**TUGAS PENDAHULUAN**

**MODUL 5**

**“Single Linked List (Bagian Kedua)“**

****

**Disusun Oleh:**

**Isabelle Putri Ardini - 2311104030**

**SE-07-01**

**Dosen:**

**Yudha Islami Sulistya**

**PROGRAM STUDI SOFTWARE ENGINEERING**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

**Soal Tugas Pendahuluan**

**Soal 1:** Mencari Elemen Tertentu dalam SLL

**Deskripsi Soal:** Buatlah program yang mengizinkan pengguna memasukkan 6 elemen interger ke dalam list. Implementasikan funtion **searchElement** untuk mencari apakah sebuah nilai tertentu ada dalam list.

**Instruksi**

1. Minta pengguna untuk memasukkan nilai yang ingin dicari.
2. Jika nilai ditemukan, tamplkan alamat dan posisi dalam angka (contoh: urutan ke 4) pada list tersebut.
3. Jika nilai tidak ditemukan, tampilkan pesan bahwa elemen tersebut tidak ada dalam list tersebut.

**NB:**

1. Gunakan pendekatan linier search untuk mencari elemen.

**Sub-Program:**

**Function** searchElement( L : **list**, i : **integer**)

{ I.S. List tidak kosong.

F.S. Menampilkan alamat dan posisi elemen i jika ditemukan}

Dictionary

current: address

position: int

Algorithms

current <- L.head

position <- 1

//melakukan perulangan selama i belum ditemukan dan posisi current belum berada pada

akhir list

While ......

//seiring pointer (current) bergerak, position bertambah

. . . . .

//lakukan perpindahan current

. . . . .

endwhile

//jika i ditemukan maka tampilkan alamat dan posisi

if....

output(...)

//jika tidak ditemukan maka tampilkan pesan yang menyatakan hal tsb

else...

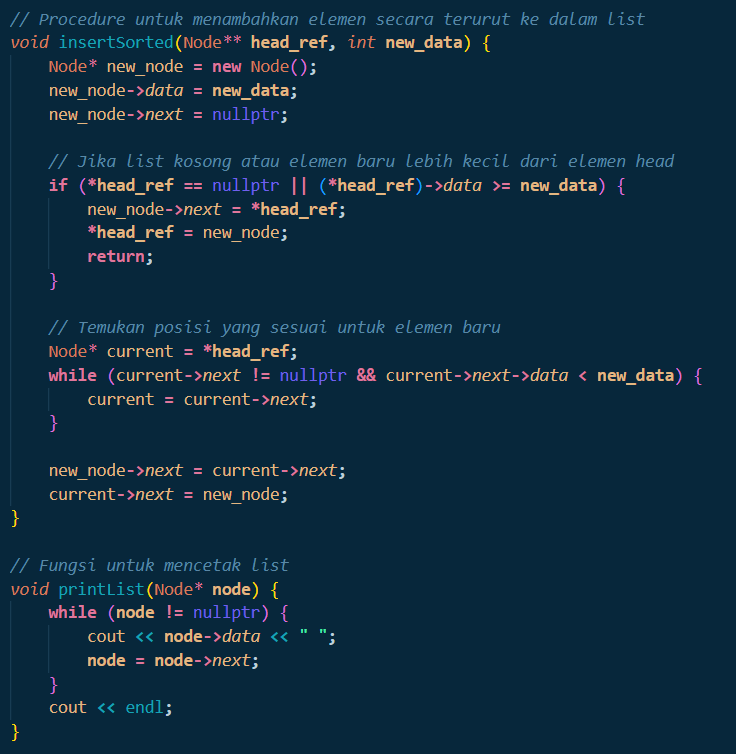
output(...)

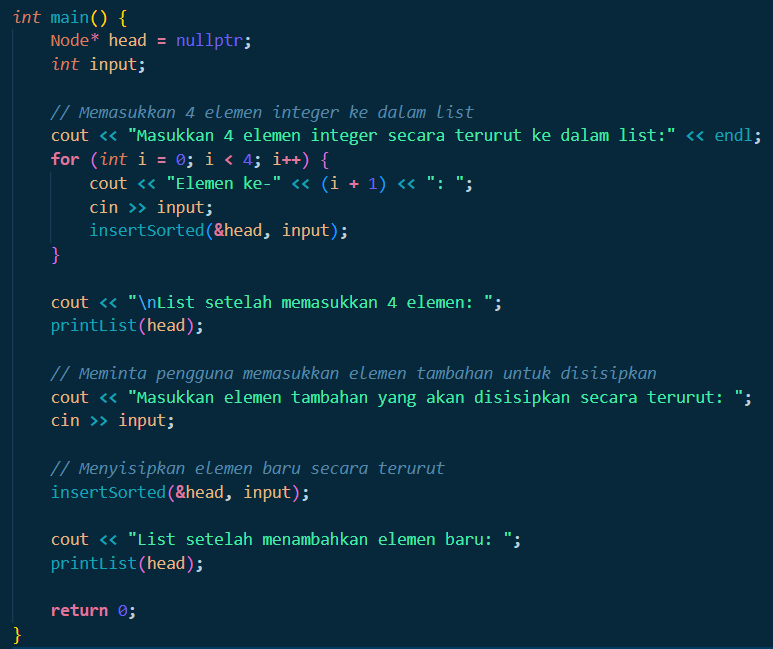
endif

endfunction

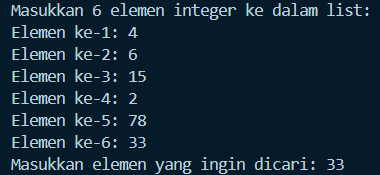
**Program:**

****

****

****

**Hasil Running:**

****

**Soal 2:** Menggunakan List Menggunakan Bubble Sort

**Deskripsi Soal:** Buatlah program yang mengizinkan pengguna memasukkan 5 elemen integer ke dalam list. Implementasikan procedure bubbleSortList untuk mengurutkan elemen-elemen dalam list dari nilai terkecil ke terbesar.

**Instruksi**

1. Setelah mengurutkan, tampilkan elemen-elemen list urutan yang benar.

**Langkah-langkah Bubble Sort pada SLL**

1. Inisialisasi:

* Buat pointer current yang akan digunakan untuk menelusuri list.
* Gunakan variabel boolean swapped untuk mengawasi apakah ada pertukaran yang dilakukan pada iterasi saat ini.

1. Traversing dan Pertukaran:

* Lakukan iterasi berulang sampai tidak ada pertukaran yang dilakukan:
* Atur swapped ke false di awal setiap iterasi.
* Set current ke head dari list.
* Selama current.next tidak null (masih ada node berikutnya):
* Bandingkan data pada node current dengan data pada node current.next.
* Jika data pada current lebih besar dari data pada current.next, lakukan pertukaran:
* Tukar data antara kedua node (bukan pointer).
* Set swapped menjadi true untuk menunjukkan bahwa ada pertukaran yang dilakukan.
* Pindahkan current ke node berikutnya (current = current.next).

1. Pengulangan:

* Ulangi langkah 2 sampai tidak ada lagi pertukaran yang dilakukan (artinya list sudah terurut).

**Contoh Proses Bubble Sort**

* List awal : 4 – 2 – 3 – 1 dan akan melakukan sorting membesar / ascending
* Iterasi pertama:
* Bandingkan 4 dan 2: 4 > 2, lakukan penukaran, 2 – 4 – 3 – 1
* Bandingkan 4 dan 3: 4 > 3, lakukan penukaran, 2 – 3 – 4 – 1
* Bandingkan 4 dan 1: 4 > 1, lakukan penukaran, 2 – 3 – 1 – 4
* Kondisi list di akhir iterasi: 2 – 3 – 1 – 4
* Iterasi kedua:
* Bandingkan 2 dan 3: 2 < 3, tidak terjadi penukaran
* Bandingkan 3 dan 1: 3 > 1, lakukan penukaran, 2 – 1 – 3 – 4
* Bandingkan 3 dan 4: 3 < 4, tidak terjadi penukaran
* Kondisi list di akhir iterasi: 2 – 1 – 3 – 4
* Iterasi ketiga:
* Bandingkan 2 dan 1: 2 > 1, lakukan penukaran, 1 – 2 – 3 – 4
* Bandingkan 2 dan 3: 2 < 3, tidak terjadi penukaran
* Bandingkan 3 dan 4 : 3 < 4, tidak terjadi penukaran
* Kondisi list di akhir iterasi: 1 – 2 – 3 – 4

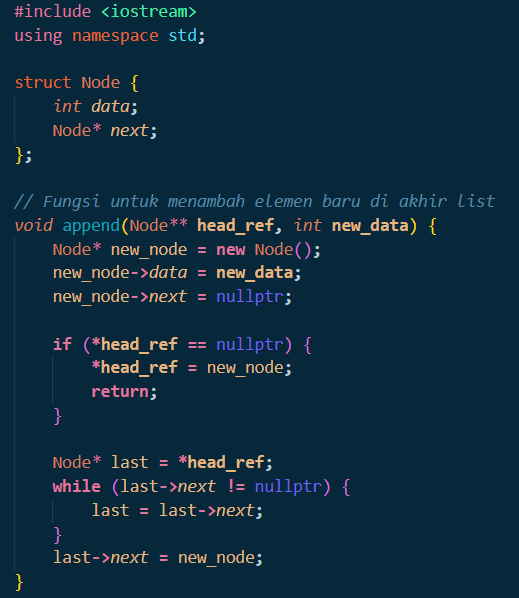
**Sub-Program:**

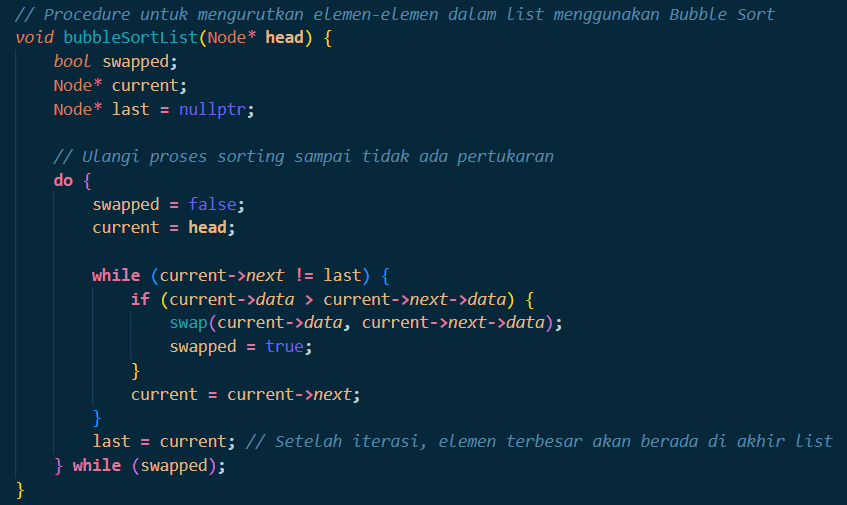
**Procedure** bubbleSort( **in/out** L : **list** )

{ I.S. List tidak kosong.

F.S. elemen pada list urut membesar berdasarkan infonya}

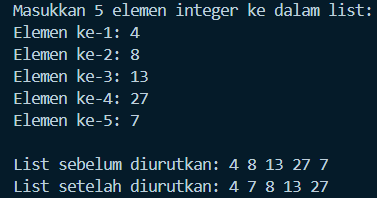
**Program:**

****

****

****

**Hasil Running:**

****

**Soal 3:** Menambahkan Elemen Secara Terurut

**Deskripsi Soal:** Buatlah program yang mengizinkan pengguna memasukkan 4 elemen

integer ke dalam list secara manual. Kemudian, minta pengguna memasukkan elemen

tambahan yang harus ditempatkan di posisi yang sesuai sehingga list tetap terurut.

**Instruksi**

1. Implementasikan procedure insertSorted untuk menambahkan elemen baru ke dalam list yang sudah terurut.
2. Tampilkan list setelah elemen baru dimasukkan.

**Sub-Program:**

**Procedure** insertSorted( **in/out** L : **list**, **in** P : **address**)

{ I.S. List tidak kosong.

F.S. Menambahkan elemen secara terurut}

Dictionary

Q, Prev: address

found: bool

Algorithms

Q <- L.head

found <- false

//melakukan perulangan selama found masih false dan Q masih menunjuk elemen pada list

While ......

//melakukan pengecekan apakah info dari elemen yang ditunjuk memiliki nilai lebih

kecil dari pada P

if ....

//jika iya maka Prev diisi elemen Q, dan Q diisi elemen setelahnya

....

//jika tidak maka isi found dengan nilai ’true’

else

. . .

Endif

//lakukan perpindahan Q

....

endwhile

//melakukan pengecekan apakah Q elemen head

if ....

//jika iya, maka tambahkan P sebagai head

....

//melakukan pengecekan apakah Q berisi null (sudah tidak menunjuk elemen pada list

else if ...

//jika iya, maka tambahkan P sebagai elemen terakhir

...

//jika tidak keduanya, maka tambahkan P pada posisi diantara Prev dan Q

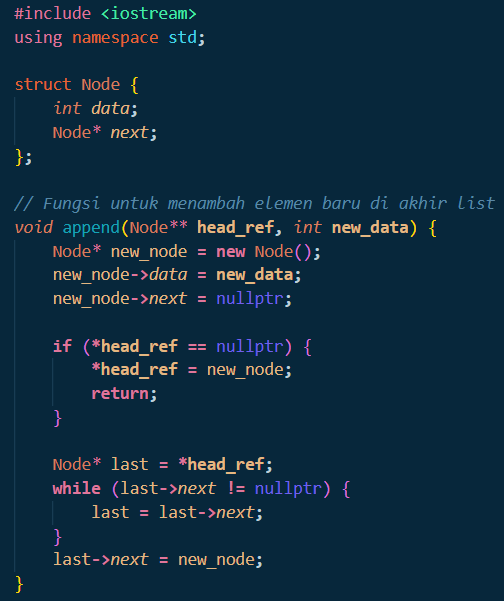
else

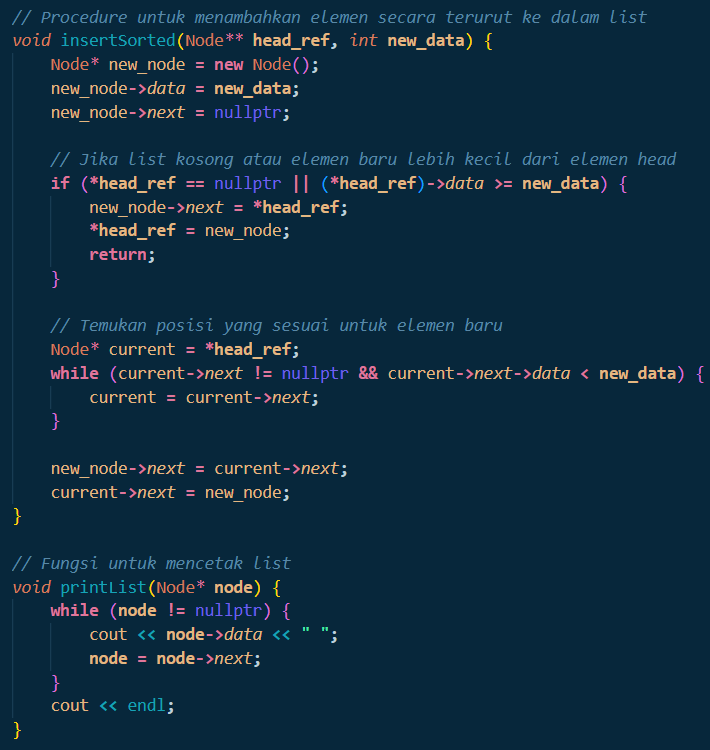
....

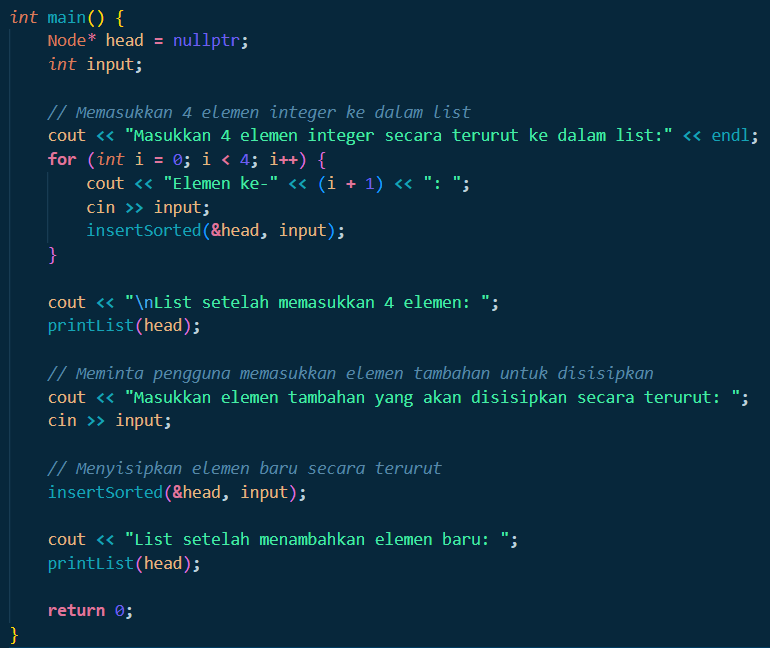
endif

endprocedure

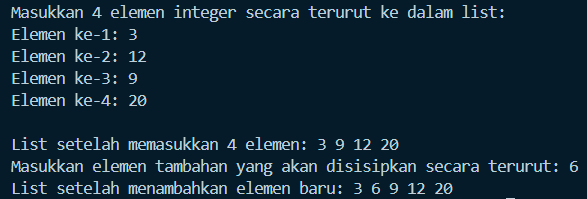
**Program:**

****

****

****

**Hasil Running:**

****