Skema Pemrosesan List Berkait

IF1210 – Algoritma dan Pemrograman 1
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Skema dasar pemrosesan List berkait

- Traversal
- Pencarian sekuensial (*search*)

Skema traversal

Digunakan untuk memroses setiap elemen List dengan cara yang sama.

Mekanisme: mengunjungi setiap elemen List secara berurutan dimulai dari elemen pertama hingga elemen terakhir.

Jenis traversal:

- Dasar (tanpa perlakuan khusus untuk List kosong)
- Dengan perlakuan khusus untuk List kosong
- Untuk List yang tidak kosong

Skema traversal dasar

```
procedure SKEMAlistTraversal1(input l: List)
{ I.S. List l terdefinisi, mungkin kosong. }
{ F.S. Semua elemen List l "dikunjungi" dan telah diproses. }
{ Traversal sebuah List linier tanpa pemrosesan khusus untuk List kosong. }
KAMUS LOKAL
  p: Address                              { address untuk traversal, type terdefinisi }
  procedure proses(input p: Address) { pemrosesan elemen ber-address p }
  procedure inisialisasi                  { aksi sebelum proses dilakukan }
  procedure terminasi                     { aksi sesudah semua pemrosesan elemen selesai }
ALGORITMA
  inisialisasi
  p ← l
  while (p ≠ NIL) do
    proses(p)
    p ← p↑.next
  terminasi
```

Skema traversal dengan penanganan list kosong

```
procedure SKEMAlistTraversal2(input l: List)
{ I.S. List l terdefinisi, mungkin kosong. }
{ F.S. Semua elemen List l "dikunjungi" dan telah diproses. }
{ Traversal sebuah List linier dengan pemrosesan khusus untuk List kosong. }
KAMUS LOKAL
  { SAMA SEPERTI SKEMA 1, tidak ditulis untuk menghemat tempat }
ALGORITMA
  if l = NIL then
    output("List kosong")
  else
    inisialisasi
    p ← l
    repeat
      proses(p)
      p ← p↑.next
    until (p = NIL)
    terminasi
```

Skema traversal untuk List tidak kosong

```
procedure SKEMAlistTraversal3(input l: List)
{ I.S. List l terdefinisi, tidak kosong: minimal mengandung satu elemen. }
{ F.S. Semua elemen List l "dikunjungi" dan telah diproses. }
{ Traversal untuk List linier yang sudah dipastikan tidak kosong. }
KAMUS LOKAL
  { SAMA SEPERTI SKEMA 1 dan 2, tidak ditulis untuk menghemat tempat }
ALGORITMA
  inisialisasi
  p ← l
  iterate
    proses(p)
  stop (p↑.next = NIL)
    p ←  p↑.next
  terminasi
```

Skema pencarian sekuensial

List linier tidak memungkinkan *binary search*.

Mekanisme: mengunjungi elemen-elemen List secara berurutan dimulai dari elemen pertama hingga ditemukan elemen yang memenuhi syarat pencarian, atau semua elemen telah dikunjungi.

Jenis skema pencarian:

- Dengan boolean
- Tanpa boolean

Pencarian berdasarkan nilai elemen (1)

```
procedure SKEMAlistSearch1(input l: List, input x: ElType,
                           output p: Address, output found: boolean)
{ I.S. List linier l sudah terdefinisi dan siap dikonsultasi, x terdefinisi }
{ F.S. p adalah address di mana x ditemukan, p = NIL jika tidak ketemu }
{      found menyatakan apakah nilai x yang dicari ditemukan }
{ Menggunakan skema search dengan boolean }
KAMUS LOKAL
  -
ALGORITMA
  p ← l
  found ← false
  while (p ≠ NIL) and (not found) do
    if (x = p↑.info) then
      found ← true
    else
      p ← p↑.next
  { p = NIL or found}
  { Jika found maka p = Address dari nilai yg dicari }
  { p = NIL jika nilai tidak ditemukan}
```

Pencarian berdasarkan nilai elemen (2)

```
procedure SKEMAlistSearch2(input l: List, input x: ElType,
                           output p: Address, output found: boolean)
{ I.S. List linier l sudah terdefinisi dan siap dikonsultasi, x terdefinisi }
{ F.S. p adalah address di mana x ditemukan, p = NIL jika tidak ketemu }
{      found menyatakan apakah nilai x yang dicari ditemukan }
{ Menggunakan skema search tanpa boolean }
KAMUS LOKAL
  -
ALGORITMA
  p ← l
  if p = NIL then  { List kosong }
    found ← false
  else { List tidak kosong }
    while (p↑.next ≠ NIL) and (p↑.info ≠ x) do
      p ← p↑.next
    { p↑.next = NIL or p↑.info = x }
    found ← (p↑.info = x)
    if (not found) then
      p ← NIL
```

Pencarian elemen yang memenuhi kondisi

```
procedure SKEMAlistSearchX(input l: List, input kondisi(p): boolean,
                           output p: Address, output found: boolean)
{ I.S. List linier l sudah terdefinisi dan siap dikonsultasi, kondisi(p) adalah
       suatu ekspresi boolean yang merupakan fungsi dari elemen beralamat p }
{ F.S. p adalah address di mana kondisi(p) terpenuhi, p = NIL jika tidak ketemu }
{      found menyatakan apakah ada p yang memenuhi kondisi(p) }
KAMUS LOKAL
  -
ALGORITMA
  p ← l
  found ← false
  while (p ≠ NIL) and (not found) do
    if kondisi(p) then
      found ← true
    else
      p ←  p↑.next
  { p = NIL or found}
  { Jika found maka p adalah elemen List dengan kondisi(p) true }
```