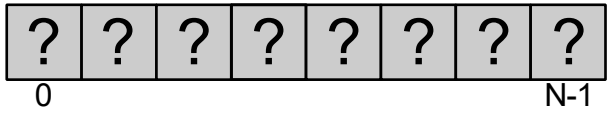
Implementasi ADT List dengan Array (elemen tersebar)

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

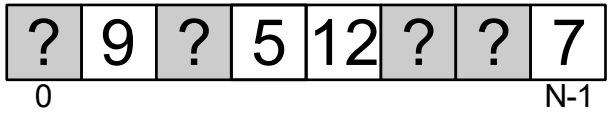
Elemen list tersebar dalam array (alt-3)

Ide: dengan menggunakan nilai khusus (*mark*), elemen sebenarnya tidak perlu disimpan secara kontigu.

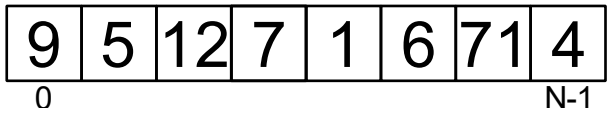
Contoh *array* kosong:



Contoh *array* terisi sebagian:



Contoh *array* penuh:



Array kosong harus diinisialisasi dengan *mark*.

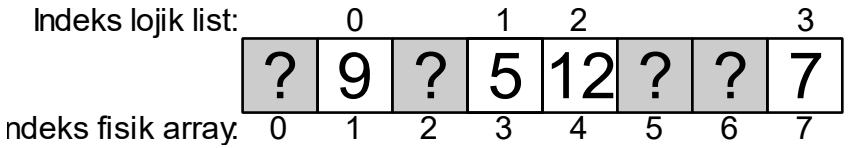
Mengunjungi elemen berikutnya harus cek apakah = *mark* (tidak bisa langsung indeks+1).

*contoh: kapasitas=N

Operasi-operasi pada alt-3

isEmpty: *true* jika semua elemen bernilai khusus (*mark*).

indexOf: *skip* elemen-elemen yang bernilai *mark*.



length: traversal, cacah yang bukan *mark*.

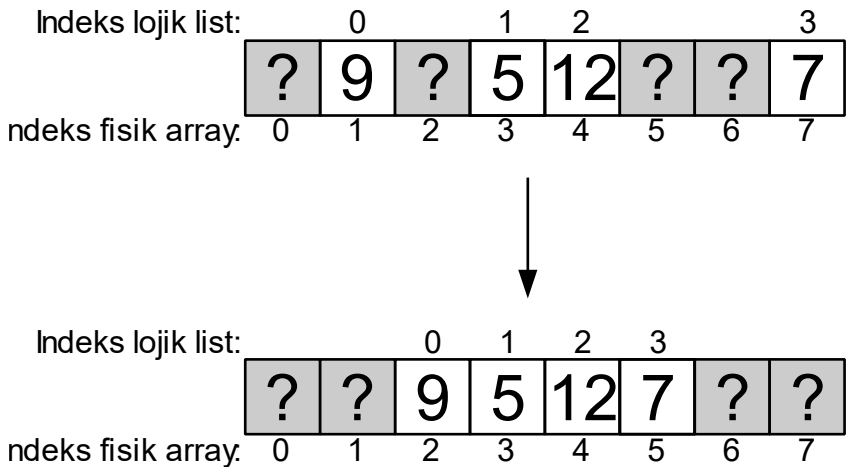
getElmt, **setElmt**: harus dilakukan mulai indeks fisik = 0 sambil menghitung indeks lojik.

Pola **traversal**: harus dilakukan dari 0 .. kapasitas-1, skip elemen-elemen bernilai *mark*.

insert: (sulit??) Paling sederhana: lakukan “defragmen” sebelum insert.

delete: set elemen yang hendak dihapus menjadi bernilai *mark*.

“Defragmen” (memampatkan)



Contoh algoritma: length

```
function length(l: List) → integer
{ ... }
KAMUS LOKAL
  ctr,i: integer
ALGORITMA
  ctr ← 0
  i traversal [0..CAPACITY-1]
    if i≠MARK then
      ctr ← ctr+1
  → ctr
```

Analisis efisiensi 5 alternatif

alt-1a vs. alt-2a vs. alt-1b vs. alt-2b vs. alt-3

Dari segi penggunaan memori, semuanya sama (sesuai ukuran alokasi di awal).

Dari segi waktu, mana yang lebih baik untuk operasi-operasi berikut:

- insert first
- insert last
- insert di tengah
- delete first
- delete last
- delete di tengah

untuk kasus-kasus *best case*, *worst case*, dan rata-rata?