
Umur anda adalah 23
masukkan bilangan pecahan : 2.3
Bilangan pecahan 2.300000
masukkan huruf : z
Huruf anda z
masukkan nama : nur
Nama anda nur_
```

# STRUCT

- Struct adalah tipe data bentukan yang berisi kumpulan variabel-variabel yang bernaung dalam satu nama yang sama dan memiliki kaitan satu sama lain.
- Berbeda dengan array hanya berupa kumpulan variabel yang bertipe data sama, struct bisa memiliki variabel-variabel yang bertipe data sama atau berbeda, bahkan bisa menyimpan variabel yang bertipe data array atau struct itu sendiri.
- Variabel-variabel yang menjadi anggota struct disebut dengan elemen struct.

## Bentuk umum:

```
typedef struct {  
    tipe_data <nama_variable_1>;  
    tipe_data <nama_variable_2>  
    . . . .;  
    Tipe_data <nama_variable_n>  
}
```

# STRUCT

Contoh kode:

## a. Dengan typedef

```
typedef struct Mahasiswa {  
    char NIM[8];  
    char nama[50];  
    float ipk;  
};
```

## b. Tanpa typedef

```
struct {  
    char NIM[8];  
    char nama[50];  
    float ipk;  
} mhs;
```

# STRUCT

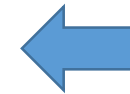
## Cara Penggunaan Struct:

- Penggunaan/pemakaian tipe data struct dilakukan dengan membuat suatu variabel yang bertipe data struct
- Pengaksesan elemen struct dilakukan secara individual dengan menyebutkan nama variabel struct diikuti dengan operator titik (.)

# STRUCT

Latihan :

```
1  #include<stdio.h>
2  #include<conio.h>
3  int main()
4  {
5      //deklarasi struct
6      system('clear');
7      struct mahasiswa
8      {
9          char nim[10];
10         char nama[10];
11         int nilai;
12     }mhs[10];
13     int i=0;
14     int n=0;
15     //deskripsi
16     printf("masukan jumlah data: ");
17     scanf("%d",&n);
18     for(i=1;i<=n;i++)
19     {
20         printf("\nnim = ");scanf("%s",mhs[i].nim);
21         printf("\nnama = ");scanf("%s",mhs[i].nama);
22         printf("\nnilai = ");scanf("%d",mhs[i].nilai);
23         printf("\n");
24     }
25     for(i=1;i<=n;i++)
26     {
27         printf("\nnim = %s",mhs[i].nim);
28         printf("\nnama = %s",mhs[i].nama);
29         printf("\nnilai = %d",mhs[i].nilai);
30         printf("\n");
31     }
32     getch();
33 }
```



Running program berikut ini dan bagaimana hasil yang didapat

TERIMA KASIH