PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

PERTEMUAN KE 11 JOBSHEET 9 Linked List

Dosen Pengajar : Triana Fatmawati, S.T., M.T.

Muhammad Afiq Firdaus

2341760189 / 21

SIB1E

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS

JURUSAN TEKNOLOGI INFORMASI

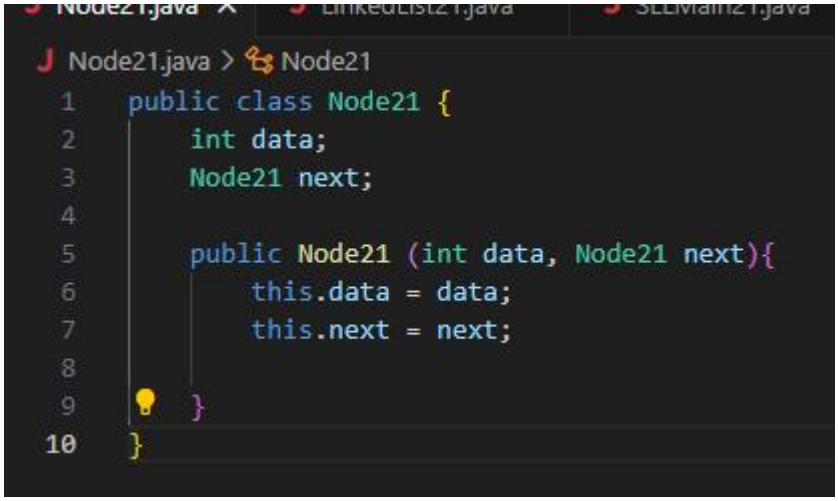
POLITEKNIK NEGERI MALANG

2024

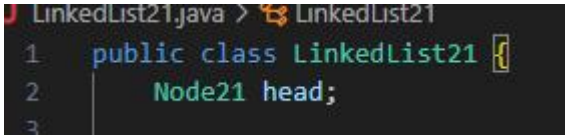
Praktikum 1 Pembuatan Linked List

Tambahkan class-class berikut:
a. Node.java
b. LinkedList.java
c. SLLMain.java

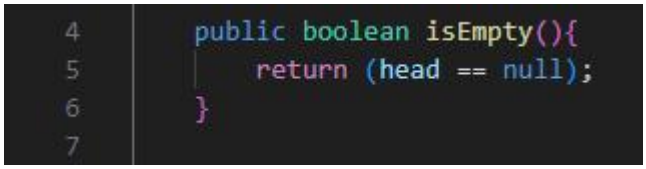
Deklarasikan class **Node** yang memiliki atribut data untuk menyimpan elemen dan atribut next bertipe Node untuk menyimpan node berikutnya. Tambahkan constructor berparameter untuk mempermudah inisialisasi



Deklarasikan class **LinkedList** yang memiliki atribut head. Atribut head menyimpan node pertama pada linked list



Sebagai langkah berikutnya, akan diimplementasikan method-method yang terdapat pada class LinkedList. Tambahkan method isEmpty()



Implementasi method print() untuk mencetak dengan menggunakan proses traverse.

```
8   public void print(){
9       if(!isEmpty()){
10          System.out.println(x:"Isi linked list");
11          Node21 currentNode = head;
12
13          while(currentNode != null){
14              System.out.println(currentNode.data + "\t");
15              currentNode = currentNode.next;
16          }
17          System.out.println(x:"");
18      }else{
19          System.out.println(x:"Linked list kosong");
20      }
21  }
```

Implementasikan method addFirst() untuk menambahkan node baru di awal linked list

```
23      public void addFirst(int input){
24          Node21 newNode = new Node21(input, next:null);
25
26          if (isEmpty()){
27              head = newNode;
28          }else{
29              newNode.next = head;
30              head = newNode;
31          }
32      }
```

Implementasikan method addLast() untuk menambahkan node baru di akhir linked list

```
34      public void addLast(int input){
35          Node21 newNode = new Node21(input, next:null);
36
37          if(isEmpty()){
38              head = newNode;
39          }else{
40              Node21 currentNode = head;
41
42              while (currentNode.next != null){
43                  currentNode = currentNode.next;
44              }
45              currentNode.next = newNode;
46          }
47      }
```

Implementasikan method insertAfter() menambahkan node baru pada posisi setelah node yang berisi data tertentu (key)

```
49      public void inserAfter(int key, int input){
50          Node21 newNode = new Node21(input, next:null);
51
52          if(!isEmpty()){
53              Node21 currentNode = head;
54
55              do{
56                  if(currentNode.data == key){
57                      newNode.next = currentNode.next;
58                      currentNode.next = newNode;
59                      break;
60                  }
61                  currentNode = currentNode.next;
62              }while (currentNode != null);
63          }else{
64              System.out.println(x:"Linked list kosong");
65          }
66      }
```

Pada class **SLLMain**, buatlah fungsi main, kemudian buat object myLinkedList bertipe LinkedList. Lakukan penambahan beberapa data. Untuk melihat efeknya terhadap object myLinkedList, panggil method print()

SLLMain21.java

```java
public class SLLMain21{
    Run | Debug
    public static void main(String[] args) {
        LinkedList21 myLinkedList = new LinkedList21();
        myLinkedList.print();
        myLinkedList.addFirst(input:800);
        myLinkedList.print();
        myLinkedList.addFirst(input:700);
        myLinkedList.print();
        myLinkedList.addLast(input:500);
        myLinkedList.print();
        myLinkedList.inserAfter(key:700, input:300);
        myLinkedList.print();
    }
}
```

Verifikasi Hasil Percobaan

```
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 11>  & 'C:
gram Files\Java\jdk-21\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\HP
Data\Roaming\Code\User\workspaceStorage\da4cf9b82fa8f8a73247e3dffb085b1c\redhat.java\jdt_ws\Perte
 11_fa28d1b\bin' 'SLLMain21'
Linked list kosong
Isi linked list
800

Isi linked list
700
800

Isi linked list
700
800
500

Isi linked list
700
300
800
500

PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 11>
```

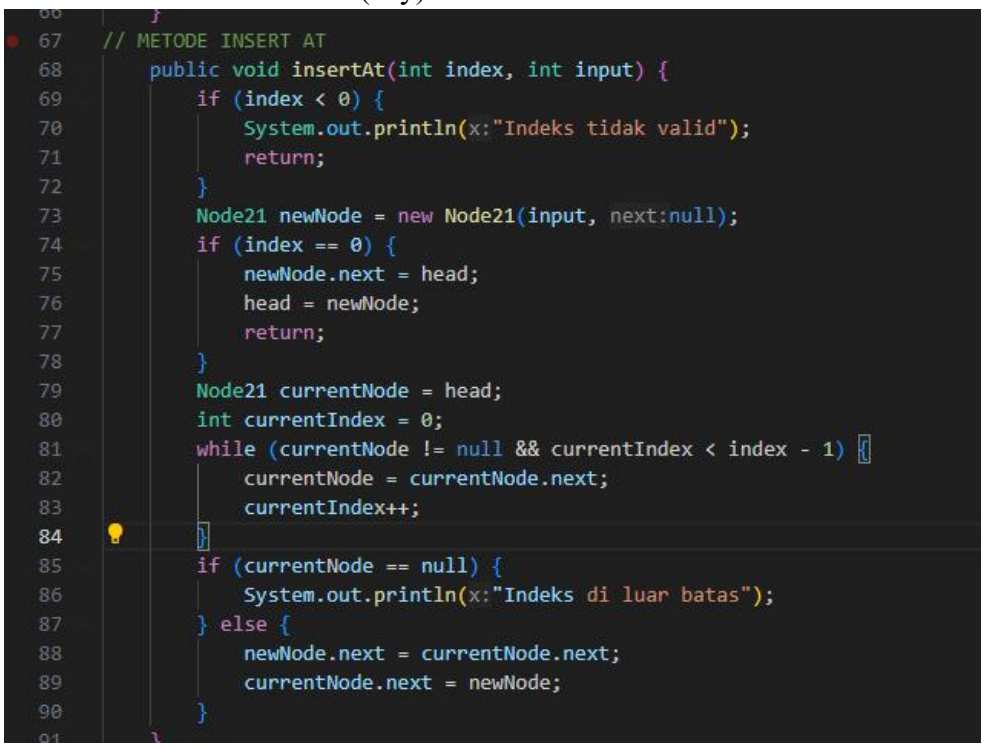
Pertanyaan

1. Mengapa class LinkedList tidak memerlukan method isFull() seperti halnya Stack dan Queue?
2. Mengapa class LinkedList hanya memiliki atribut head yang menyimpan informasi node pertama? Bagaimana informasi node kedua dan lainnya diakses?
3. Pada langkah, jelaskan kegunaan kode berikut

```java
if (currentNode.data == key) {
    newNode.next = currentNode.next;
    currentNode.next = newNode;
    break;
}
```

4. Implementasikan method insertAt(int index, int key) dari tugas mata kuliah ASD (Teori)

Jawaban

1. karena struktur data LinkedList pada dasarnya tidak memiliki batasan ukuran yang tetap seperti Stack atau Queue yang diimplementasikan dengan array
2. struktur dasar dari LinkedList didesain untuk bekerja secara dinamis dan fleksibel dengan memori yang dialokasikan untuk setiap elemen (node) secara individu.
3. untuk menyisipkan sebuah node baru ke dalam LinkedList setelah node yang memiliki nilai data tertentu (key).
4. 

```
67  // METODE INSERT AT
68      public void insertAt(int index, int input) {
69          if (index < 0) {
70              System.out.println(x:"Indeks tidak valid");
71              return;
72          }
73          Node21 newNode = new Node21(input, next:null);
74          if (index == 0) {
75              newNode.next = head;
76              head = newNode;
77              return;
78          }
79          Node21 currentNode = head;
80          int currentIndex = 0;
81          while (currentNode != null && currentIndex < index - 1) {
82              currentNode = currentNode.next;
83              currentIndex++;
84          }
85          if (currentNode == null) {
86              System.out.println(x:"Indeks di luar batas");
87          } else {
88              newNode.next = currentNode.next;
89              currentNode.next = newNode;
90          }
```

Praktikum 2 Mengakses dan menghapus node pada Linked List

Tambahkan method getData() untuk mengembalikan nilai elemen di dalam node pada index tertentu

```
183     public int getData(int index) {
184         if (index < 0 || isEmpty()) {
185             System.out.println(x:"Indeks tidak valid atau Linked list kosong");
186             return -1; // Mengembalikan -1 untuk indeks tidak valid atau list kosong
187         }
188         Node21 currentNode = head;
189         int currentIndex = 0;
190         while (currentNode != null) {
191             if (currentIndex == index) {
192                 return currentNode.data;
193             }
194             currentNode = currentNode.next;
195             currentIndex++;
196         }
197         System.out.println(x:"Indeks di luar batas");
198         return -1; // Mengembalikan -1 jika indeks di luar batas
199     }
```

Tambahkan method indexOf() untuk mengetahu index dari node dengan elemen tertentu

```
201     public int indexOf(int key) {
202         Node21 currentNode = head;
203         int index = 0;
204         while (currentNode != null && currentNode.data != key) {
205             currentNode = currentNode.next;
206             index++;
207         }
208         if (currentNode == null) {
209             return -1;
210         } else {
211             return index;
212         }
213     }
```

Tambahkan method removeFirst() untuk menghapus node pertama pada linked list

```
215     public void removeFirst() {
216         if (!isEmpty()) {
217             head = head.next;
218         } else {
219             System.out.println(x:"Linked list kosong");
220         }
221     }
222
```

Tambahkan method removeLast() untuk menghapus node terakhir pada linked list

```
223 public void removeLast() {
224     if (isEmpty()) {
225         System.out.println(x:"Linked list kosong");
226     } else if (head.next == null) {
227         head = null;
228     } else {
229         Node21 currentNode = head;
230         while (currentNode.next != null) {
231             if (currentNode.next.next == null) {
232                 currentNode.next = null;
233                 break;
234             }
235             currentNode = currentNode.next;
236         }
237     }
238 }
```

Method remove() digunakan untuk mengapus node yang berisi elemen tertentu

```
240     public void remove(int key) {
241         if (isEmpty()) {
242             System.out.println(x:"Linked list kosong");
243         } else if (head.data == key) {
244             removeFirst();
245         } else {
246             Node21 currentNode = head;
247             while (currentNode.next != null) {
248                 if (currentNode.next.data == key) {
249                     currentNode.next = currentNode.next.next;
250                     break;
251                 }
252                 currentNode = currentNode.next;
253             }
254         }
255     }
```

Kemudian, coba lakukan pengaksesan dan penghapusan data di method main pada class SLLMain dengan menambahkan kode berikut

```
        // Menghapus elemen dan menampilkan isi linked list setelah setiap penghapusan
        myLinkedList.remove(key:300);
        myLinkedList.print();

        myLinkedList.removeFirst();
        myLinkedList.print();

        myLinkedList.removeLast();
        myLinkedList.print();
    }
}
```

Verifikasi Hasil Percobaan

```
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 11>  c:; cd 'c:\Muh
ktikum\Pertemuan 11'; & 'C:\Program Files\Java\jdk-21\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessag
rage\da4cf9b82fa8f8a73247e3dffb085b1c\redhat.java\jdt_ws\Pertemuan 11_fa28d1b\bin' 'SLLMain21'
Linked list kosong
Isi linked list: 800
Isi linked list: 700    800
Isi linked list: 700    800     500
Isi linked list: 700    300     800     500
Data pada index ke-1: 300
Isi linked list: 700    800     500
Isi linked list: 800    500
Isi linked list: 800
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 11>
```

Pertanyaan

1. Jelaskan maksud potongan kode di bawah pada method remove()

```
if (currentNode.next.data == key) {
    currentNode.next = currentNode.next.next;
    break;
}
```

2. Jelaskan maksud if-else block pada method indexOf() berikut

```
if (currentNode == null) {
    return -1;
} else {
    return index;
}
```

3. Error apa yang muncul jika argumen method getData() lebih besar dari jumlah node pada linked list? Modifikasi kode program untuk menghandle hal tersebut.
4. Apa fungsi keyword break pada method remove()? Bagaimana efeknya jika baris tersebut dihapus?

Jawaban

1. untuk menghapus node dengan nilai tertentu dari linked list
2. untuk mencari indeks dari node yang memiliki nilai tertentu (key) dalam linked list
3. maka akan terjadi **NullPointerException**. Hal ini terjadi karena currentNode akan menjadi null saat iterasi mencapai akhir linked list, dan kemudian mencoba mengakses currentNode.data, yang tidak ada (karena currentNode adalah null).
   Modifikasi

```
public int getData(int index) {
    if (index < 0 || isEmpty()) {
        System.out.println(x:"Indeks tidak valid atau Linked list kosong");
        return -1; // Mengembalikan -1 untuk indeks tidak valid atau list kosong
    }
    Node21 currentNode = head;
    int currentIndex = 0;
    while (currentNode != null) {
        if (currentIndex == index) {
            return currentNode.data;
        }
        currentNode = currentNode.next;
        currentIndex++;
    }
    System.out.println(x:"Indeks di luar batas");
    return -1; // Mengembalikan -1 jika indeks di luar batas
}
```

4. Keyword break pada metode remove(int key) memiliki fungsi penting untuk menghentikan eksekusi loop setelah node yang memiliki nilai key ditemukan dan dihapus dari linked list. Jika baris break dihapus, maka loop akan terus berjalan sampai mencapai akhir linked list, meskipun node dengan nilai key sudah ditemukan dan dihapus

Tugas

1. Implementasikan method-method berikut pada class LinkedList:
   a. insertBefore() untuk menambahkan node sebelum keyword yang diinginkan
   b. insertAt(int index, int key) untuk menambahkan node pada index tertentu
   c. removeAt(int index) untuk menghapus node pada index tertentu
2. Dalam suatu game scavenger hunt, terdapat beberapa point yang harus dilalui peserta untuk menemukan harta karun. Setiap point memiliki soal yang harus dijawab, kunci jawaban, dan pointer ke point selanjutnya. Buatlah implementasi game tersebut dengan linked list.

Jawaban

1.

```java
// TUGAS 1
public class SLLMain21 {
    Run | Debug
    public static void main(String[] args) {
        LinkedList21 myLinkedList = new LinkedList21();
        myLinkedList.print();

        myLinkedList.addFirst(input:800);
        myLinkedList.print();

        myLinkedList.addFirst(input:700);
        myLinkedList.print();

        myLinkedList.addLast(input:500);
        myLinkedList.print();

        myLinkedList.insertAfter(key:700, input:300);
        myLinkedList.print();

        // a. insertBefore() demonstration
        myLinkedList.insertBefore(key:800, input:600);
        myLinkedList.print();

        // b. insertAt() demonstration
        myLinkedList.insertAt(index:2, input:900);
        myLinkedList.print();

        // c. removeAt() demonstration
        myLinkedList.removeAt(index:2);
        myLinkedList.print();

        // Demonstrate getData()
        System.out.println("Data pada index ke-1: " + myLinkedList.getData(index:1));

        // Demonstrate indexOf()
        System.out.println("Data 300 berada pada index ke: " + myLinkedList.indexOf(key:300));

        // Demonstrate remove()
        myLinkedList.remove(key:300);
        myLinkedList.print();

        // Demonstrate removeFirst()
        myLinkedList.removeFirst();
        myLinkedList.print();

        // Demonstrate removeLast()
        myLinkedList.removeLast();
        myLinkedList.print();
    }
}
```

```
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 11>  c:; cd 'c:\Muh
ktikum\Pertemuan 11'; & 'C:\Program Files\Java\jdk-21\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessag
rage\da4cf9b82fa8f8a73247e3dffb085b1c\redhat.java\jdt_ws\Pertemuan 11_fa28d1b\bin' 'SLLMain21'
Linked list kosong
Isi linked list:
800
Isi linked list:
700     800
Isi linked list:
700     800     500
Isi linked list:
700     300     800     500
Isi linked list:
700     300     600     800     500
Isi linked list:
700     300     900     600     800     500
Isi linked list:
700     300     600     800     500
Data pada index ke-1: 300
Data 300 berada pada index ke: 1
Isi linked list:
700     600     800     500
Isi linked list:
600     800     500
Isi linked list:
600     800
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 11>
```

2. Berikut adalah hasil dari code yang telah dibuat

```
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 11>  & '
ionMessages' '-cp' 'C:\Users\HP\AppData\Roaming\Code\User\workspaceStorage\da4cf9b82fa8f8a73247
Apa ibukota Indonesia?
jakarta
Jawaban benar!
2 + 2 sama dengan berapa?
4
Jawaban benar!
Siapa presiden pertama Indonesia?
soekarno
Jawaban benar!
Apa warna langit di siang hari?
biru
Jawaban benar!
Selamat! Anda telah menemukan harta karun!
PS C:\Muhammad Afiq Firdaus\Semester 2\Algoritma dan Struktur Data\Praktikum\Pertemuan 11>
```