

Struktur Berkait

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

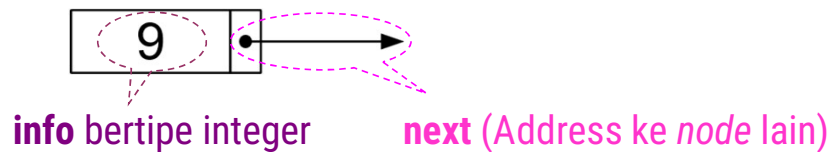
Definisi

Struktur berkait terdiri atas *node* yang terkait dengan *node* lain.

Node merupakan sebuah *tuple* yang terdiri atas dua bagian:

- 1) Sebuah nilai dengan tipe tertentu (info),
- 2) Sebuah penunjuk ke *node* lain (next).
Bisa jadi tidak menunjuk ke mana pun (NIL).

Ilustrasi:



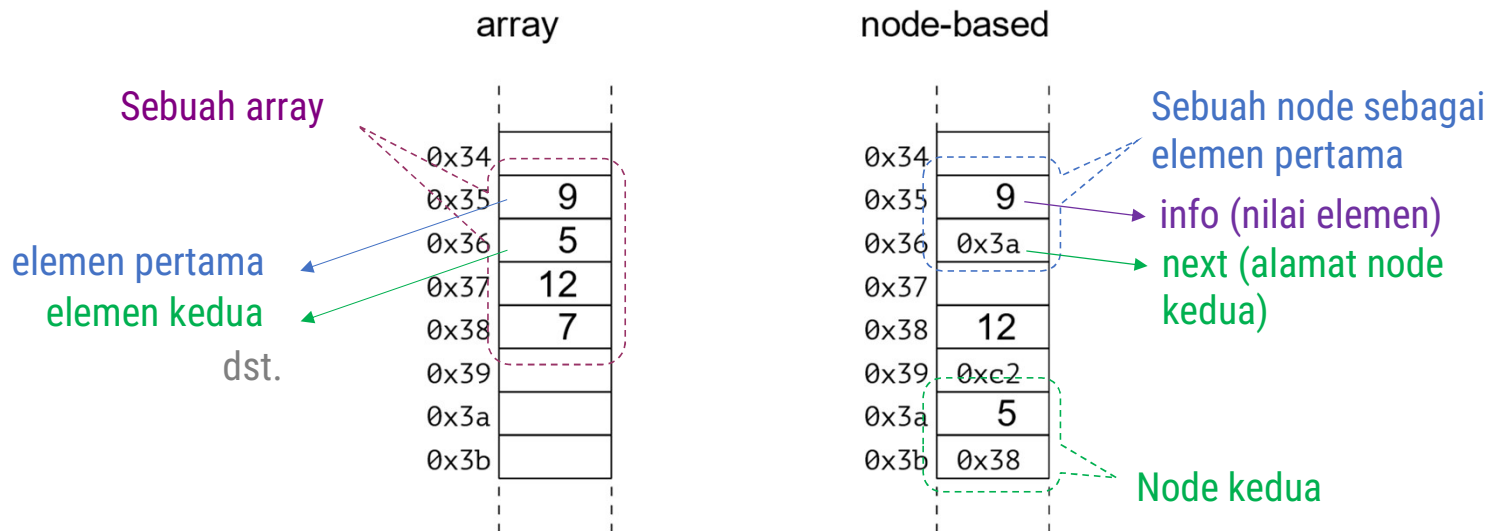
Hal ini memungkinkan penyimpanan elemen-elemen tanpa harus kontigu.

Disebut juga struktur *node-based*, *linked list*, atau *linear list*.

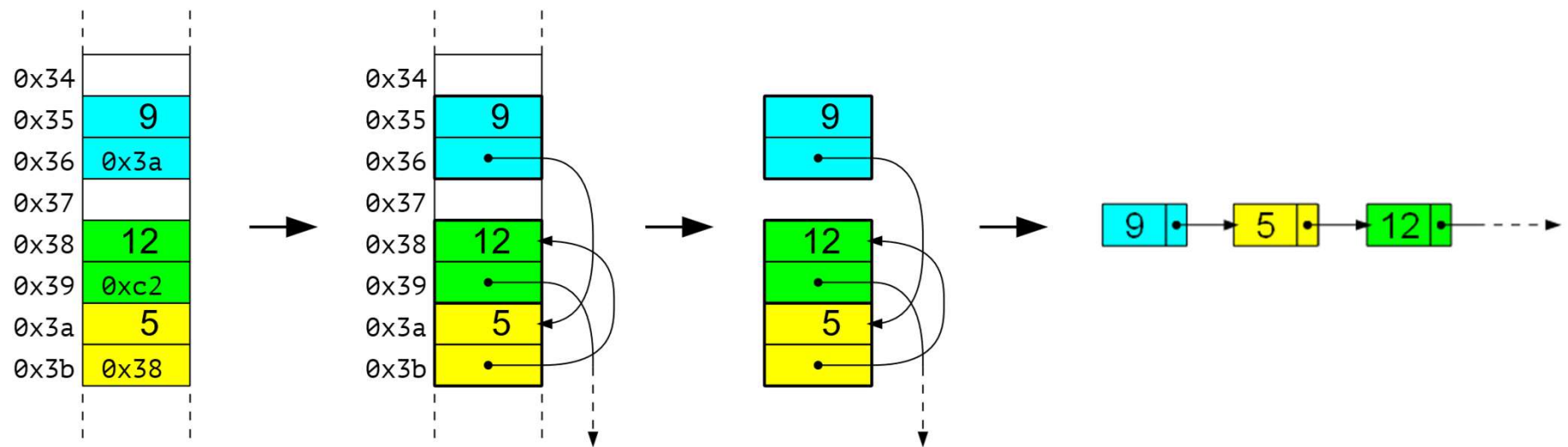
Memori fisik: array vs. node-based

- **Array:** elemen-elemen berada pada lokasi bersebelahan.
- **Node:** “next” mencatat alamat elemen berikutnya.

Contoh, dengan elemen bertipe *byte* (1 elemen mengisi 1 slot memori):



Ilustrasi struktur berkait



Karakteristik struktur data berbasis node

Memori dialokasi sesuai kebutuhan.

- Jika ada 3 elemen maka hanya perlu memori sebesar ukuran *node* $\times 3$.
- Berbeda dengan *array* yang, misalnya sudah dialokasikan 100 elemen maka menggunakan memori sebesar ukuran elemen $\times 100$ meskipun pada suatu waktu hanya 3 elemen yang efektif.

Ukuran memori per elemen menjadi lebih besar.

Ukuran elemen + ukuran *pointer*.

Secara umum mengorbankan efisiensi ruang (memori) demi efisiensi waktu.

(Tidak untuk semua jenis operasi.)

Node dalam Notasi Algoritmik

{ Deklarasi tipe bentukan }

type ElType: integer

type Address: pointer to Node

type Node: < info: ElType,
 next: Address >

{ Deklarasi variabel }

p1: Address

p2: Address

{ Inisialisasi dan penggunaan variabel }

p1 ← alokasi(9) { p1 menunjuk ke Node dengan info=9 dan next=NIL }

p2 ← alokasi(5) { p2 menunjuk ke Node dengan info=5 dan next=NIL }

p1↑.next ← p2 { Address next pada p1 menunjuk ke node yang ditunjuk p2 }

Node dalam Bahasa C (deklarasi tipe bentukan)

```
/* node.h */
#ifndef NODE_H
#define NODE_H

typedef int ElType;
typedef struct node* Address;
typedef struct node {
    ElType info;
    Address next;
} Node;

#define INFO(p) (p)->info
#define NEXT(p) (p)->next
Address newNode(ElType val);

#endif
```

```
/* node.c */
#include "node.h"
#include <stdlib.h>

Address newNode(ElType val) {
    Address p = (Address)
        malloc(sizeof(Node));
    if (p!=NULL) {
        INFO(p) = val;
        NEXT(p) = NULL;
    }
    return p;
}
```

Node dalam Bahasa C (contoh penggunaan)

```
/* Deklarasi variabel */
```

```
Address p1, p2;
```

```
/* Inisialisasi dan penggunaan variabel */
```

```
p1 = newNode(9); /* p1 menunjuk ke Node dengan info=9 dan next=NIL */
```

```
p2 = newNode(5); /* p2 menunjuk ke Node dengan info=5 dan next=NIL */
```

```
NEXT(p1) = p2; /* Address next pada p1 menunjuk ke node yang ditunjuk p2 */
```