**LAPORAN PRAKTIKUM**

**MATA KULIAH ALGORITMA DAN STRUKTUR DATA**

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

**PERTEMUAN - 11 - Linked List**

****

**Nama : M. Zidna Billah Faza**

**NIM : 2341760030**

**Prodi : D-IV Sistem Informasi Bisnis**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**2024**

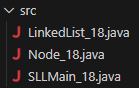
**Percobaan 1**

1. Buat folder baru Jobsheet 11

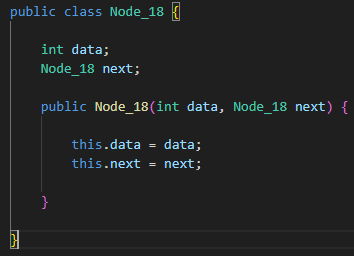


1. Tambahkan class-class berikut:

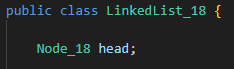
* Node.java
* LinkedList.java
* SLLMain.java



1. 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



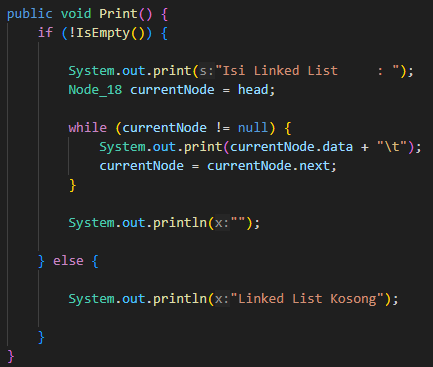
1. Deklarasikan class LinkedList yang memiliki atribut head. Atribut head menyimpan node pertama pada linked list



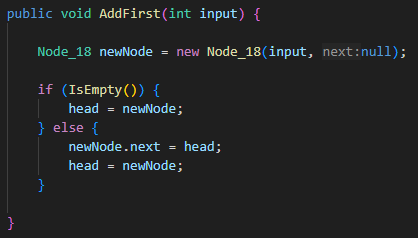
1. Sebagai langkah berikutnya, akan diimplementasikan method-method yang terdapat pada class LinkedList.
2. Tambahkan method isEmpty()



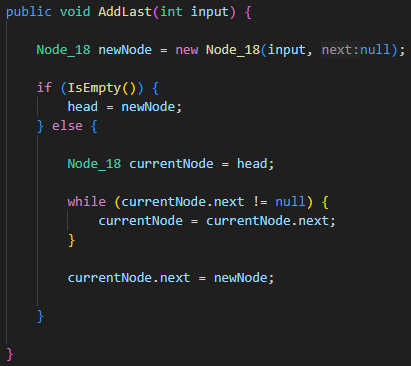
1. Implementasi method print() untuk mencetak dengan menggunakan proses traverse.



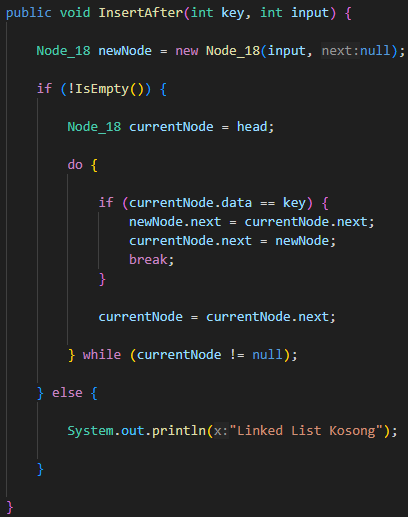
1. Implementasikan method addFirst() untuk menambahkan node baru di awal linked list



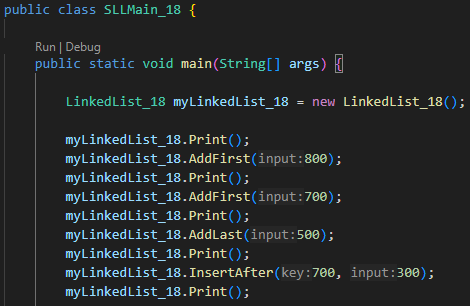
1. Implementasikan method addLast() untuk menambahkan node baru di akhir linked list



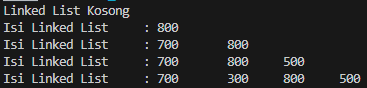
1. Implementasikan method insertAfter() menambahkan node baru pada posisi setelah node yang berisi data tertentu (key)



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



Pertanyaan Percobaan 1

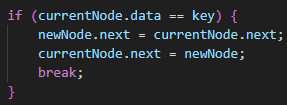
1. Mengapa class LinkedList tidak memerlukan method isFull() seperti halnya Stack dan Queue?

Karena pada class LinkedList tidak memiliki batasan ukuran tertentu karena bersifat dinamis sehingga node-node baru daoat ditambahkan selama memori masih ada

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

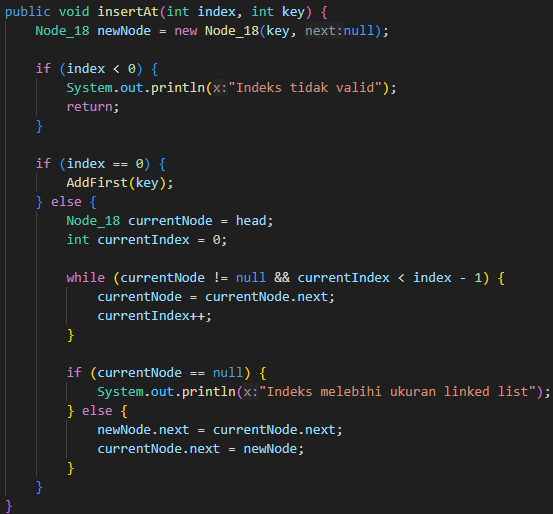
Karena head adalah titik awal dari LinkedList. Dari head, kita dapat mengakses seluruh LinkedList karena setiap node memiliki referensi ke node berikutnya (disebut 'next').

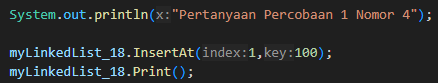
1. Pada langkah, jelaskan kegunaan kode berikut



Kode ini digunakan untuk menyisipkan sebuah node baru setelah node yang memiliki nilai data yang sama dengan kunci (key). Jadi, jika dalam LinkedList terdapat node dengan nilai data yang sama dengan kunci (key), maka kode ini akan menyisipkan node baru di antara node tersebut dan node berikutnya.

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

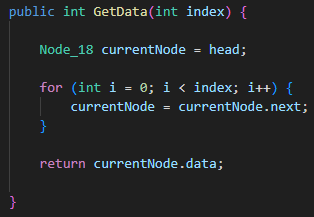




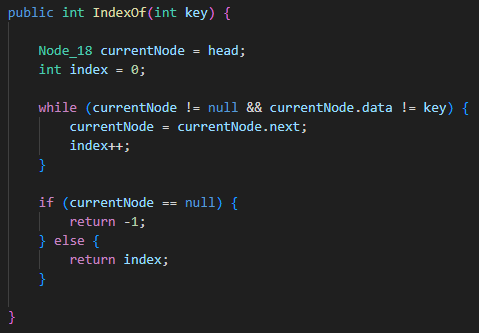


**Percobaan 2**

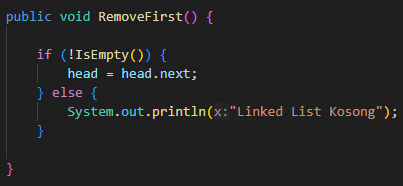
1. Tambahkan method getData() untuk mengembalikan nilai elemen di dalam node pada index tertentu



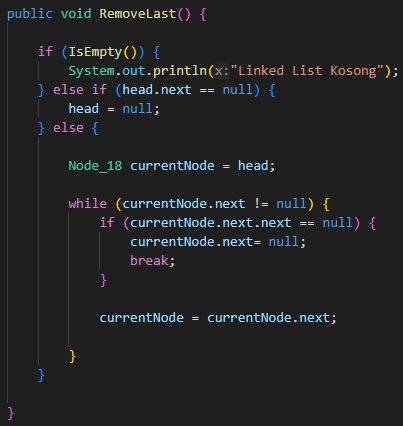
1. Tambahkan method indexOf() untuk mengetahu index dari node dengan elemen tertentu



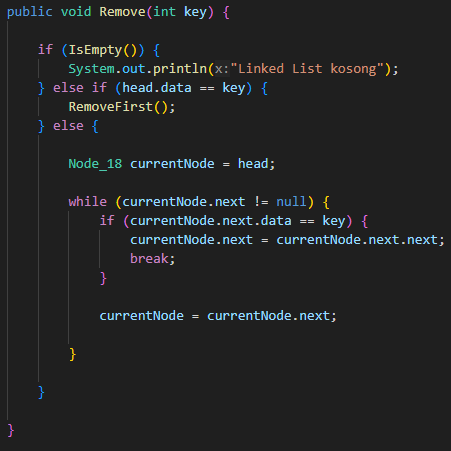
1. Tambahkan method removeFirst() untuk menghapus node pertama pada linked list



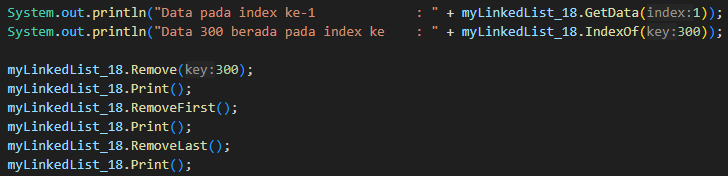
1. Tambahkan method removeLast() untuk menghapus node terakhir pada linked list



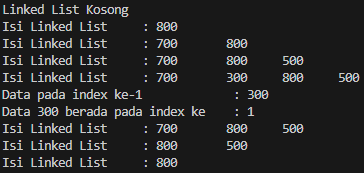
1. Method remove() digunakan untuk mengapus node yang berisi elemen tertentu



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

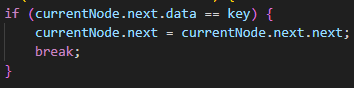


1. Compile dan run program kemudian amati hasilnya



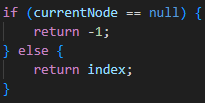
**Pertanyaan Percobaan 2**

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



kode tersebut berfungsi untuk menghapus node dari LinkedList yang memiliki nilai data yang sama dengan kunci (key).

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

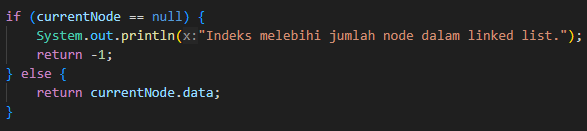


Pada method indexOf(), blok if-else tersebut bertujuan untuk menentukan hasil pencarian indeks (index) dari node yang memiliki nilai data sesuai dengan kunci (key).

* Jika tidak ditemukan node dengan nilai data yang sama dengan kunci (key), currentNode akan null. Dalam blok if, jika currentNode null, menandakan kunci (key) tidak ada dalam LinkedList, maka -1 dikembalikan.
* Jika ditemukan node dengan nilai data yang sama dengan kunci (key), currentNode tidak null. Pada blok else, nilai index dari node tersebut dikembalikan.

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

Jika argumen method getData() lebih besar dari jumlah node pada linked list, yang muncul biasanya adalah NullPointerException, karena kita mencoba mengakses node yang tidak ada.



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

Ketika node yang memiliki nilai data yang sama dengan kunci (key) ditemukan dan dihapus, break digunakan untuk keluar dari loop while

Jika baris break tersebut dihapus, iterasi akan terus berlanjut meskipun operasi penghapusan sudah dilakukan. Ini bisa mengakibatkan masalah, seperti mencoba mengakses currentNode.next setelah currentNode menjadi null, yang dapat menyebabkan NullPointerException atau iterasi yang tidak terbatas

**Tugas Praktikum**

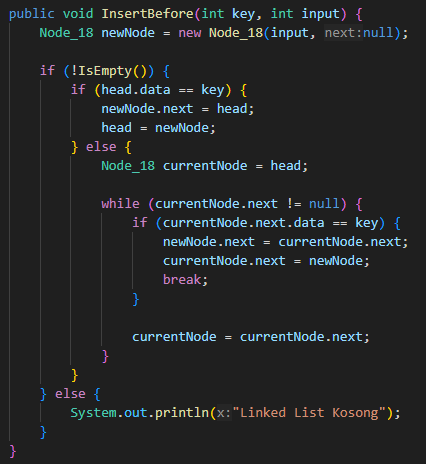
1. Implementasikan method-method berikut pada class LinkedList:

* insertBefore() untuk menambahkan node sebelum keyword yang diinginkan
* insertAt(int index, int key) untuk menambahkan node pada index tertentu
* removeAt(int index) untuk menghapus node pada index tertentu

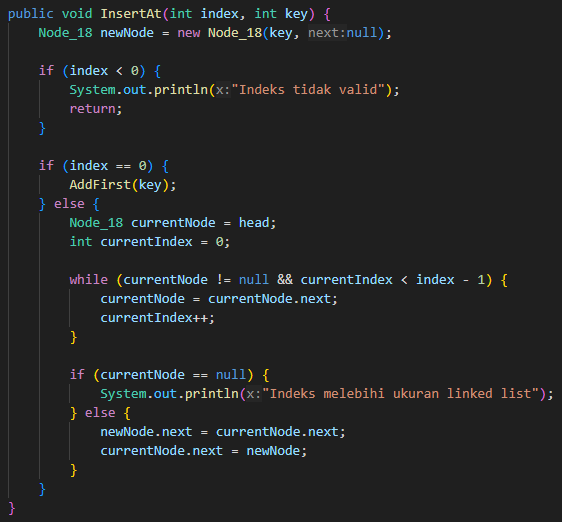
1. Dalam suatu game scavenger hunt, terdapat beberapa point yang harus dilalui peserta untuk menemukan harta karun. Setiap point memiliki soal yang harus dijawab, kunci jawaban, dan pointer ke point selanjutnya. Buatlah implementasi game tersebut dengan linked list.

Jawaban

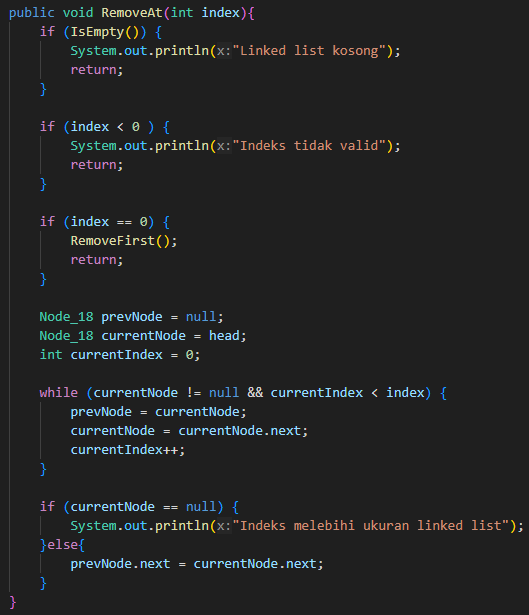
1a InsertBefore



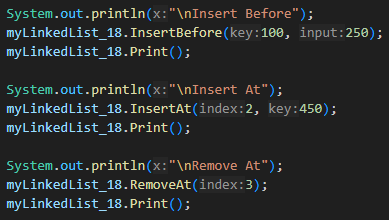
1b InsertAt

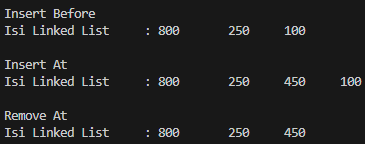


1c RemoveAt

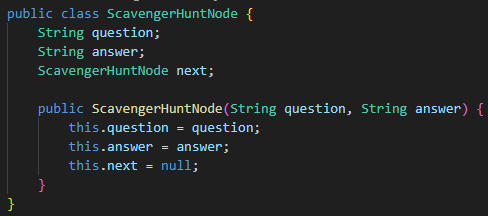


1d Output

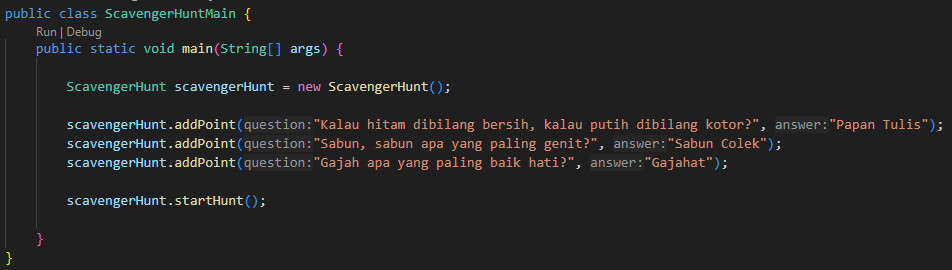




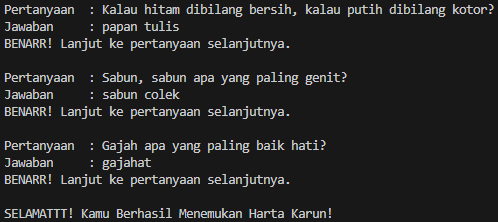
2a Program







2b Output



**Repository : <https://github.com/zidnafaz/Praktikum-Algoritma-Struktur-Data>**