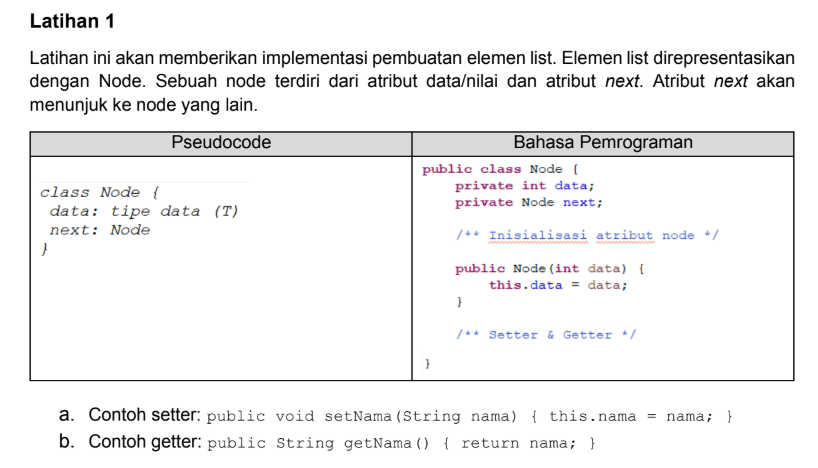
* **NAMA :** CHANDRA HARKAT RAHARJA  
  NPM : 233040089  
  KELAS : PRAKTIKUM PEMROGRAMAN I (B)  
  GITHUB : [PP12025\_B\_233040089/TugasDanPertemuanCoding/TugasPertemuan3 at main · ComradeChandra/PP12025\_B\_233040089](https://github.com/ComradeChandra/PP12025_B_233040089/tree/main/TugasDanPertemuanCoding/TugasPertemuan3)

Latihan-1



Penjelasan:  
 **Deklarasi Kelas dan Atribut**

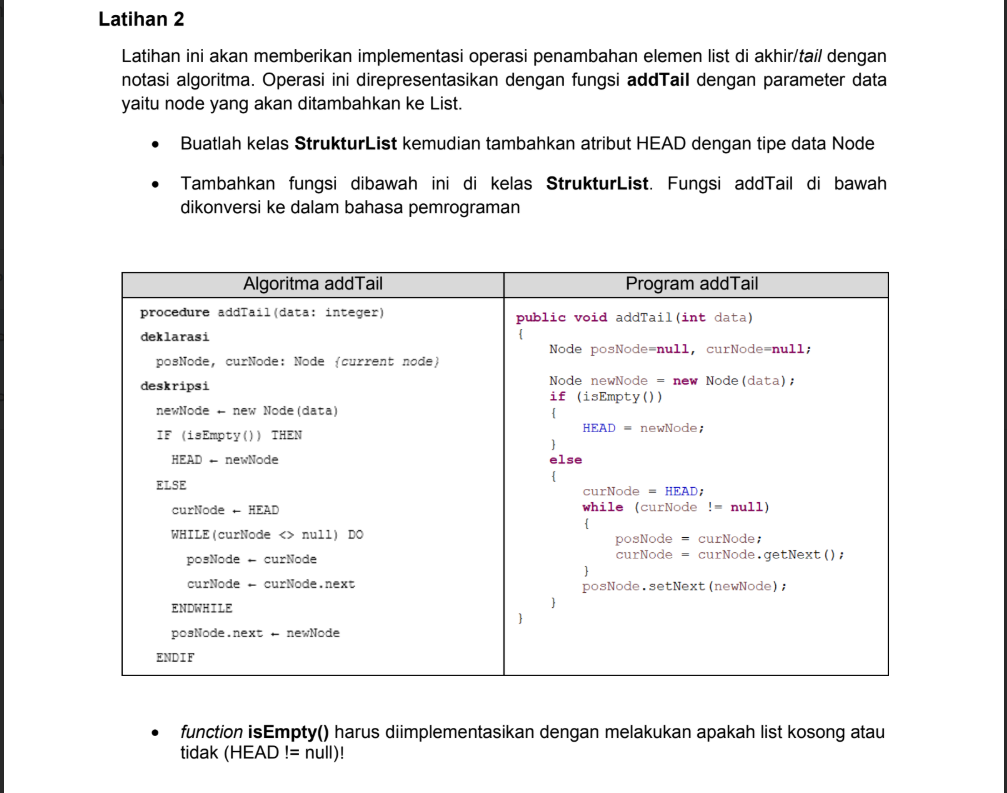
* public class Node {  
  Mendefinisikan kelas dengan nama "Node" yang bersifat publik, sehingga dapat diakses dari kelas lain.
* private int data;  
  Variabel privat "data" bertipe integer digunakan untuk menyimpan nilai atau informasi pada node.
* private Node next;  
  Variabel privat "next" bertipe Node menyimpan referensi ke node berikutnya. Ini memungkinkan kita menghubungkan node satu dengan node lainnya membentuk linked list.

**Konstruktor**

* public Node(int data) {  
  Konstruktor ini dipanggil saat kita membuat objek Node baru. Parameter "data" digunakan untuk menginisialisasi nilai yang ingin disimpan.
* this.data = data;  
  Baris ini mengassign nilai dari parameter ke variabel instance "data".  
  (Catatan: Variabel "next" tidak diinisialisasi di sini, sehingga secara default bernilai null, yang berarti
* node ini belum terhubung ke node lain.)

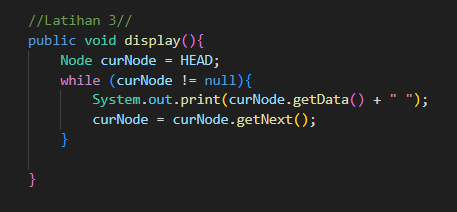
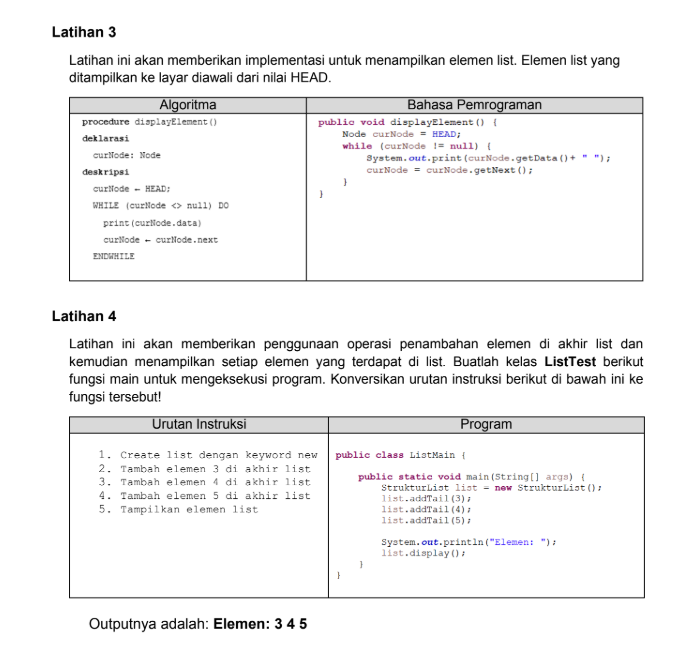
**Getter dan Setter untuk Atribut**

* Getter untuk data:  
  public int getData() { return data; }  
  Method ini mengembalikan nilai yang tersimpan dalam variabel "data", sehingga bisa diakses dari luar kelas.
* Setter untuk data:  
  public void setData(int data) { this.data = data; }  
  Method ini mengatur atau mengganti nilai dalam variabel "data" dengan nilai baru yang diberikan.
* Getter untuk next:  
  public Node getNext() { return next; }  
  Method ini mengembalikan referensi ke node berikutnya, sehingga kita dapat mengetahui node mana yang terhubung setelah node ini.
* Setter untuk next:  
  public void setNext(Node next) { this.next = next; }  
  Method ini digunakan untuk menetapkan referensi ke node berikutnya, menghubungkan node ini dengan node lain dalam linked list.

-Latihan 2  
  

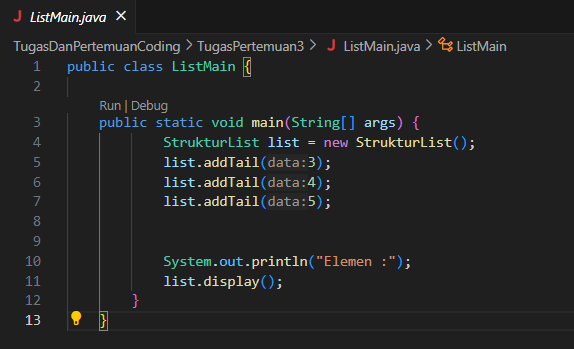

Penjelasan:

* **Deklarasi Kelas dan Atribut**
* public class StrukturList { ... }  
  Kelas StrukturList digunakan untuk menyimpan dan mengelola node-node dalam linked list.
* Node HEAD;  
  Variabel HEAD bertipe Node menandakan awal (head) dari linked list. Jika HEAD bernilai null, berarti list masih kosong.
* **Metode isEmpty()**
* public boolean isEmpty() { return (HEAD == null); }  
  Metode ini memeriksa apakah linked list kosong dengan melihat apakah HEAD bernilai null.
  + Jika HEAD == null, maka list kosong dan metode mengembalikan true.
  + Jika HEAD != null, maka list tidak kosong dan metode mengembalikan false.
* **Metode addTail(int data)**
* public void addTail(int data) { ... }  
  Metode ini menambahkan node baru di bagian akhir (tail) dari linked list.
* Node posNode = null, curNode = null;  
  Variabel posNode dan curNode dideklarasikan untuk membantu penelusuran hingga ke ujung list.
* Node newNode = new Node(data);  
  Membuat node baru dengan data yang diterima sebagai parameter.
* if (isEmpty()) { HEAD = newNode; } else { ... }
  + Jika list masih kosong (isEmpty() bernilai true), maka node baru langsung menjadi HEAD.
  + Jika list tidak kosong, penelusuran dilakukan untuk mencari node terakhir.
* curNode = HEAD; while (curNode != null) { posNode = curNode; curNode = curNode.getNext(); }
  + curNode diawali dari HEAD.
  + Selama curNode tidak null, simpan curNode saat ini ke posNode, lalu bergerak ke node berikutnya.
  + Ketika perulangan berakhir, posNode akan berisi node terakhir di list.
* posNode.setNext(newNode);
  + Menghubungkan node terakhir (posNode) dengan node baru (newNode).
  + Dengan demikian, node baru ditempatkan di posisi paling akhir (tail).

-Latihan 3 dan Latihan 4  


Penjelasan:

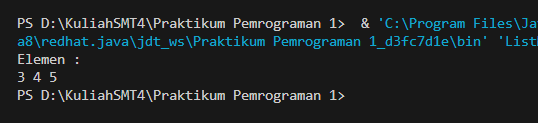
* Node curNode = HEAD;
* Variabel curNode diinisialisasi agar menunjuk ke node pertama, yaitu HEAD.
* while (curNode != null) { ... }
* Perulangan akan berjalan selama curNode belum mencapai null.
* null menandakan bahwa kita telah sampai di akhir linked list.
* System.out.print(curNode.getData() + " ");
* Mencetak nilai data (yang disimpan pada setiap node) diikuti spasi.
* curNode = curNode.getNext();
* Menggeser curNode ke node berikutnya, sehingga perulangan dapat terus bergerak maju sampai akhir list.

Latihan – 4  


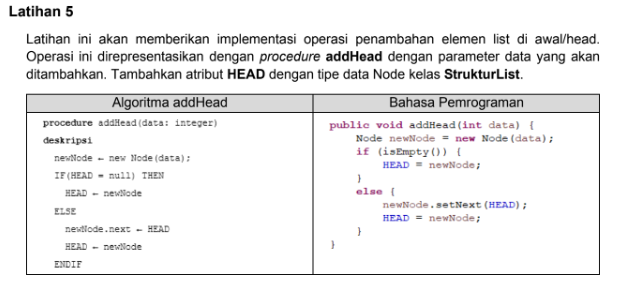
Penjelasan:

* **Deklarasi Kelas dan Method Main**
* Kelas **ListMain** dideklarasikan sebagai kelas publik.
* Di dalamnya terdapat **method main** yang menjadi titik awal eksekusi program.
* M**embuat Objek StrukturList**
* Pada baris StrukturList list = new StrukturList();, dibuat objek list yang merupakan instansiasi dari kelas StrukturList.
* Objek ini akan menyimpan data dalam bentuk linked list.
* **Menambahkan Elemen di Akhir List**
* Tiga pemanggilan list.addTail(3), list.addTail(4), dan list.addTail(5) menambahkan node dengan nilai 3, 4, dan 5 di akhir list.
* Akibatnya, isi list secara berurutan adalah 3 → 4 → 5.
* **Menampilkan Elemen List**
* System.out.println("Elemen :"); mencetak teks “Elemen :” sebagai penanda.
* list.display() memanggil method yang akan mencetak semua data di dalam list, mulai dari node pertama hingga node terakhir.

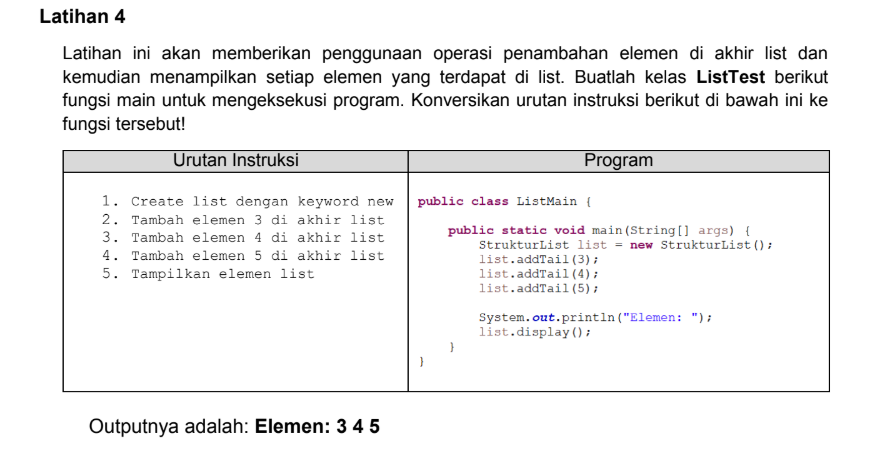
Hasil Outputnya adalah:



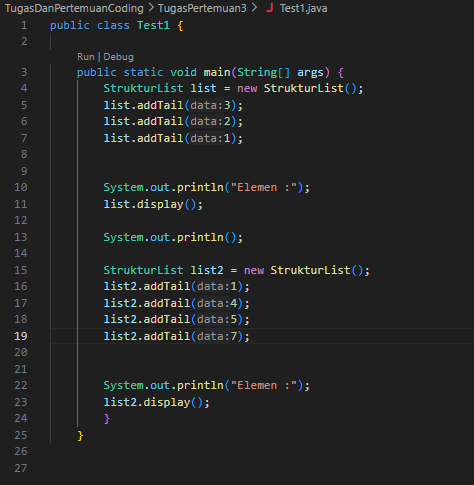
-Latihan 5

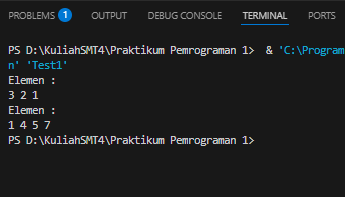


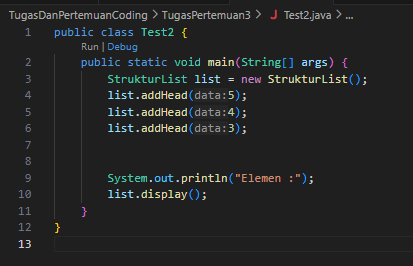
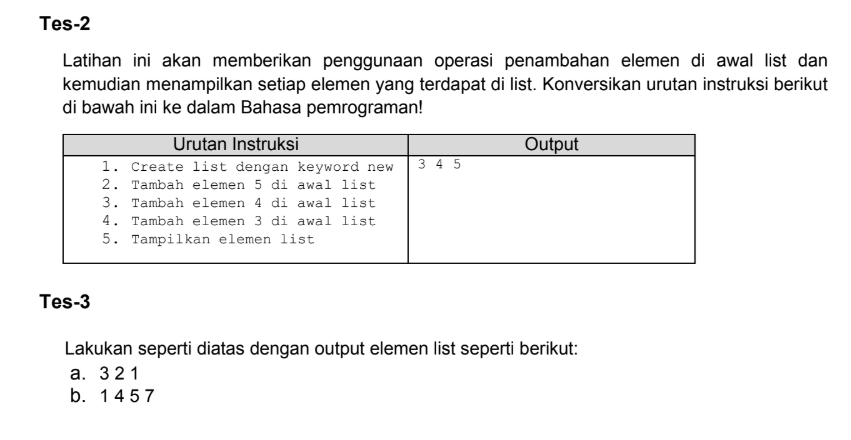
* **Membuat Node Baru**
* Node newNode = new Node(data);  
  Baris ini membuat objek node baru dengan nilai data yang diterima sebagai parameter.
* **Cek Apakah List Kosong**
* if (isEmpty()) { ... } else { ... }  
  Metode isEmpty() umumnya memeriksa apakah HEAD == null.
* Jika HEAD == null, berarti list masih kosong, sehingga newNode langsung dijadikan HEAD.
* Jika list tidak kosong, kita perlu menghubungkan node baru ke node yang sebelumnya menjadi head.
* **Menautkan Node Baru di Depan**
* newNode.setNext(HEAD);  
  Node baru (newNode) diarahkan untuk menunjuk ke node yang sebelumnya menjadi head.
* HEAD = newNode;  
  Setelah itu, head diperbarui sehingga mengarah ke newNode.

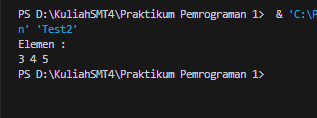
1. TES-1:   
  
Lakukan seperti diatas dengan output elemen list seperti berikut:

a. 3 2 1

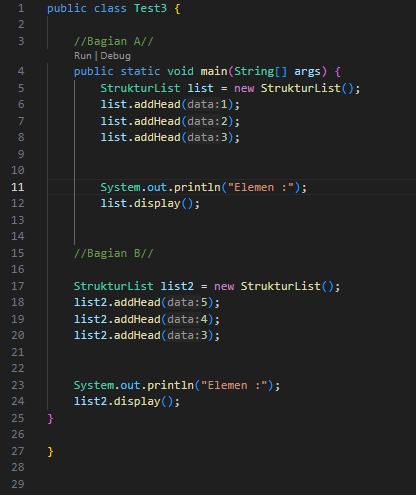
b. 1 4 5 7  
  
Jawab:  
  
Penjelasan:  
Di Class test 1 di dalam program main, membuat struktur list baru Bernama list. Lalu list ditambah di bagian akhir nomor, dengan urutan 3, 2 1. Setelah itu diperlihatkan elemenya di bagian output  
  
  
 Begitu juga dengan struktur list 2, code ini berfungsi sama dengan code sebelumnya hanya saja dengan list yang berberda, yakni 1,4,5,7. Hasil Querry akan seperti di gambar berikut:



2. TES-2 dan TES-3  
  
  
Jawab:  
-Tes-2

  
-Penjelasan:  
Di Dalam Class Test-2 di dalam program main, ini berfungsi untuk membuat struktur list baru dengan nama list. Setelah itu code ini akan menambahkan nilai di awal dengan urutan 5, 4, 3. Ketika di Run, hasilnya akan Seperti gambar berikutnya.

TES-3:

Penjelasan:  
Hampir sama dengan code sebelumnya, code ini berfungsi untuk membuat list baru dengan nama list, hasil untuk query pertama adalah list dengan urutan 3, 2, 1.  
  
  
Sementara untuk bagian B, hanya dibedakan dengan baris code yang membuat list baru bernama list2, yang akan menghasilkan list dengan urutan 5, 4,3.   
  
Hasil untuk kedua code ada di gambar berikutnya.

