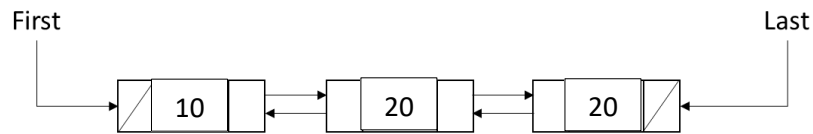


JURNAL PRAKTIKUM MODUL 4
MK STRUKTUR DATA 2019/2020-2
Double Linked-List dengan pointer First dan Last

Sebuah Double Linked List dengan pointer kepala First dan Last digunakan untuk menyimpan data berupa integer. Berikut ilustrasinya:



Buatlah ADT nya (DLL_First_Last.h, DLL_First_Last.cpp, dan Test_DLL_First_Last.cpp)!

A. Spesifikasi (Silakan ditulis ulang dalam Bahasa C++)

```
Type infotype : int
Type address : pointer to elmList
Type elmList <
    info : infotype
    next : address
    prev : addrees
>
Type List <
    First : address
    Last : address
>

Procedure createList (input/ouput L : List)
Function createElemen (dataBaru: infotype) → address
Procedure insertFirst (input/ouput L : List, input P : address)
Procedure insertLast (input/ouput L : List, input P : address)
Procedure InsertAfter (input Prec : address, P : address);
Procedure insertAscending (input/ouput L : List, input dataBaru : infotype)
Procedure deleteFirst (input/ouput L : List, output P : address)
Procedure deleteLast (input/ouput L : List, output P : address)
Procedure deleteAfter (input Prec: address, output P : address)
Procedure deleteElm (input/ouput L : List, input dataHapus : infotype)
Procedure printList (input L : List);
Function findElemen (L: List, dataDicari: infotype) → integer
Function frequencyofElm (L: List, dataDicari: infotype) → integer
```

JURNAL PRAKTIKUM MODUL 4
MK STRUKTUR DATA 2019/2020-2
Double Linked-List dengan pointer First dan Last

B. Implementasi (Silakan ditulis ulang dalam Bahasa C++)

Procedure createList (**input/ouput** L : List)

{ IS. –

FS. Terbentuk sebuah list di mana, first dan last dari L bernilai NIL. }

Kamus

Algoritma

First (L) \leftarrow NIL

Last (L) \leftarrow NIL

Function createElemen (dataBaru: infotype) \rightarrow address

{ Return alamat alokasi memori sebuah elmList yang berisi dataBaru. }

Kamus

P: address

Algoritma

alokasi (P)

info (P) \leftarrow dataBaru

next (P) \leftarrow NIL

prev (P) \leftarrow NIL

\rightarrow P

Procedure insertLast (**input/ouput** L : List, **input** P : address)

{ IS. Terdefinisi pointer P berisi alamat elmList, dan sebuah list L (L mungkin kosong).

FS. elmList yang ditunjuk oleh P ditambahkan ke dalam list sebagai elemen terakhir. }

Kamus

Algoritma

if First (L) = NIL then

First (L) \leftarrow P

Last (L) \leftarrow P

else

next (Last(L)) \leftarrow P

prev (P) \leftarrow Last(L)

Last (L) \leftarrow P

Procedure InsertAfter (**input** Prec : address, P : address);

{ IS. Terdefinisi pointer Prec dan P berisi alamat elmList. Prec \neq Last(L).

FS. elmList yang ditunjuk oleh P ditambahkan ke dalam list setelah elmList yang ditunjuk oleh Prec. }

Kamus

Algoritma

next (P) \leftarrow next (Prec)

```
prev(next(Prec))  $\leftarrow$  P
next (Prec)  $\leftarrow$  P
prev (P)  $\leftarrow$  Prec
```

Procedure deleteFirst (input/ouput L : List, output P : address)

```
{  IS. Terdefinisi sebuah list L (L tidak kosong dan mungkin berisi satu elemen).
   FS. P berisi alamat elmList yang pertama, elmList yang ditunjuk oleh P dihapus dari list
}
```

Kamus

Algoritma

```
P  $\leftarrow$  First (L)
if next (First (L)) = NIL then
    First (L)  $\leftarrow$  NIL
    Last (L)  $\leftarrow$  NIL
else
    First (L)  $\leftarrow$  next (First (L))
    prev (First(L))  $\leftarrow$  NIL
    next (P)  $\leftarrow$  NIL
```

Procedure deleteAfter (input Prec: address, output P : address)

```
{  IS. Terdefinisi pointer Prec berisi alamat elmList. Prec  $\neq$  Last(L). next(Prec)  $\neq$  Last(L).
   FS. P berisi alamat elmList setelah Prec, elmList yang ditunjuk oleh P dihapus dari list }
```

Kamus

Algoritma

```
P  $\leftarrow$  next (Prec)
next (Prec)  $\leftarrow$  next (P)
prev (next(P))  $\leftarrow$  prec
next (P)  $\leftarrow$  NIL
prev (P)  $\leftarrow$  NIL
```

Procedure printList (input L : list);

```
{  IS. Terdefinisi sebuah list L
   FS. Menampilkan semua info elmList di list. }
```

Kamus

Algoritma

```
P  $\leftarrow$  First (L)
while P  $\neq$  NIL do
    output (info (P))
    P  $\leftarrow$  next (P)
```

JURNAL PRAKTIKUM MODUL 4
MK STRUKTUR DATA 2019/2020-2
Double Linked-List dengan pointer First dan Last

--

JURNAL PRAKTIKUM MODUL 4
MK STRUKTUR DATA 2019/2020-2
Double Linked-List dengan pointer First dan Last

C. Program Utama (Silakan ditulis ulang dalam Bahasa C++)

Kamus

L: List

Algoritma

```
createList (L)
printList (L)           {}
insertAscending (L, 25)
printList (L)           {25}
insertAscending (L, 10)
printList (L)           {10 25}
insertAscending (L, 32)
printList (L)           {10 25 32}
insertAscending (L, 3)
printList (L)           {3 10 25 32}
deleteElm (L, 32)
printList (L)           {3 10 25}
deleteElm (L, 3)
printList (L)           {10 25}
deleteElm (L, 10)
printList (L)           {25}
deleteElm (L, 25)
printList (L)           {}
```

JURNAL PRAKTIKUM MODUL 4
MK STRUKTUR DATA 2019/2020-2
Double Linked-List dengan pointer First dan Last

D. TUGAS TERBIMBING

Buat implementasi procedure berikut ini:

Procedure insertFirst (**input/ouput** L : List, **input** P : address)

{ *IS. Terdefinisi pointer P berisi alamat elmList, dan sebuah list L (L tidak kosong).*
 FS. elmList yang ditunjuk oleh P ditambahkan ke dalam list sebagai elemen terakhir. }

Procedure insertAscending (**input/ouput** L : List, **input** dataBaru : infotype)

{ *IS. Terdefinisi sebuah data, dan sebuah list L (L mungkin kosong).*
 FS. dataBaru ditambahkan ke dalam list dengan aturan: data di dalam list harus selalu
 terurut secara menaik (ascending).
 Note: Gunakan procedure insertFirst, insertLast, dan insertAfter yang sudah dibuat
 sebelumnya. }

Procedure deleteLast (**input/ouput** L : List, **output** P : address)

{ *IS. Terdefinisi sebuah list L (L tidak kosong dan berisi lebih dari satu elemen).*
 FS. P berisi alamat elmList yang terakhir, elmList yang ditunjuk oleh P dihapus dari list
 }

Procedure deleteElm (**input/ouput** L : List, **input** dataHapus : infotype)

{ *IS. Terdefinisi sebuah list L (L mungkin kosong).*
 FS. Elemen dengan info = dataHapus, dihapus dari list.
 Note: Gunakan procedure deleteFirst, deleteLast, dan deleteAfter yang sudah dibuat
 sebelumnya. }

JURNAL PRAKTIKUM MODUL 4
MK STRUKTUR DATA 2019/2020-2
Double Linked-List dengan pointer First dan Last

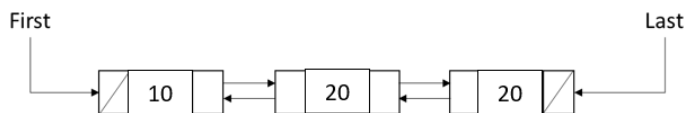
E. TUGAS MANDIRI (50 Menit)

1. Buat implementasi function berikut ini:

Function findElemen (L: List, dataDicari: infotype) → boolean

{ Mencari sebuah elemen dalam list L. Kembalikan "true" jika ditemukan dan "false" jika tidak ditemukan.

Ilustrasi:



Elemen 20 ditemukan di List.

}

Function frequencyofElm (L: List, dataDicari: infotype) → integer

{ IS. List mungkin kosong. Jika list tidak kosong, maka data dalam list sudah terurut.

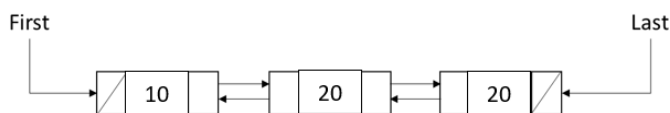
FS. Mengembalikan jumlah kemunculan suatu elemen yang ada pada list. Jika list kosong atau elemen tidak ditemukan di List, maka kembalikan 0.

Note: Gunakan function findElemen yang sudah dibuat sebelumnya.

Ilustrasi:



Elemen 20 ditemukan di list (true).



Elemen 20 muncul 2 kali pada List.

}

2. Buat program utama seperti berikut:

Kamus

Algoritma

insertAscending (L, 25)

printList (L) {25}

insertAscending (L, 10)

printList (L) {10 25}

insertAscending (L, 10)

JURNAL PRAKTIKUM MODUL 4
MK STRUKTUR DATA 2019/2020-2
Double Linked-List dengan pointer First dan Last

printList (L)	{10 10 25}
insertAscending (L, 25)	
printList (L)	{10 10 25 25}
insertAscending (L, 25)	
printList (L)	{10 10 25 25 25}
findElemen(L, 10)	{true}
frequencyofElm(L, 10)	{2}
frequencyofElm(L, 25)	{3}
frequencyofElm(L, 2)	{0}