JOBSHEET IX

LINKED LIST

**NAMA : AHMAD HAYYIN B (01)**

**NIM : 2341760199**

1. Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Membuat struktur data linked list
2. Membuat linked list pada program
3. Membedakan permasalahan apa yang dapat diselesaikan menggunakan linked list

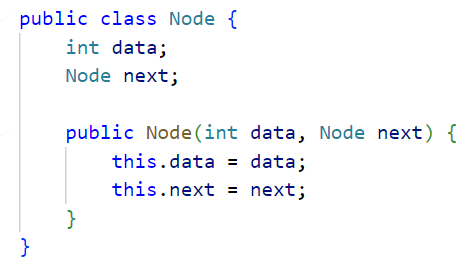
2. Praktikum

2.1 Pembuatan Linked List

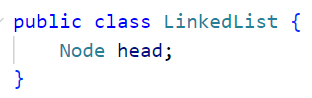
**Waktu percobaan: 50 menit**

Didalam praktikum ini, akan dilakukan implementasi pembuatan linked list menggunakan array dan penambahan node ke dalam linked list

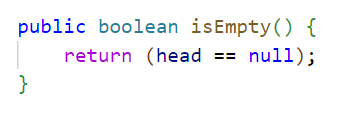
1. Buat folder baru Praktikum09
2. Tambahkan class-class berikut:
   1. Node.java
   2. LinkedList.java
   3. SLLMain.java
3. Deklarasikan class Node yang memiliki atribut data untuk menyimpan elemen dan atribut next bertipe Node untuk menyimpan node berikutnya. Tambahkan constructor berparameter untuk mempermudah inisialisasi



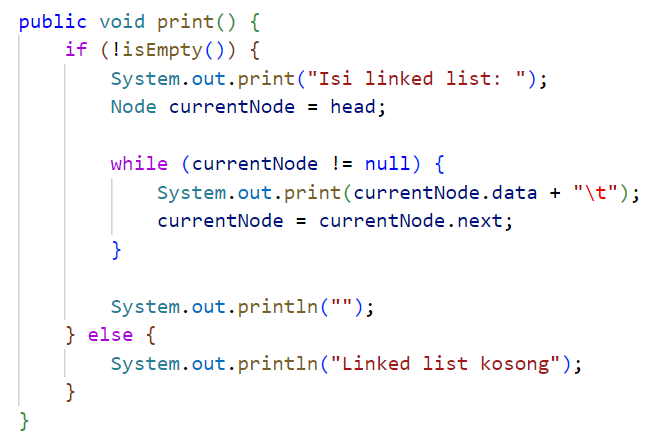
1. Deklarasikan class LinkedList yang memiliki atribut head. Atribut head menyimpan node pertama pada linked list



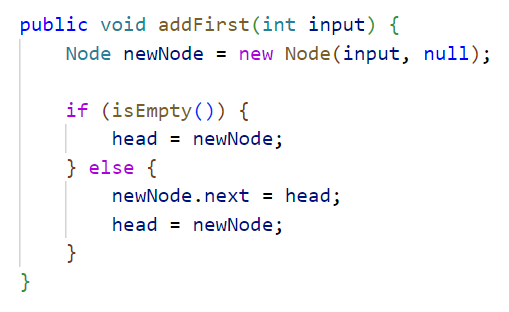
1. Sebagai langkah berikutnya, akan diimplementasikan method-method yang terdapat pada class LinkedList.
2. Tambahkan method **isEmpty()**



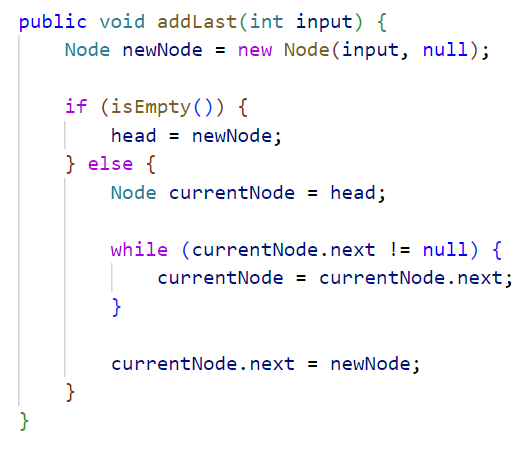
1. Implementasi method print() untuk mencetak dengan menggunakan proses traverse.



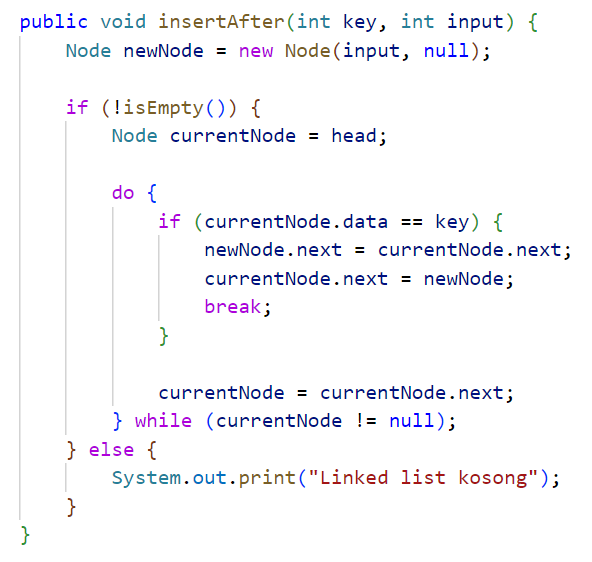
1. Implementasikan method **addFirst()** untuk menambahkan node baru di awal linked list



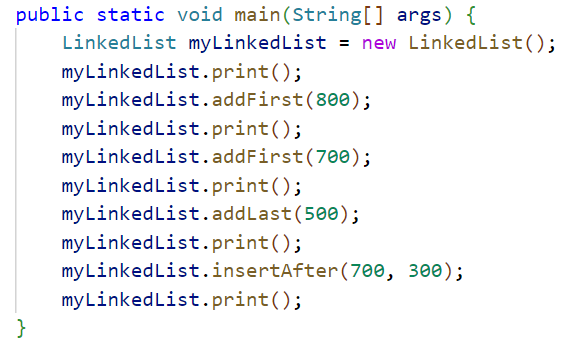
1. Implementasikan method **addLast()** untuk menambahkan node baru di akhir linked list



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

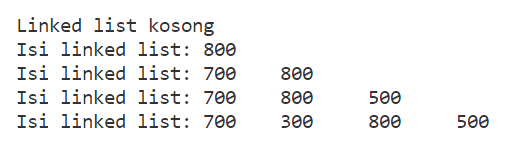


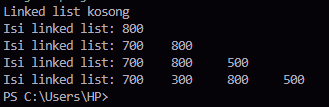
1. Pada class SLLMain, buatlah fungsi **main**, kemudian buat object myLinkedList bertipe LinkedList. Lakukan penambahan beberapa data. Untuk melihat efeknya terhadap object myLinkedList, panggil method print()



1. Verifikasi Hasil Percobaan

Cocokkan hasil run program Anda dengan output berikut ini.





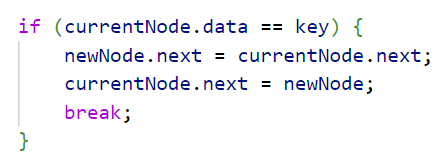
1. Pertanyaan
2. Mengapa class LinkedList tidak memerlukan method isFull() seperti halnya Stack dan Queue?

**LinkedList does not require the isFull() method because linked lists do not have capacity limitations like stacks and queues which use arrays for storage.**

1. Mengapa class LinkedList hanya memiliki atribut head yang menyimpan informasi node pertama? Bagaimana informasi node kedua dan lainnya diakses?

**The LinkedList class only has a head attribute that displays the information of the first node because a linked list is a data structure consisting of nodes that are always connected through pointers and references. We need to follow the next pointer from each node to get information about the second and subsequent nodes.**

1. Pada langkah, jelaskan kegunaan kode berikut



**This code is used to insert a new node (newNode) after a node that has the same data value as the key**

**Here are the main points:**

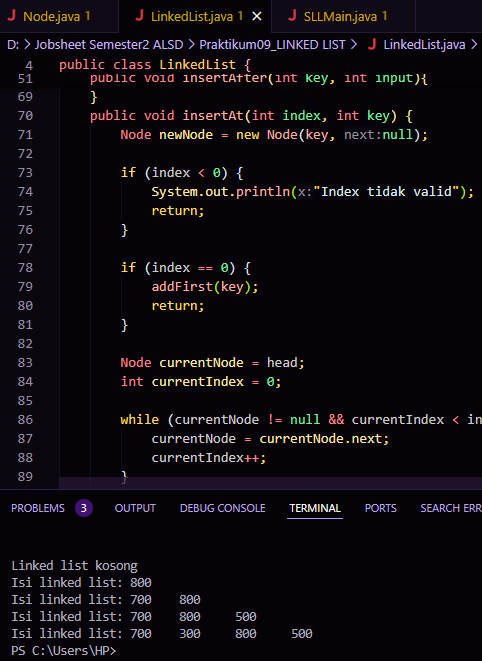
**• Determine whether the data from the current Node is the same as the key**

**• If so, newNode is connected to the node after currentNode by setting newNode.next to currentNode.next.**

**• Next, currentNode.next is changed to newNode, causing newNode to appear between currentNode and the next node.**

**• break; is used to exit the do-while loop after the insert operation is successfully completed.**

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



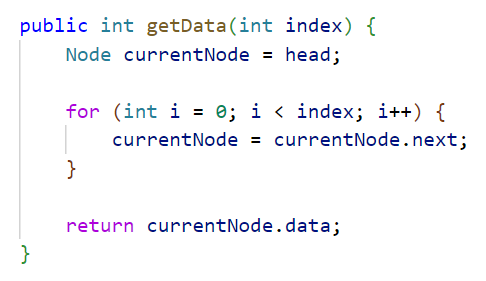
2.2 Mengakses dan menghapus node pada Linked List

**Waktu percobaan: 50 menit**

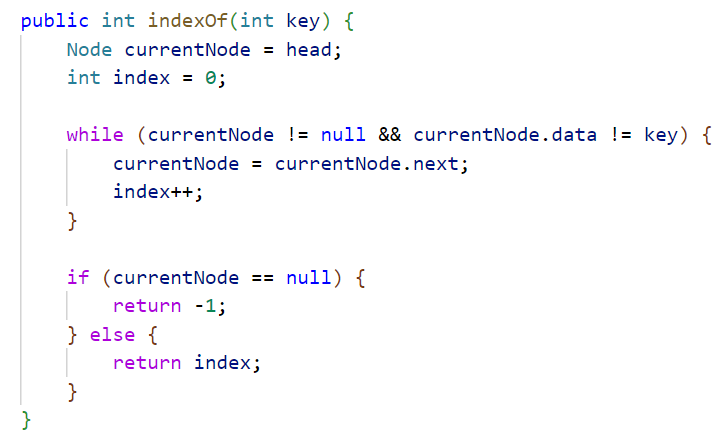
Didalam praktikum ini, kita akan mengimplementasikan method untuk melakukan pengaksesan dan penghapusan data pada linked list

2.2.1 Langkah-langkah Percobaan

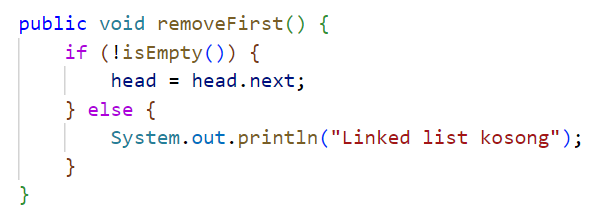
1. Tambahkan method getData() untuk mengembalikan nilai elemen di dalam node pada index tertentu



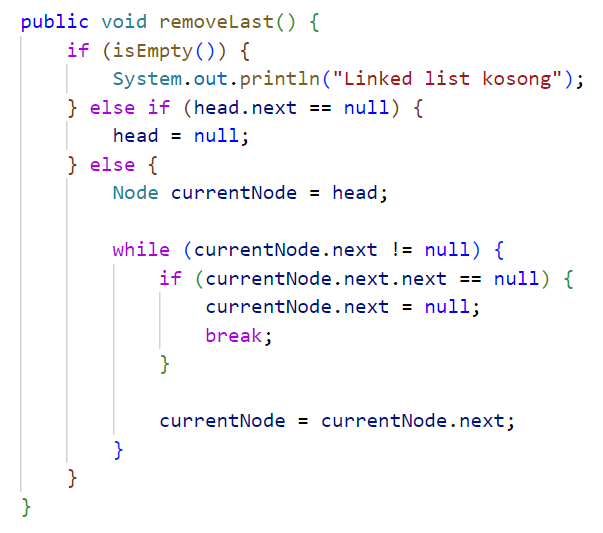
1. Tambahkan method indexOf() untuk mengetahu index dari node dengan elemen tertentu



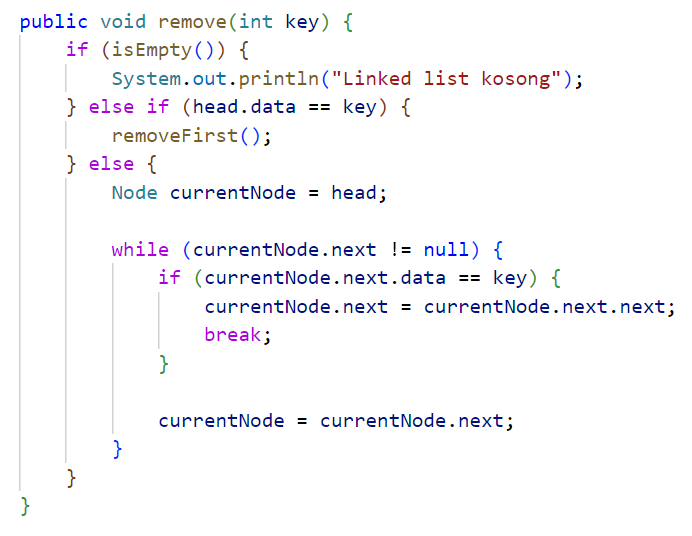
1. Tambahkan method removeFirst() untuk menghapus node pertama pada linked list



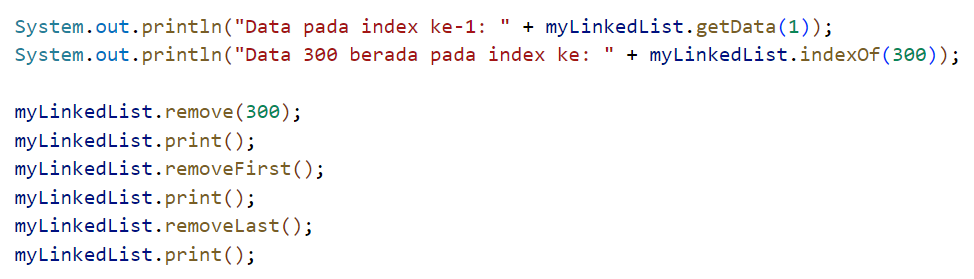
1. Tambahkan method removeLast() untuk menghapus node terakhir pada linked list



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

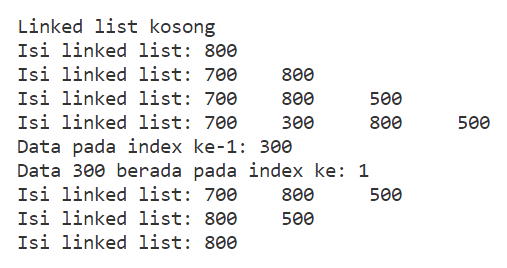


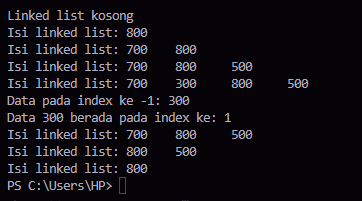
1. Kemudian, coba lakukan pengaksesan dan penghapusan data di method main pada class SLLMain dengan menambahkan kode berikut



1. Compile dan run program kemudian amati hasilnya
   * 1. Verifikasi Hasil Percobaan

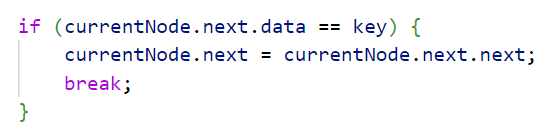
Cocokkan hasil run program dengan output berikut ini.





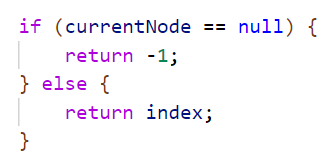
* + 1. Pertanyaan

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



**serves to delete nodes with the same data value as the given key value. If the data value of the next node (currentNode.next.data) is the same as the key, then the node is deleted by changing the next reference of currentNode to point directly to the node after it (currentNode.next.next). After that, the deletion process is stopped with break; so as not to continue iteration on the next nodes.**

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



**If-else block pada method `indexOf()` mengembalikan `-1` jika nilai `currentNode` menjadi `null`, menandakan bahwa `key` tidak ditemukan dalam linked list. Jika nilai `currentNode` tidak `null`, method mengembalikan nilai `index`, menunjukkan indeks dari node dengan nilai data yang sama dengan `key`.**

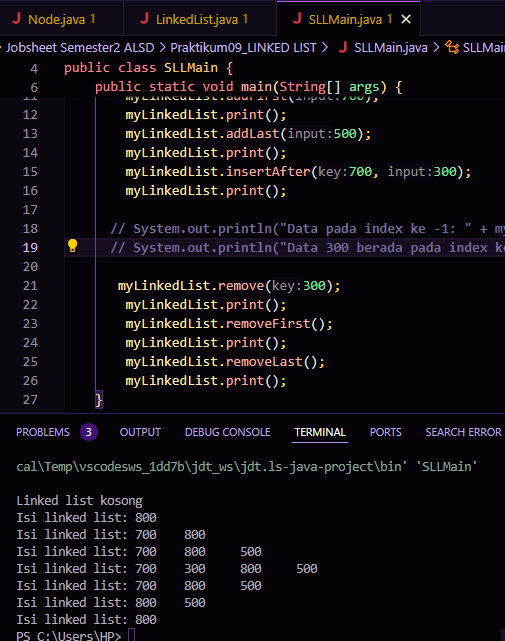
1. Error apa yang muncul jika argumen method getData() lebih besar dari jumlah node pada linked list? Modifikasi kode program untuk menghandle hal tersebut.
2. Apa fungsi keyword break pada method remove()? Bagaimana efeknya jika baris tersebut dihapus?

**The `break` keyword in the `remove()` method is used to stop the iteration after a node with the corresponding data value is found and removed. If the row is removed, the iteration will continue even though the node with the corresponding data value has been found and removed, which can cause an error.**

3. Tugas

**Waktu pengerjaan: 50 menit**

1. Implementasikan method-method berikut pada class LinkedList:
2. insertBefore() untuk menambahkan node sebelum keyword yang diinginkan
3. insertAt(int index, int key) untuk menambahkan node pada index tertentu
4. removeAt(int index) untuk menghapus node pada index tertentu



1. 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.

