Pointer dalam Bahasa C
Array statik vs. dinamis

IF2110/IF2111 – Algoritma dan Struktur Data
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Array statik vs. dinamis

***Array* statik**

- Ukuran *array* ditentukan pada saat ***compile-time***.
- Seumur hidupnya, ukuran *array* tersebut **tidak dapat diubah**.

***Array* dinamis**

- Ukuran *array* didefinisikan pada saat ***run-time*** dengan perintah alokasi memori.
- Ukuran **tidak diketahui** pada saat *compile-time*.
- Ukuran **dapat berubah** dengan dealokasi terlebih dahulu, dilanjutkan alokasi ulang.

Penggunaan array statik

1) Deklarasi (definisikan nama)

2) Inisialisasi nilai

Contoh array statik, Notasi Algoritmik

1) Deklarasi (definisikan nama)

```
t: array [0..99] of integer
```

2) Inisialisasi nilai

```
t[i] ← 9
```

Contoh array statik, Bahasa C

1) Deklarasi (definisikan nama)

```
int t[100];
```

2) Inisialisasi nilai

```
*(t+i) = 9; /* punya efek sama dengan: */
t[i] = 9;
```

Penggunaan array dinamis

1) Deklarasi (definisikan nama)

2) Alokasi (tentukan ukuran)

3) Inisialisasi nilai

4) Dealokasi (pengembalian memori)

Contoh array dinamis, Notasi Algoritmik

1) Deklarasi (definisikan nama)

```
t: array of integer
```

2) Alokasi (tentukan ukuran)

```
alokasi(t, 100)
{ t dialokasi sebesar 100, indeks
  0..99. }
```

3) Inisialisasi nilai

```
t[i] ← 9
```

4) Dealokasi (pengembalian memori)

```
dealokasi(t)
{ Setelah dealokasi, nama `t` boleh
  dialokasi lagi. }
```

Contoh array dinamis, Bahasa C

1) Deklarasi (definisikan nama)

```
int* t;
```

2) Alokasi (tentukan ukuran)

memory allocation

```
t = (int*) malloc (100 * sizeof(int));
/* T dialokasi sebesar 100, indeks
   0..99. */
```

3) Inisialisasi nilai

```
*(t+i) = 9;
/* punya efek sama dengan: */
t[i] = 9;
```

4) Dealokasi (pengembalian memori)

```
free(t);
/* Setelah dealokasi, nama `t` boleh
   dialokasi lagi. */
```

Beberapa catatan

Untuk memanfaatkan fungsi malloc & free, harus include **stdlib.h**

Proses alokasi bisa gagal sehingga setiap alokasi harus dicek apakah hasilnya gagal (NULL) atau tidak

Contoh:

```
int *arr;
arr = (int *) malloc (5*sizeof(int));
if (arr != NULL) {
    ...
} else {
    printf("Alokasi gagal\n");
}
```