  
Name : Azaria Cindy Sahasika

Number Id : 2341760169 / 06

Class : 1G – Business Information System

Lesson : Algorithm and Data Structure

Material : Material 9 – Linked List

Github Link : <https://github.com/azariacindy/algorithm-ds>

JOBSHEET IX

LINKED LIST

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

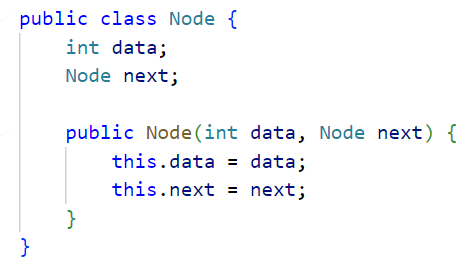
2. Praktikum

2.1 Pembuatan Linked List

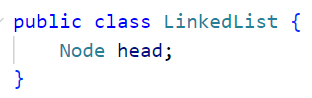
**Waktu percobaan: 50 menit**

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

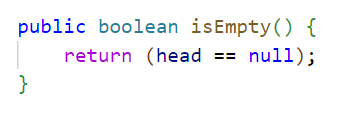
1. Buat folder baru Praktikum09
2. Tambahkan class-class berikut:
   1. Node.java
   2. LinkedList.java
   3. SLLMain.java
3. Deklarasikan class Node yang memiliki atribut data untuk menyimpan elemen dan atribut next bertipe Node untuk menyimpan node berikutnya. Tambahkan constructor berparameter untuk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



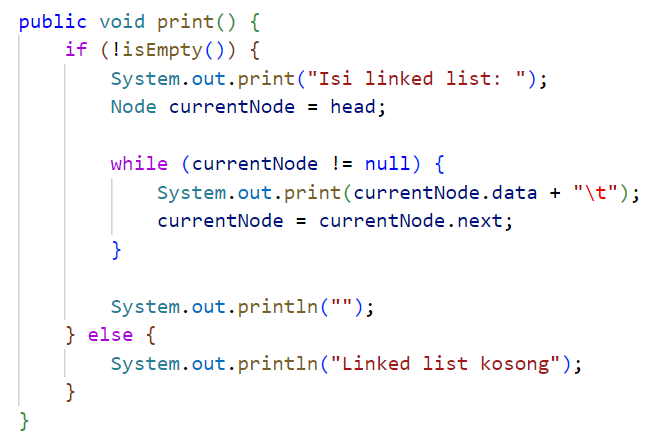
1. Deklarasikan class LinkedList yang memiliki atribut head. Atribut head menyimpan node pertama pada linked list



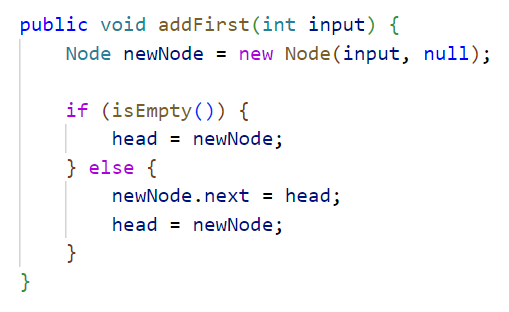
1. Sebagai langkah berikutnya, akan diimplementasikan method-method yang terdapat pada class LinkedList.
2. Tambahkan method **isEmpty()**



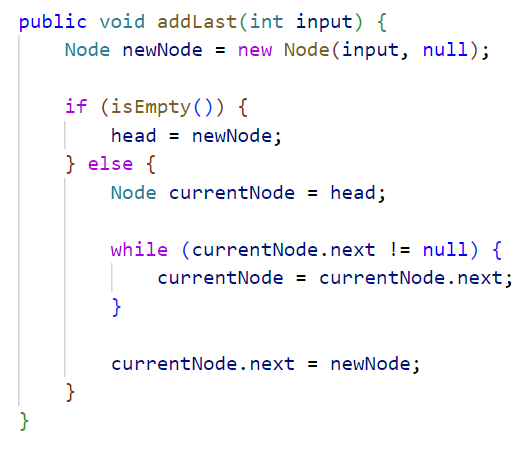
1. Implementasi method print() untuk mencetak dengan menggunakan proses traverse.



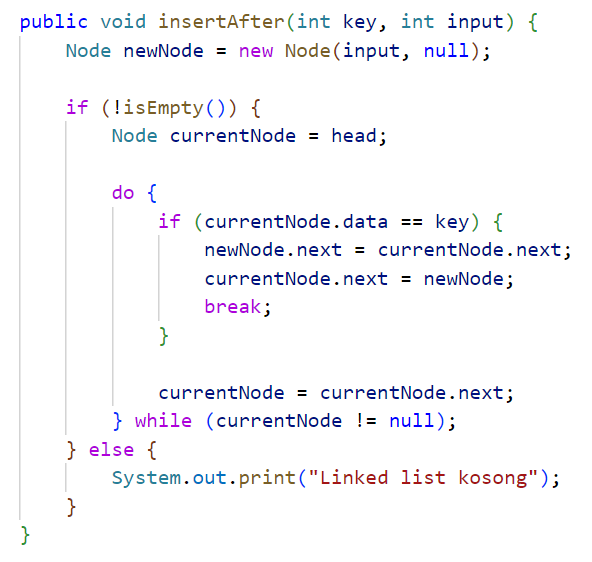
1. Implementasikan method **addFirst()** untuk menambahkan node baru di awal linked list



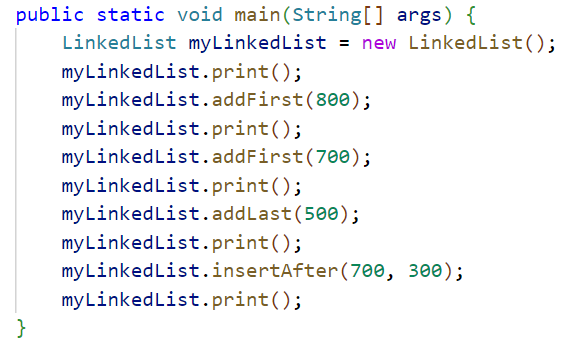
1. Implementasikan method **addLast()** untuk menambahkan node baru di akhir linked list



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

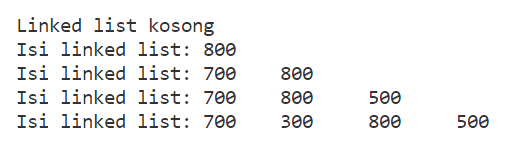


1. Pada class SLLMain, buatlah fungsi **main**, kemudian buat object myLinkedList bertipe LinkedList. Lakukan penambahan beberapa data. Untuk melihat efeknya terhadap object myLinkedList, panggil method print()



1. Verifikasi Hasil Percobaan

Cocokkan hasil run program Anda dengan output berikut ini.





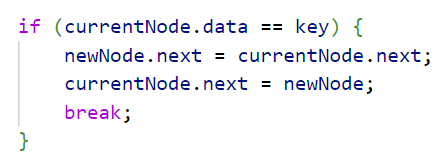
1. Pertanyaan
2. Mengapa class LinkedList tidak memerlukan method isFull() seperti halnya Stack dan Queue?

* Karena struktur data tersebut tidak memiliki batasan max dalam jumlah jumlah elemen yang dapat disimpan, seperti pada stack dan queue yang menggunakan array sebagai penyimpanan. LinkedList ini menggunakan node yang dihubungkan satu sama lain, sehingga dapat menambahkan atau menghapus elemen secara dinamis tanpa batasan max.

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

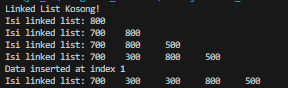
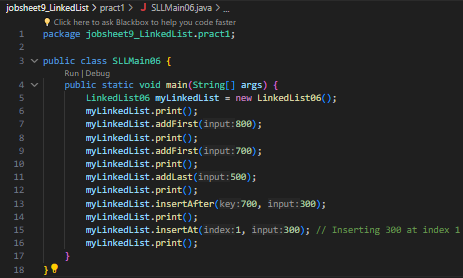
* Linked list hanya memiliki atribut ‘head’ yang dapat menyimpan informasi node pertama karena setiap node dalam LinkedList saling terhubung melalui referensi ‘next’. Untuk mengakses informasi node kedua dan seterusnya, jadi perlu mengikuti referensi ‘next’ dari node pertama(head) hingga ke node terakhir.

1. Pada langkah, jelaskan kegunaan kode berikut



* Kode tersebut sedang mencari node dengan nilai data tertentu ‘key’. Jika nilai data pada node saat ini ‘currentNode.data’ sama dengan nilai ‘key’, maka sisipkan kode baru ‘newNode’ setelah node saat ini. Kode ‘newNode.next = currentNode.next’ mengatur referensi next dari node baru (newNode) untuk menunjuk ke node berikutnya setelah currentNode. Kemudian, ‘currentNode.next = newNode’ mengatur referensi next dari currentNode untuk menunjuk ke node baru (newNode). Maka, node baru berhasil disisipkan di antara currentNode dan node berikutnya.

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





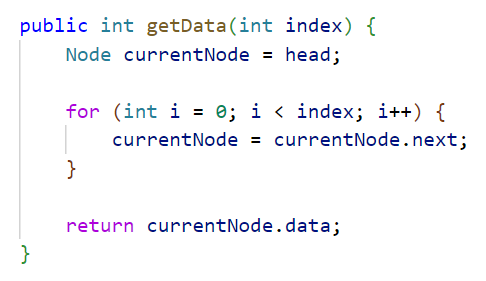
2.2 Mengakses dan menghapus node pada Linked List

**Waktu percobaan: 50 menit**

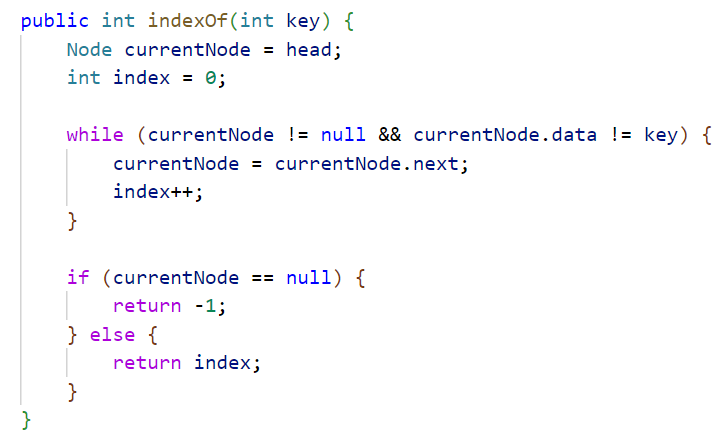
Didalam praktikum ini, kita akan mengimplementasikan method untuk melakukan pengaksesan dan penghapusan data pada linked list

2.2.1 Langkah-langkah Percobaan

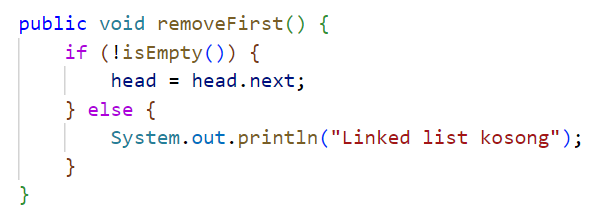
1. Tambahkan method getData() untuk mengembalikan nilai elemen di dalam node pada index tertentu



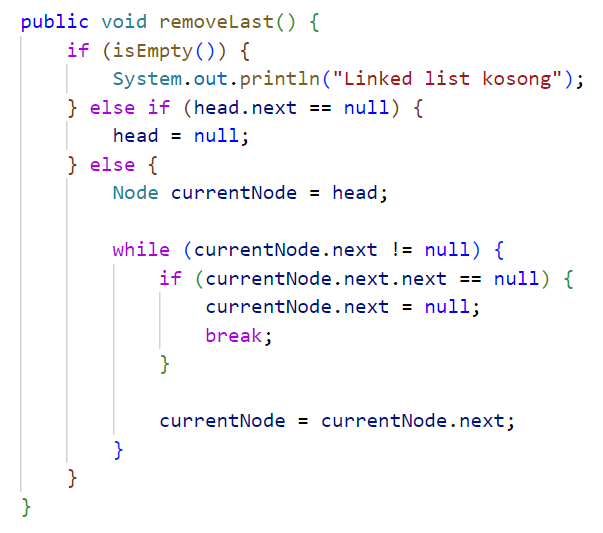
1. Tambahkan method indexOf() untuk mengetahu index dari node dengan elemen tertentu



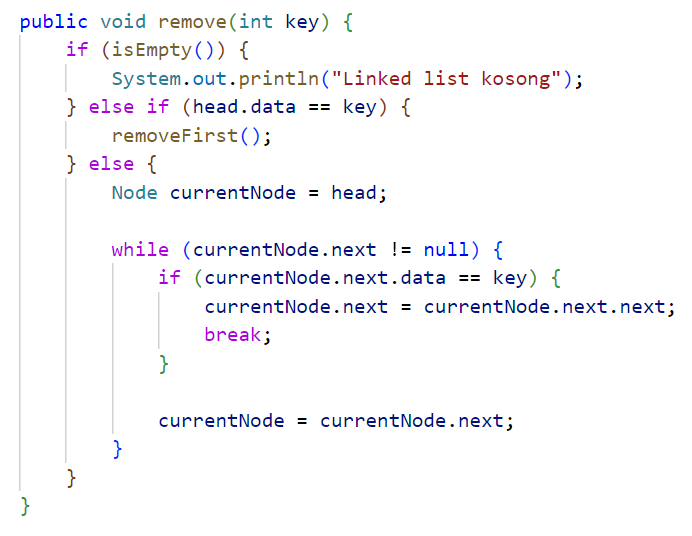
1. Tambahkan method removeFirst() untuk menghapus node pertama pada linked list



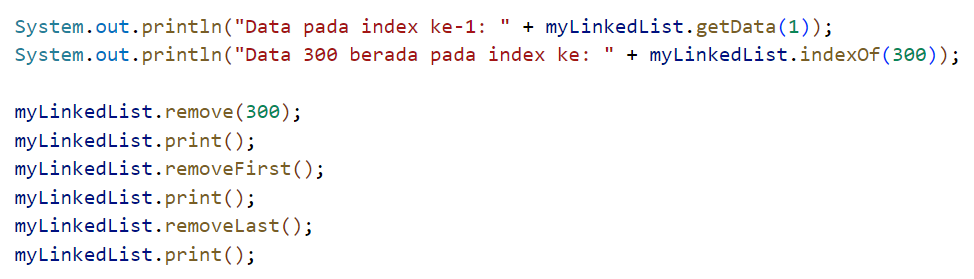
1. Tambahkan method removeLast() untuk menghapus node terakhir pada linked list



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

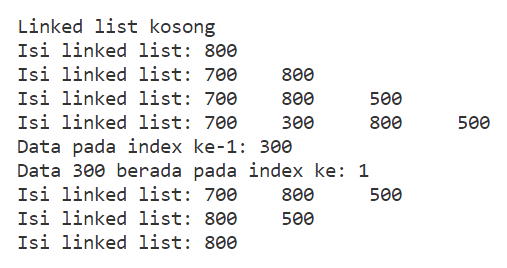


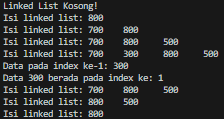
1. Kemudian, coba lakukan pengaksesan dan penghapusan data di method main pada class SLLMain dengan menambahkan kode berikut



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

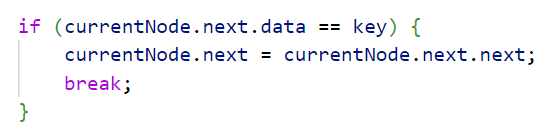
Cocokkan hasil run program dengan output berikut ini.





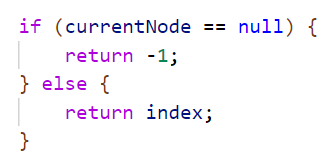
* + 1. Pertanyaan

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



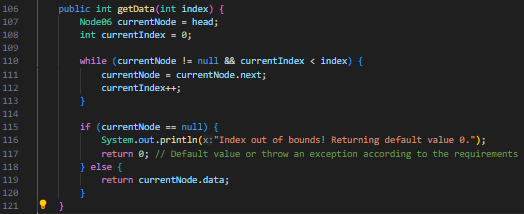
* Kode diatas digunakan untuk mengecek apakah nilai pada node selanjutnya sama dengan nilai key yang ingin dihapus. Jika iya, maka node tersebut akan dilewati dengan mengatur referensi ‘next’ dari ‘currentNode’ untuk menunjukkan ke node setelahnya sehingga node yang mengandung nilai kunci akan dihapus dari linked list.

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



* Kode diatas digunakan untuk menentukan apakah pencarian nilai key berhasilatau tidak. Jika ‘currentNode’ berakhir pada ‘null’, artinya nilai key tidak ditemukan dalam linked list, sehingga method mengembalikannilai ‘-1’. Jika tidak, maka nilai key ditemukan dan method mengembalikan indeks node tempat nilai key ditemukan.

1. Error apa yang muncul jika argumen method getData() lebih besar dari jumlah node pada linked list? Modifikasi kode program untuk menghandle hal tersebut.



1. Apa fungsi keyword break pada method remove()? Bagaimana efeknya jika baris tersebut dihapus?

* Digunakan untuk menghentikan iteraso atau looping setelah node dengan nilai key ditemukan dan dihapus. Jika baris tersebut dihapus, looping akan terus berlanjut sampai akhir linked list, meskipun node yang ingin dihapus sudah ditemukan dan dihapus. Ini akan menyebabkan waktu komputasi yang tidak efisien, karena tidak perlu mencari lebih lanjut setelah node ditemukan dan dihapus.

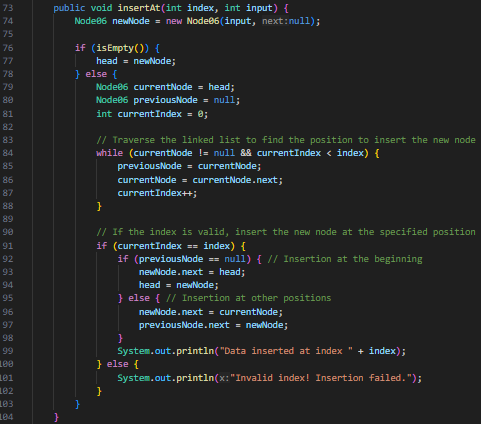
3. Tugas

**Waktu pengerjaan: 50 menit**

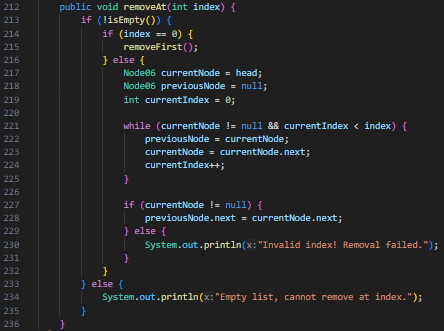
1. Implementasikan method-method berikut pada class LinkedList:
2. insertBefore() untuk menambahkan node sebelum keyword yang diinginkan



1. insertAt(int index, int key) untuk menambahkan node pada index tertentu



1. 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.