**JOBSHEET 12**

**Double Linked List**



**Name**

Sherly Lutfi Azkiah Sulistyawati

**NIM**

2341720241

**Class**

1I

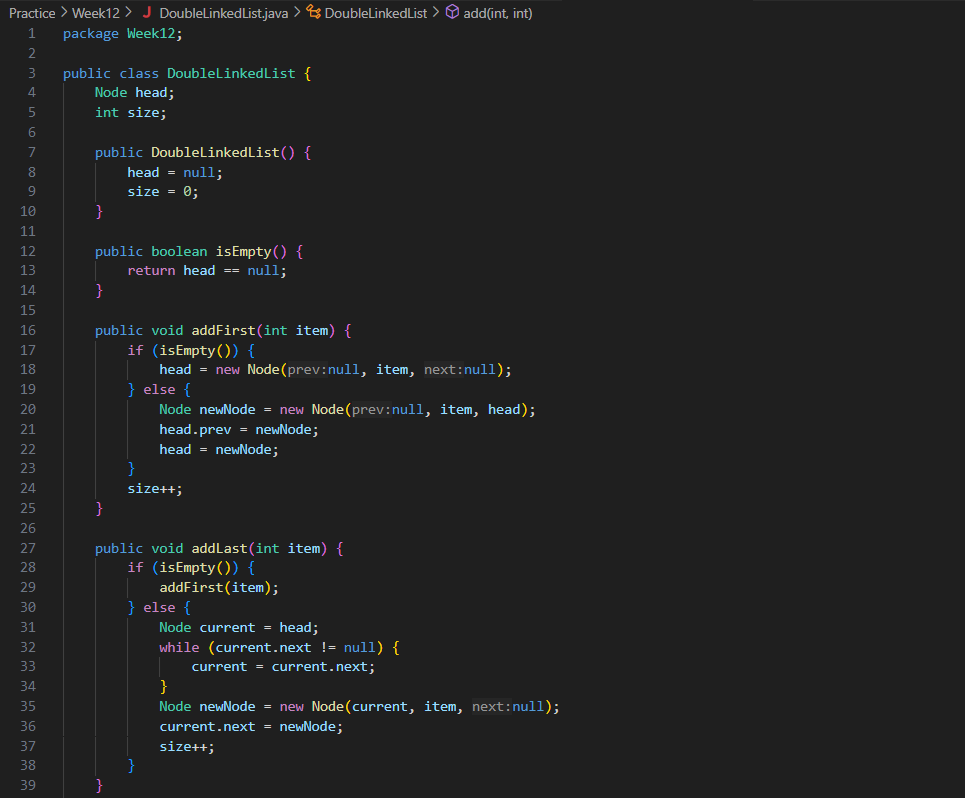
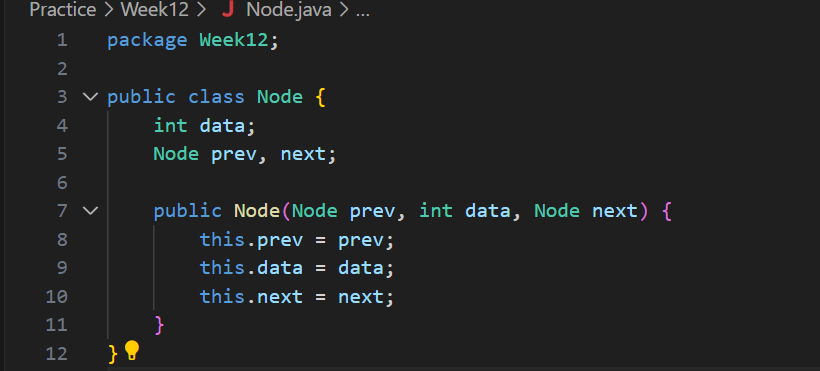
**Major**

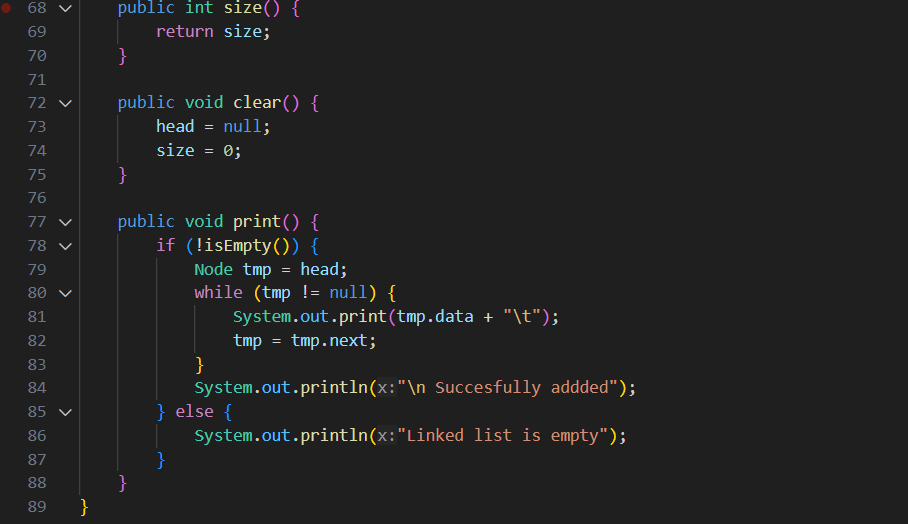
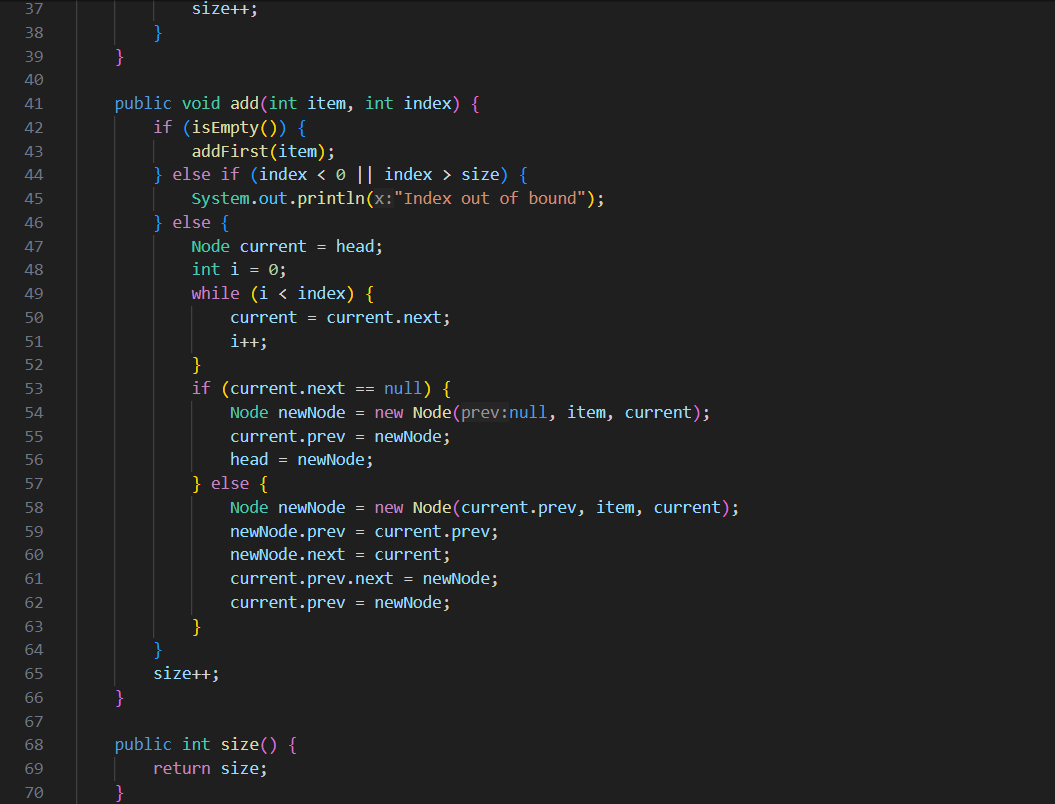
Information Technology

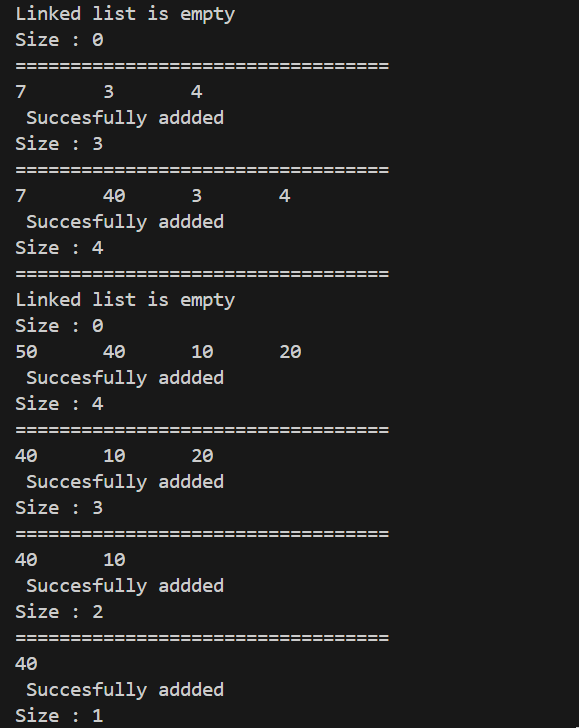
**Study Program**

D4 Informatics Engineering

**Lab Activity 1**

****

****

****

**Question**

1. What’s the difference between single linked list and double linked list?

 **Single Linked List (SLL)**:

* Each node contains a data part and a pointer to the next node in the sequence.
* Traversal can only be done in one direction (forward).
* It is simpler and uses less memory compared to DLL.

 **Double Linked List (DLL)**:

* Each node contains a data part, a pointer to the next node, and a pointer to the previous node.
* Traversal can be done in both directions (forward and backward).
* It provides more flexibility but uses more memory due to the extra pointer.

1. In **Node class**, what is the usage of attribute next and prev ?

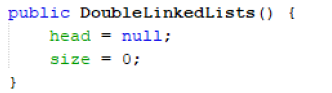
 **next**:

* Points to the next node in the linked list.
* Used to traverse the list in a forward direction.

 **prev**:

* Points to the previous node in the linked list.
* Used to traverse the list in a backward direction.

1. In constructor of **DoubleLinkedList class.** What’s the purpose of head and size attribute in this following code?



* **head:** represents the starting point of the linked list. Initializing it to null indicates the list is empty.
* **size:** keeps track of the number of elements in the linked list. Initializing it to 0 indicates the list is empty.

1. In method **addFirst(),** why do we initialize the value of Node object to be null at first? Node newNode = new Node(**null,** item, head);

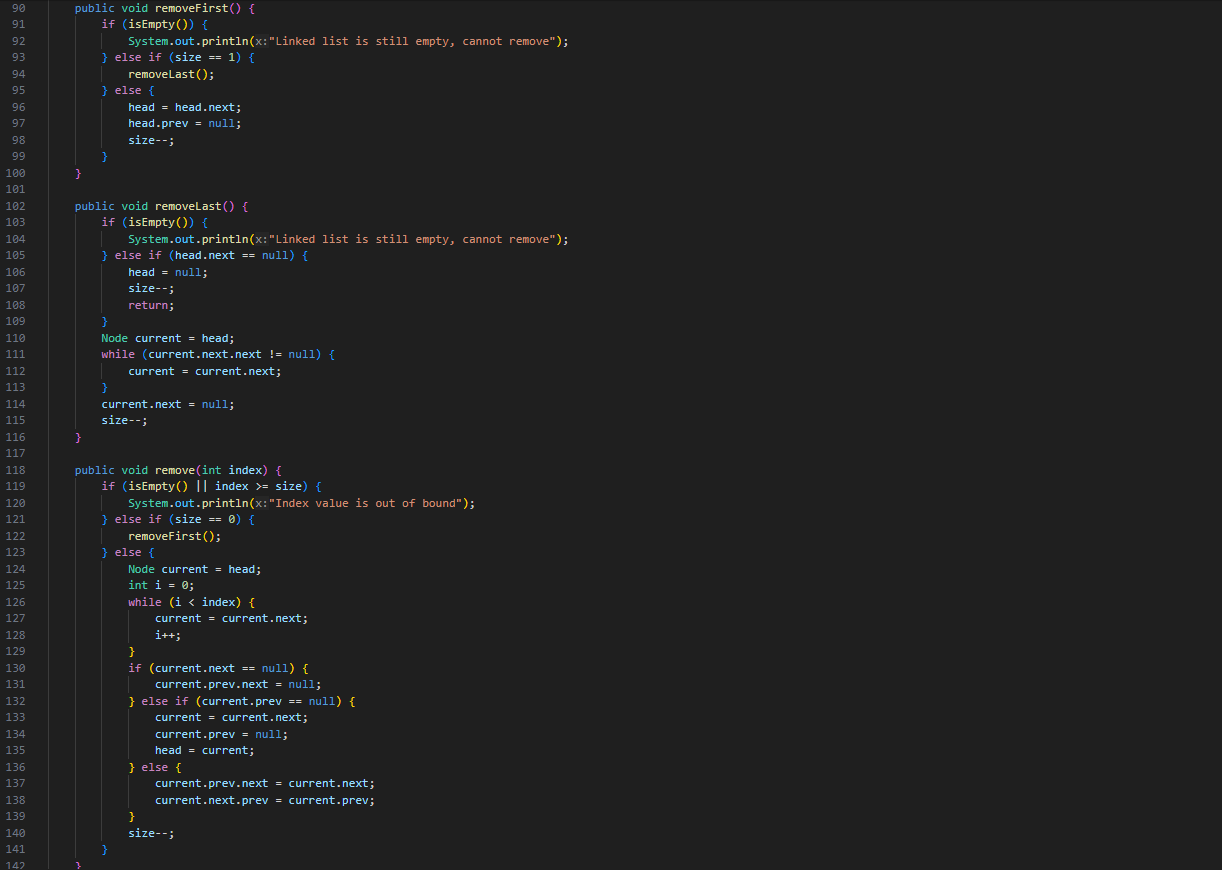
* When adding the first element to the list, the new node's prev is set to null because there are no nodes before it.
* The next is set to head because the new node will be placed at the front, pointing to the current head of the list.

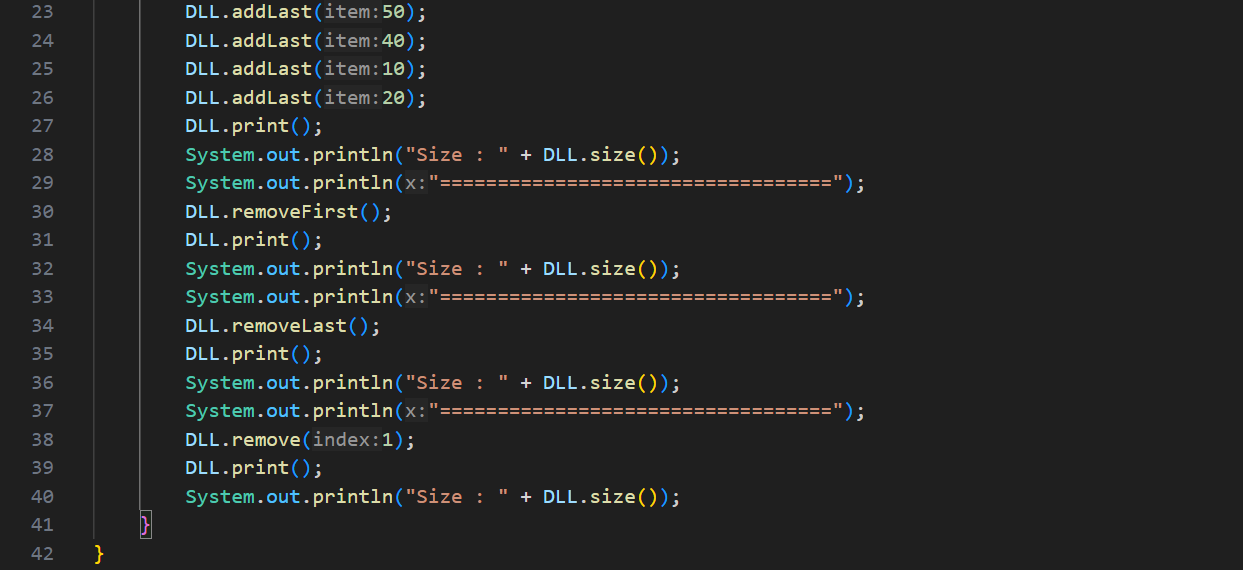
1. In method **addLast(),** what’s the purpose of creating a node object by passing the **prev** parameter with **current** and **next** with **null** ?

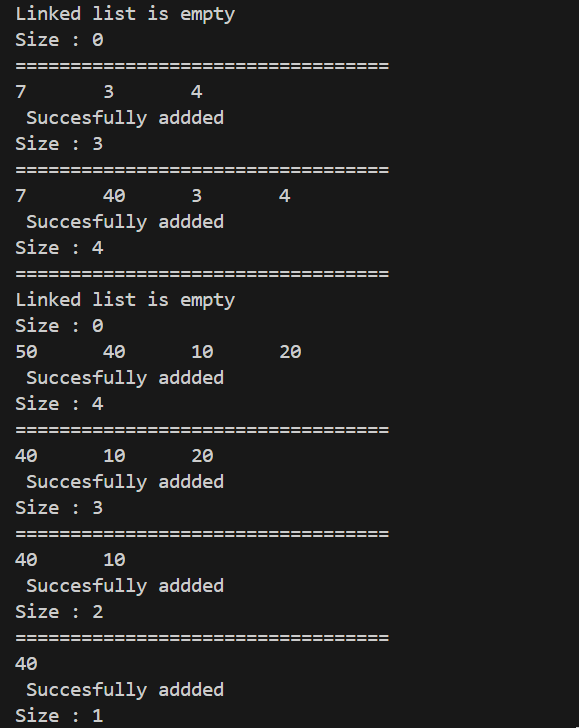
Node newNode = new Node(**current**, item, **null**);

* **current:** This represents the current last node of the list, so the prev pointer of the new node is set to current.
* **null:** The next pointer of the new node is set to null because it will be the new last node in the list.

**Lab Activity 2**

****

****

****

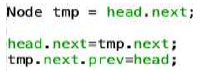
**Question**

1. What’s the meaning of these statements in removeFirst() method?

* **head = head.next:** Move the head pointer to the next node, effectively removing the first node.
* **head.prev = null:** Set the prev pointer of the new head node to null because it is now the first node.
* **size--:** Decrement the size of the list by 1.

1. How do we detect the position of the data that are in the last index in method **removeLast()**?

* Traverse the list until current.next.next is null, which means current.next is the last node. This allows you to adjust the pointers to remove the last node.

1. Explain why this program code is not suitable if we include it in **remove** command! 

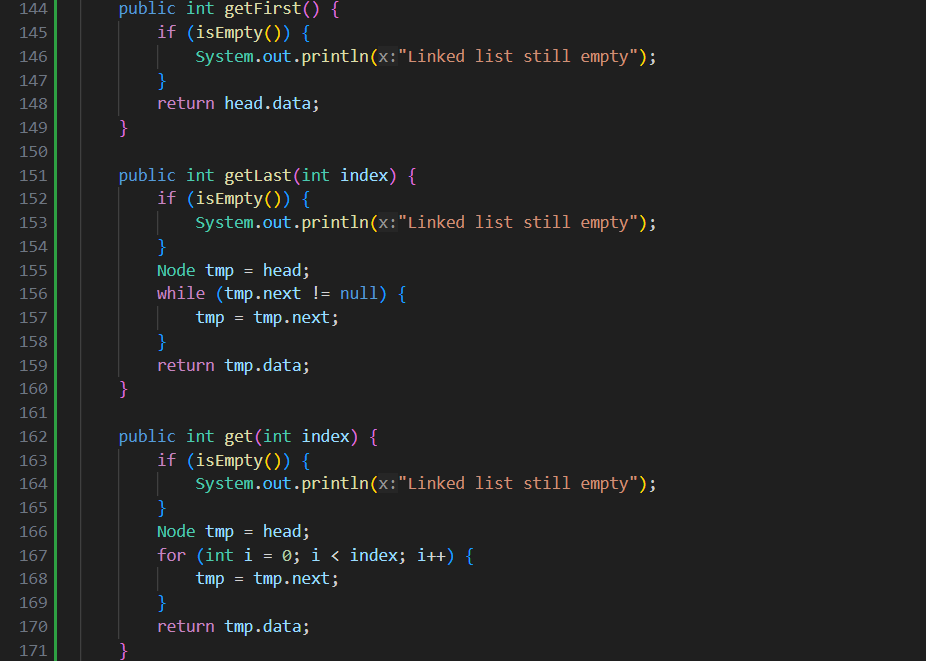
* This code only works if you are removing the second node from the list. It does not handle general cases or edge cases (e.g., if the list has only one node or if the node to be removed is not the second node).

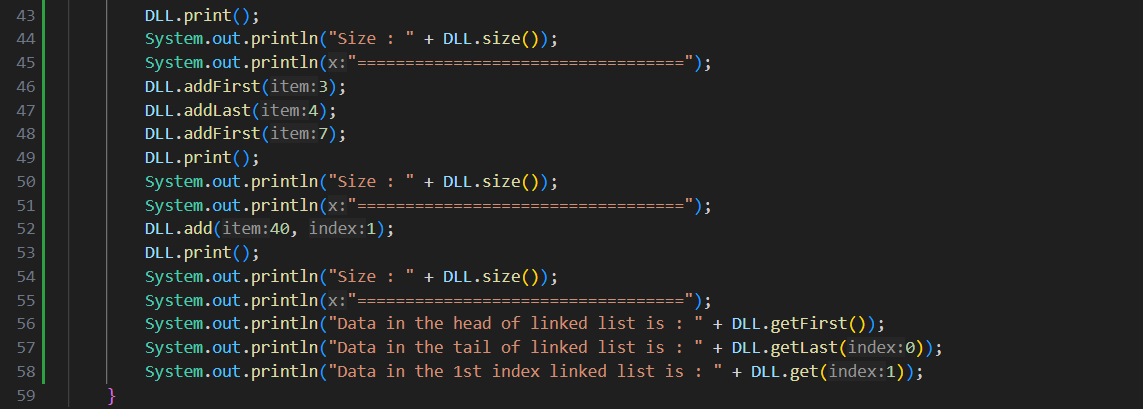
1. Explain what’s the function of this program code in method **remove**!

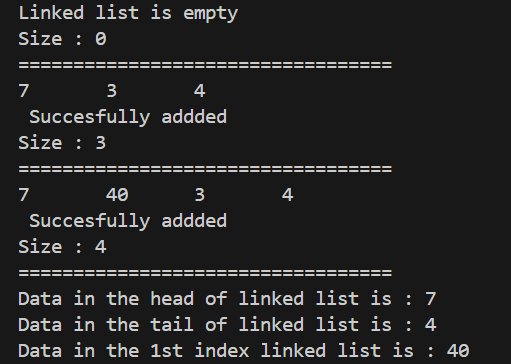


* **current.prev.next = current.next:** Adjust the next pointer of the previous node to skip the current node and point to the node after the current one.
* **current.next.prev = current.prev:** Adjust the prev pointer of the next node to skip the current node and point to the node before the current one.
* This effectively removes the current node from the list by linking the previous node directly to the next node.

**Lab Activity 3**

****

****

****

**Question**

1. What is the function of method **size()** in **DoubleLinkedList** class ?

* The size() method returns the number of elements in the list

1. How do we set the index in double linked list so that it starts from 1st index instead of 0th index?

* To make the index start from 1 instead of 0, adjust the indexing logic in the methods where indices are used:

public void add(int item, int index) {

index--; // Decrease index by 1

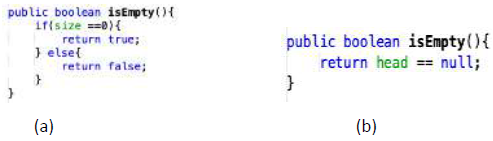
// Rest of the code remains the same

}

1. Please explain the difference between method **Add()** in double linked list and single linked list !

* **Double Linked List**:
* Handles both prev and next pointers.
* More complex insertion logic to maintain both pointers.
* **Single Linked List**:
* Only handles the next pointer.
* Simpler insertion logic.

1. What’s the logic difference of these 2 following codes?



* 1. Checks if the size of the list is 0 to determine if the list is empty.
  2. Checks if the head of the list is null to determine if the list is empty.

**Assignment**

1. Create a program with double linked list implementation that allows user to choose a menu as following image! The searching uses sequential search approach and the program should be able to sort the data in descending order. You may any choose sorting approach you prefer (bubble sort, selection sort, insertion sort, or merge sort)

**Adding a data**

****

**Add data in specified index and display the result**

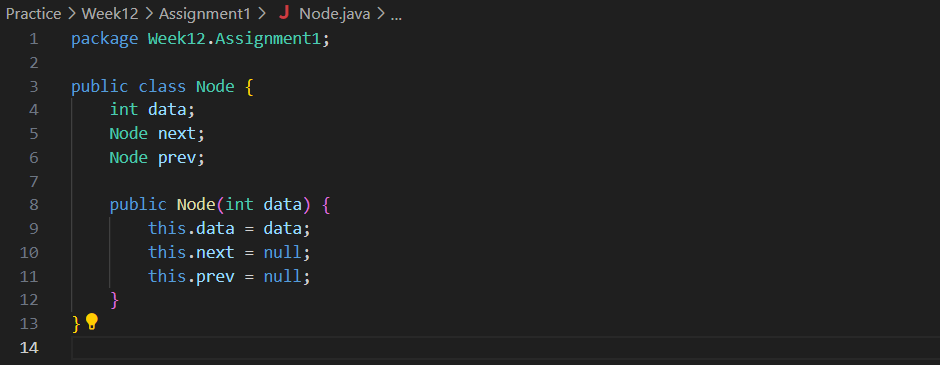
****

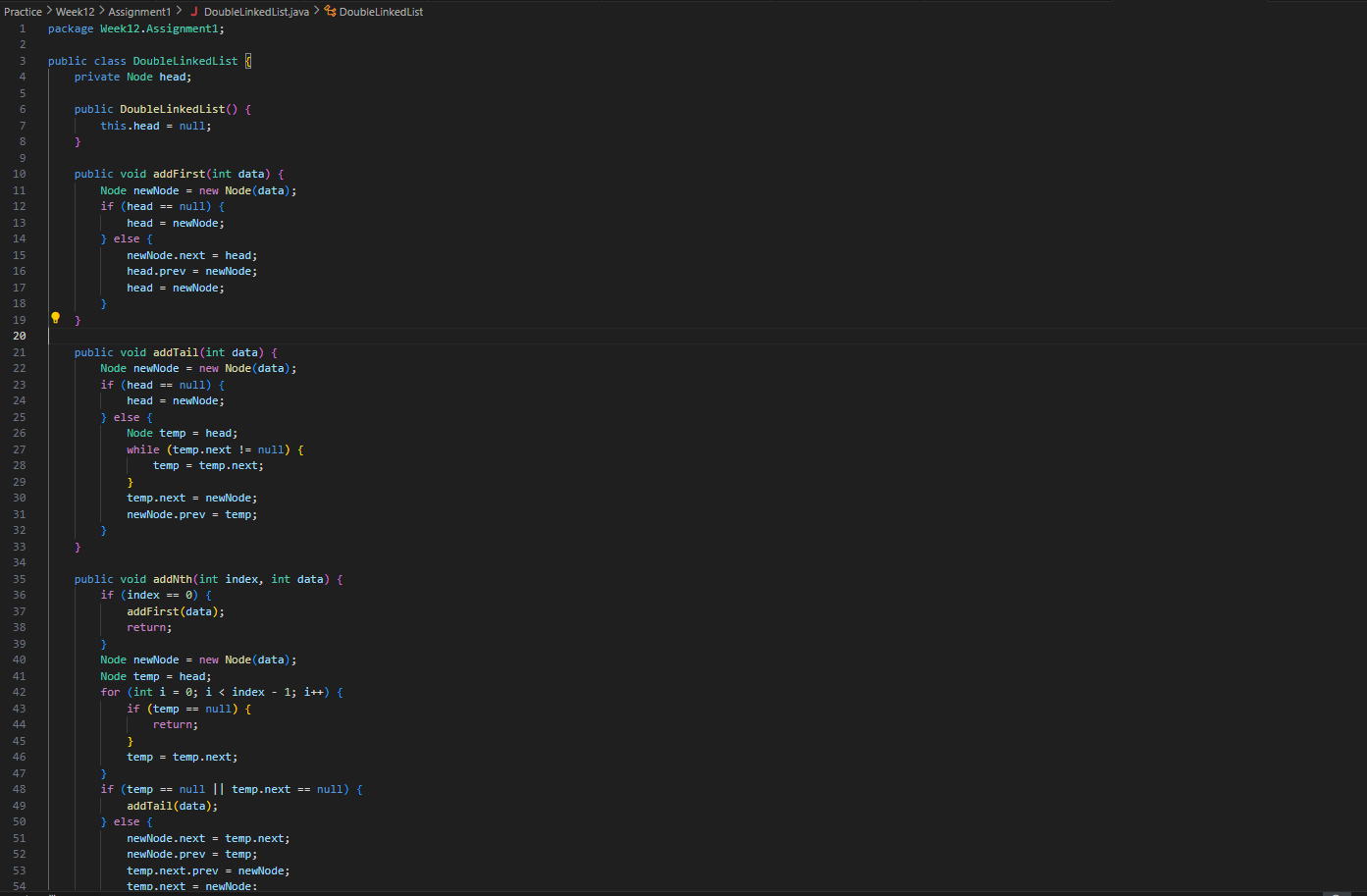
**Search Data**

****

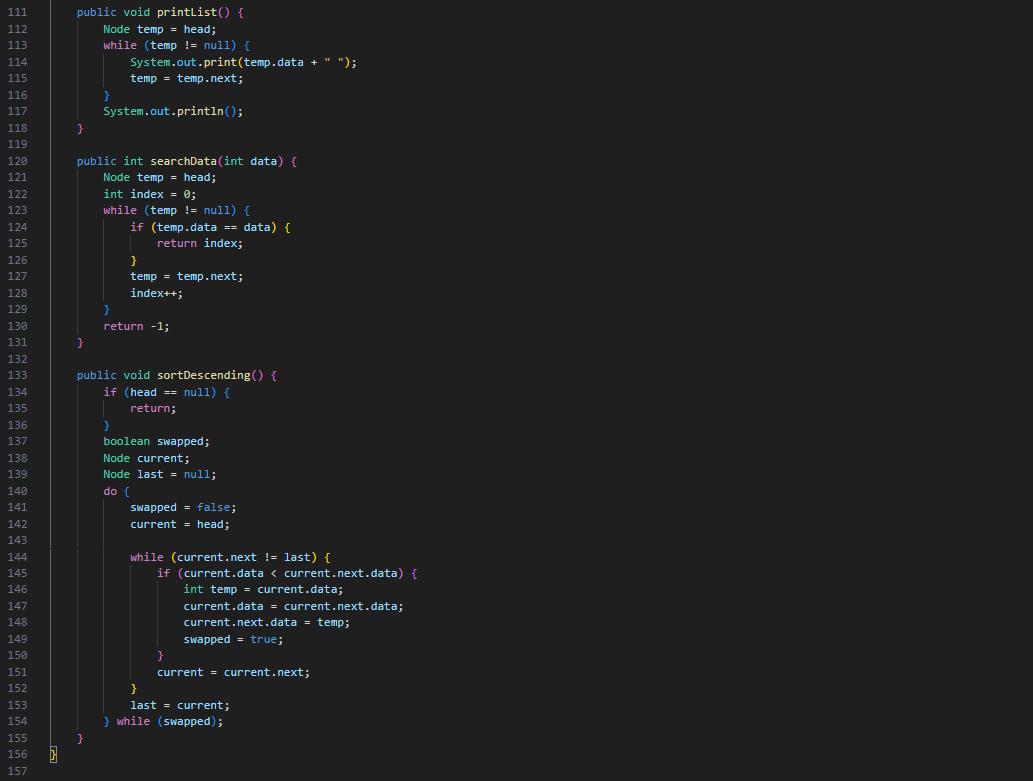
**Sorting Data**

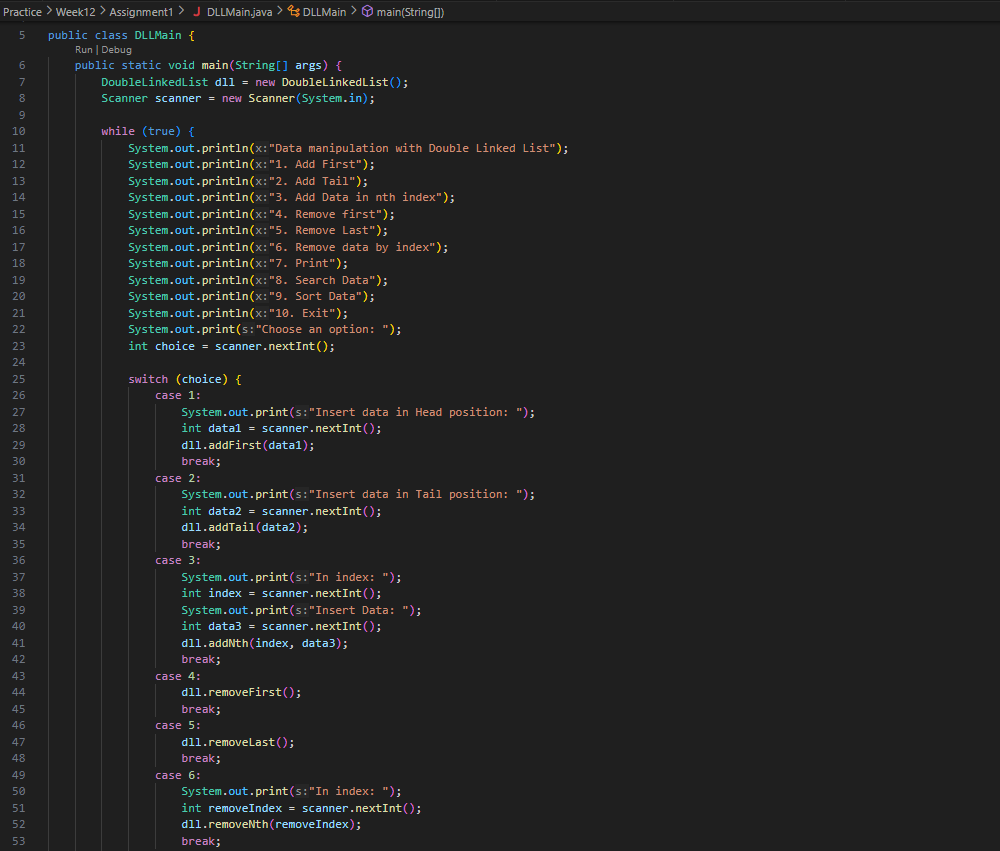
****

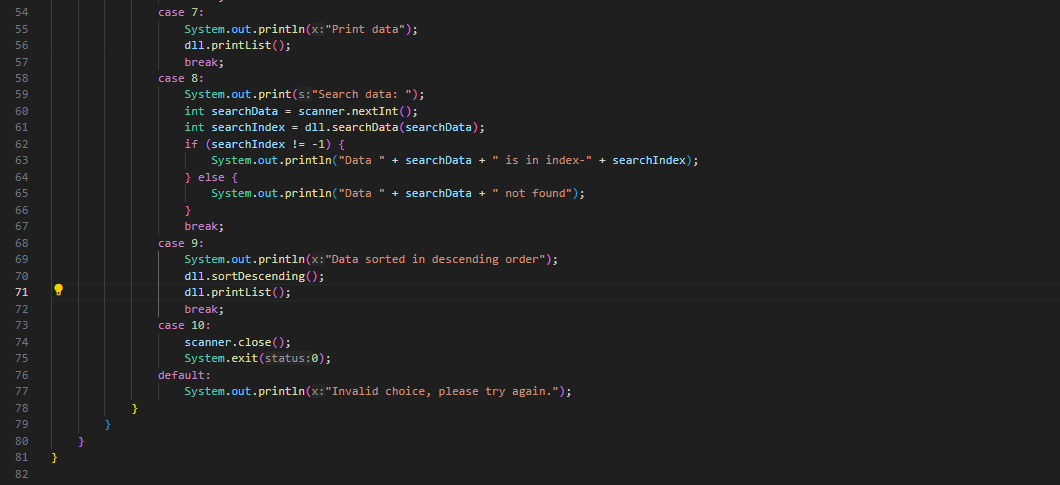
****

****

****

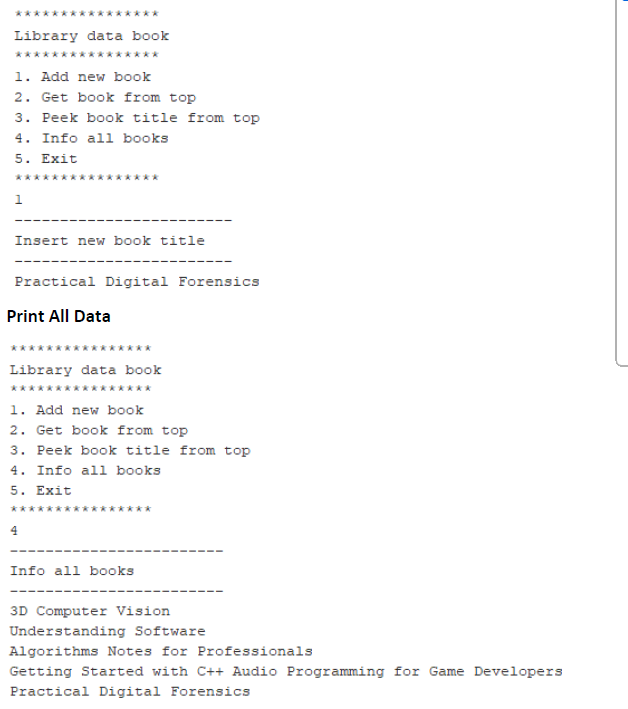
****

****

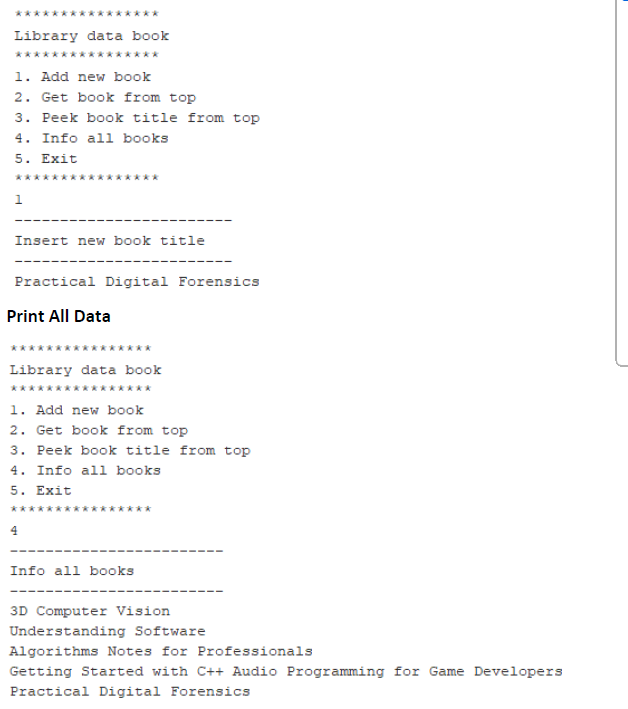
****

1. We are required to create a program which Implement Stack using double linked list. The features are described in following illustrations:

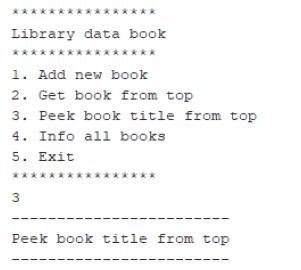
**Initial menu and add Data (push)**

****

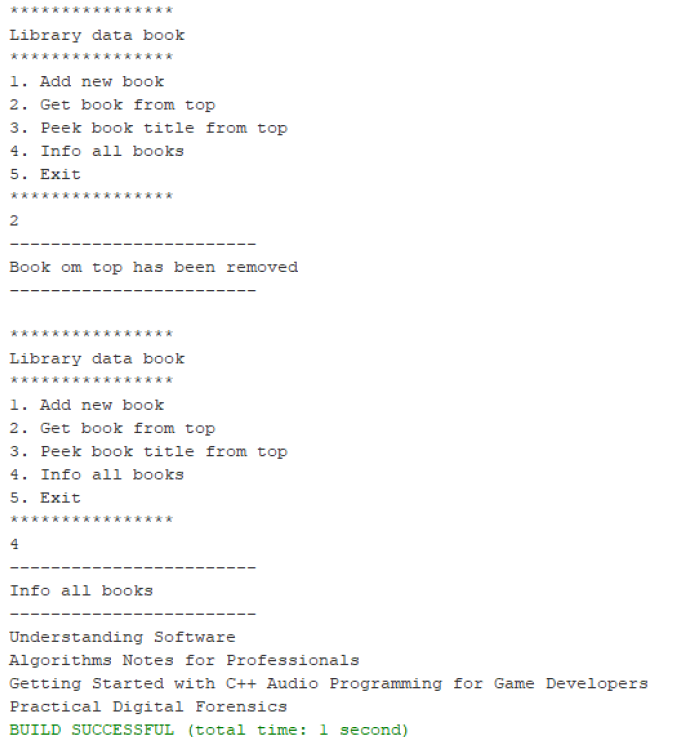
**Print All Data**

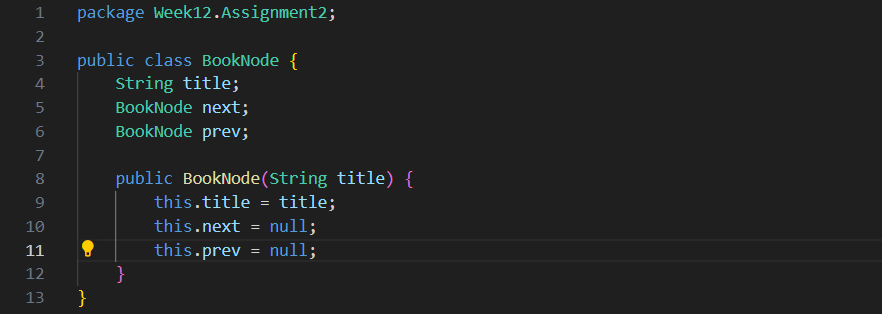
****

**See the data on top of the stack**

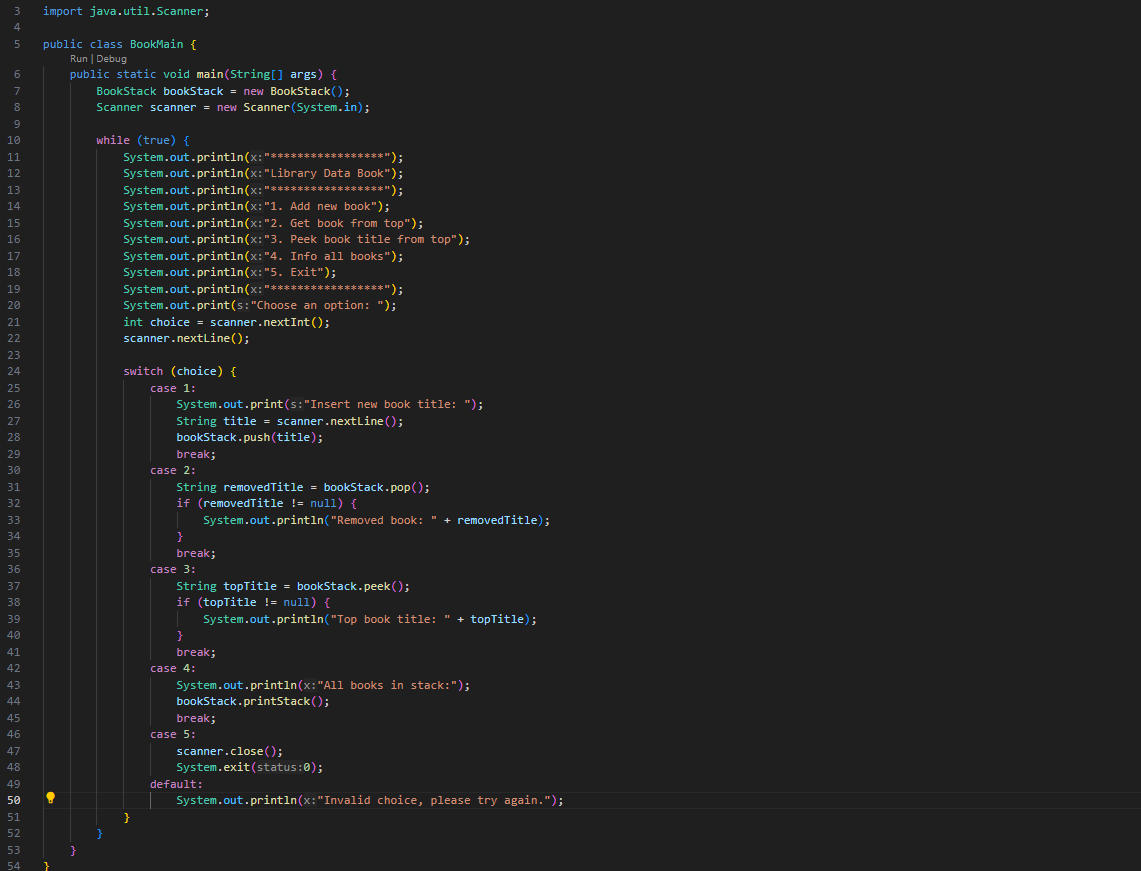
****

**Pop the data from the top of the stack**

****

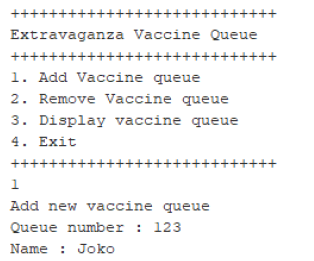
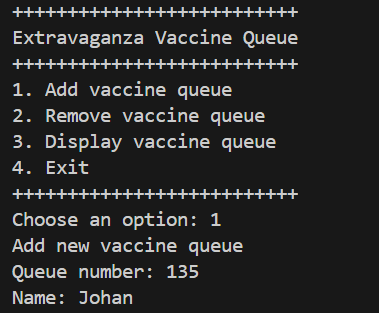
****

****

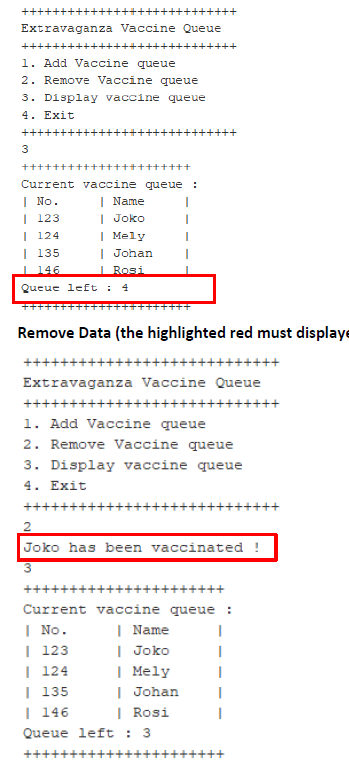
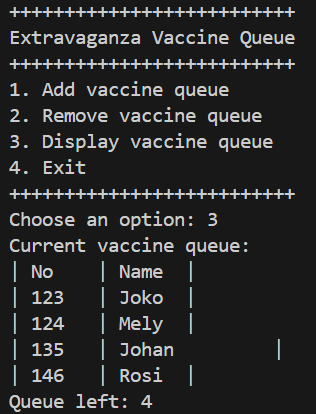
****

1. Create a program that helps vaccination process by having a queue algorithm alongside with double linked list as follows **(the amount left of queue length in menu print(3) and recent vaccinated person in menu Remove data (2) should be displayed)**

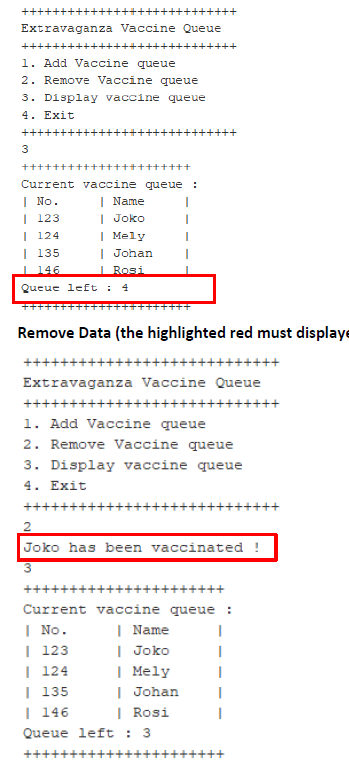
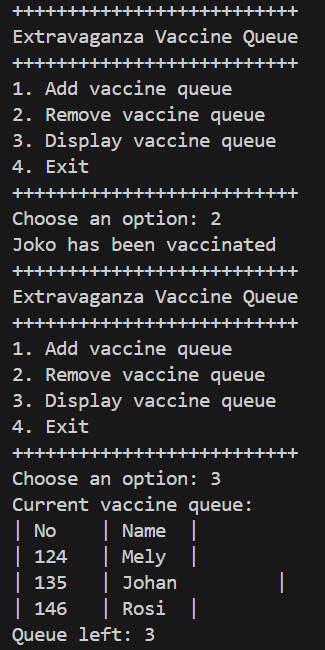
**Initial menu and adding a data**

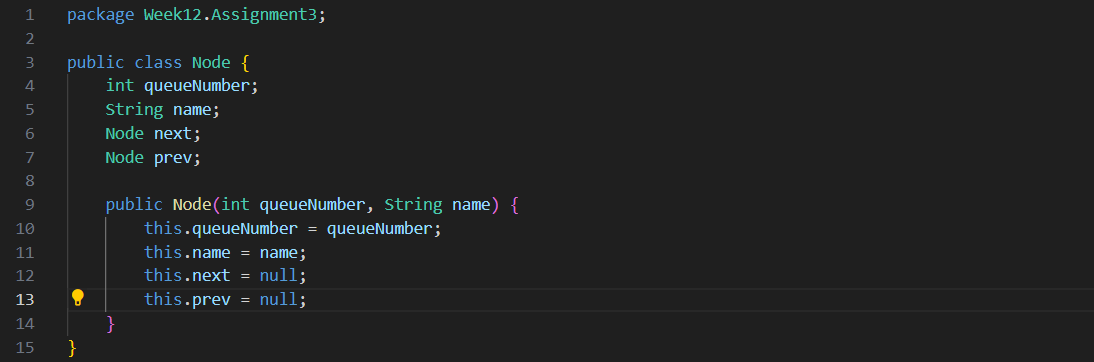
**** ****

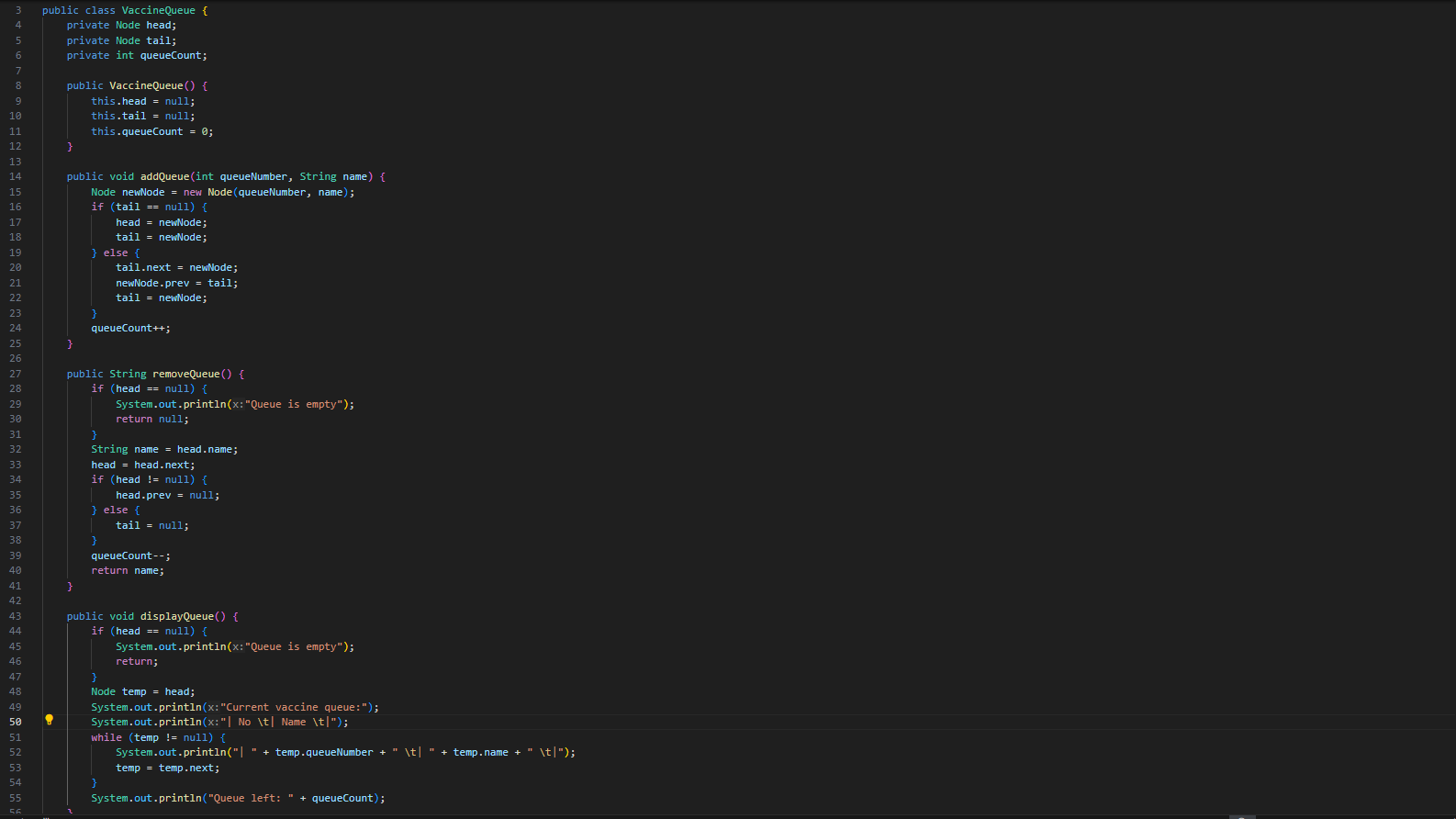
**Print data (notice the highlighted red in the result)**

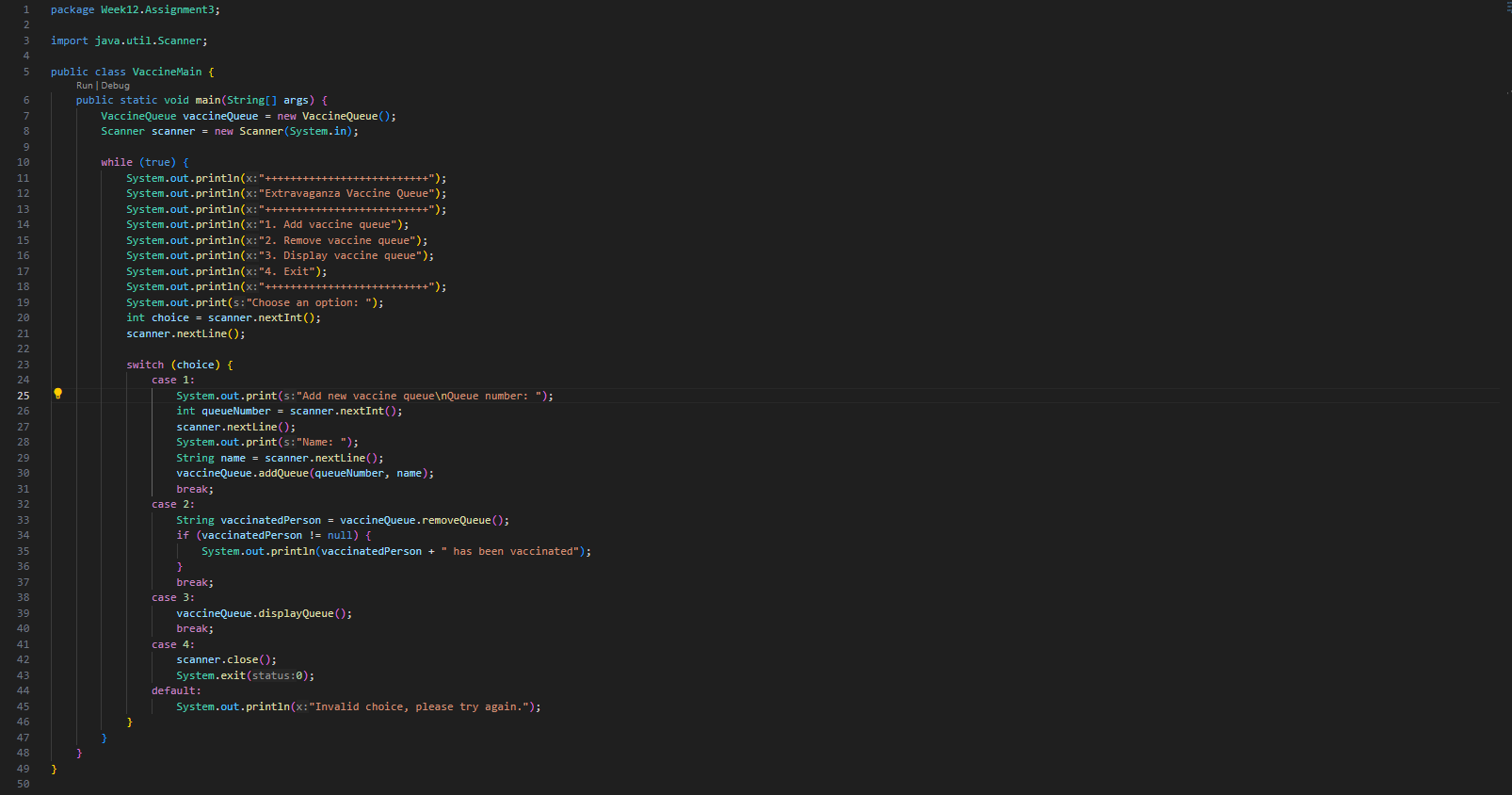
**** ****

**Remove Data (the highlighted red must displayed in the console too)**

**** ****

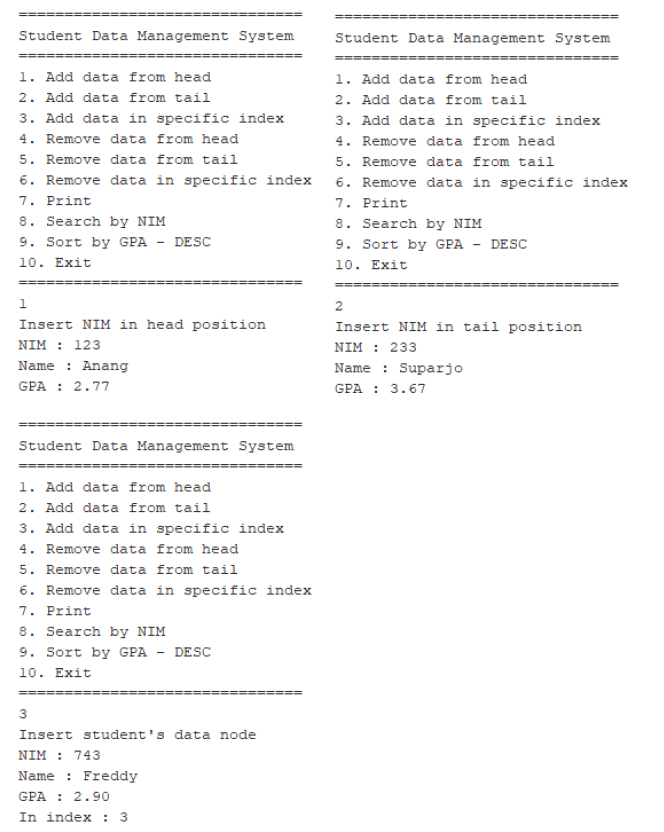




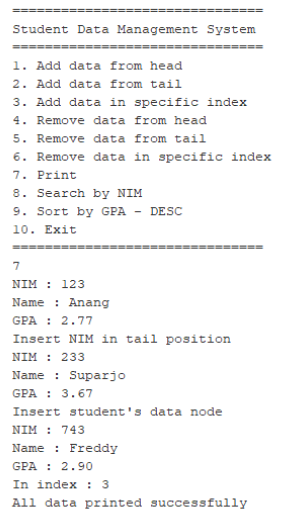


1. Create a program implementation that list students score. Each student’s data consist of their nim, name, and gpa. The program should implement double linked list and should be able to search based on NIM and sort the GPA in descending order. **Students class must be implemented in this program**

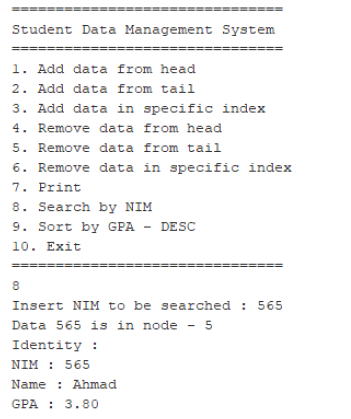
**Initial menu and adding data**

****

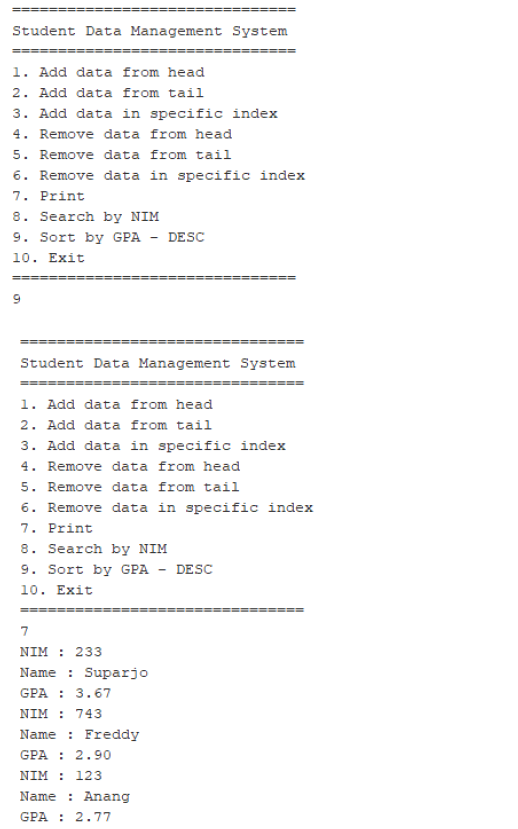
**Printing data**

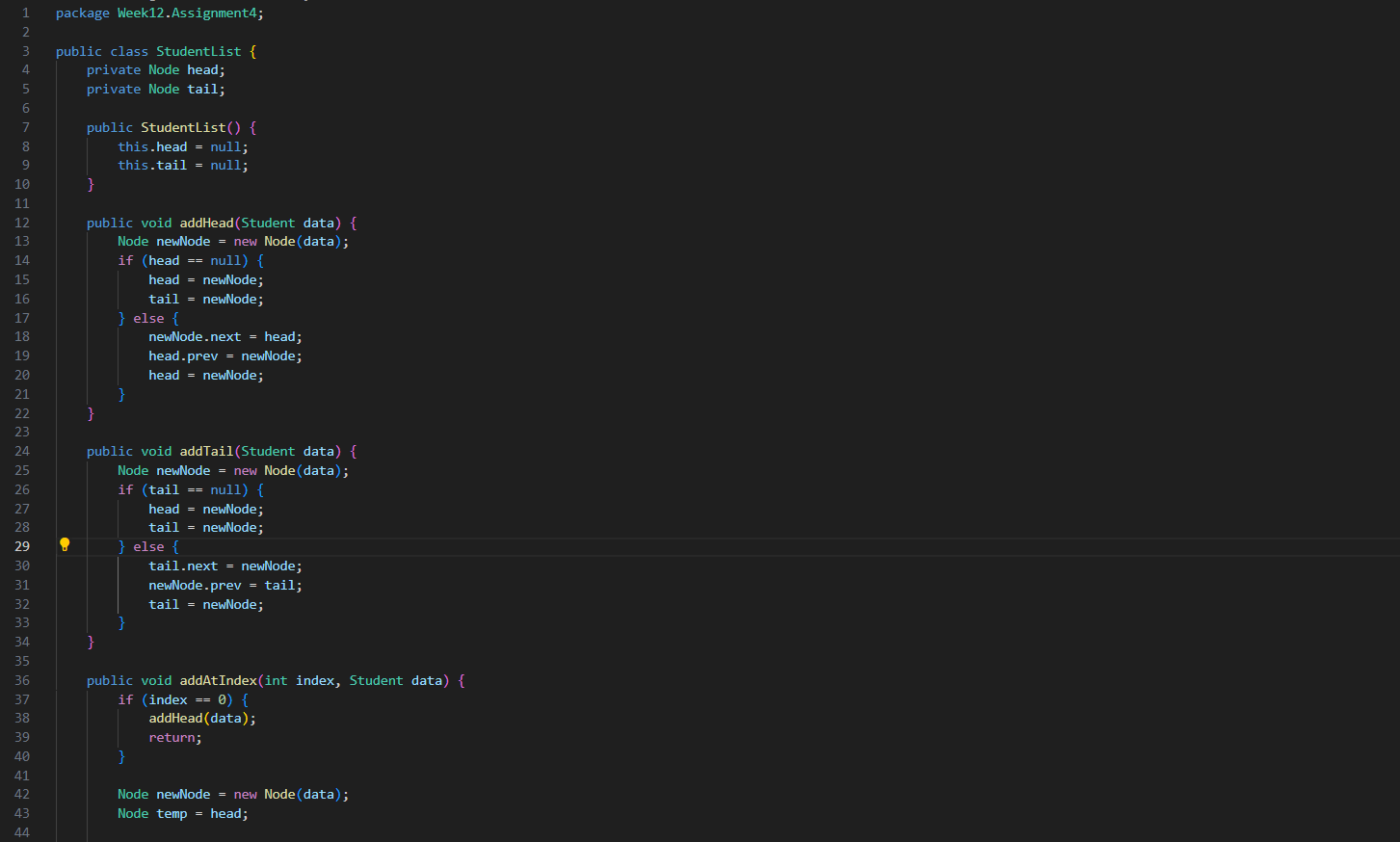
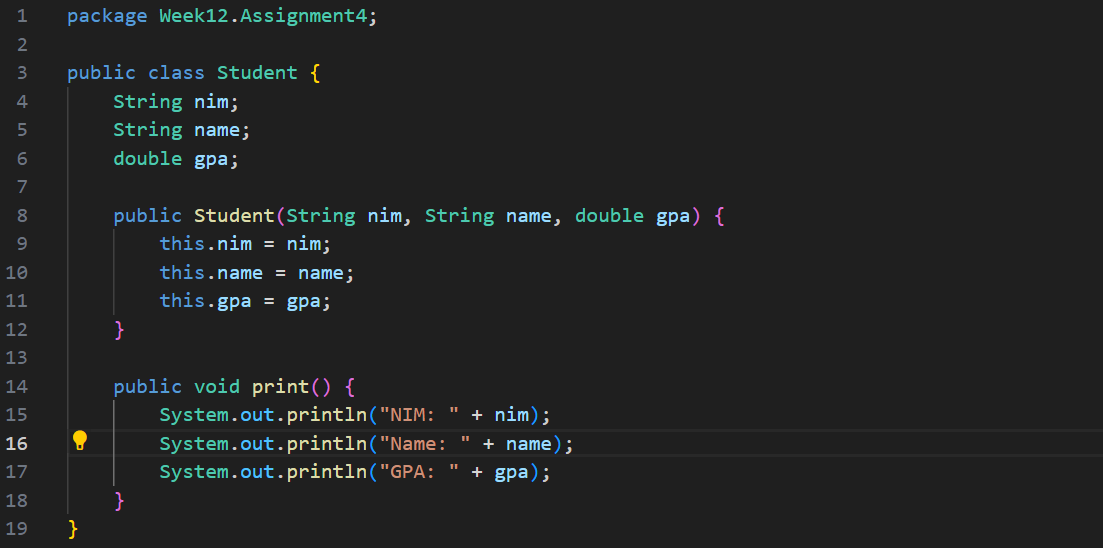
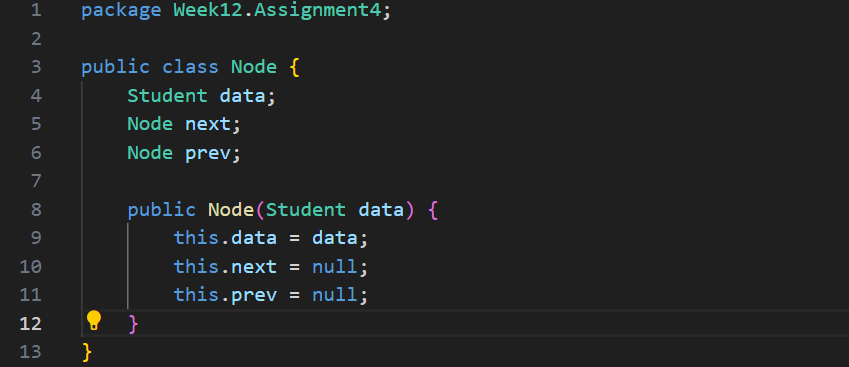
****

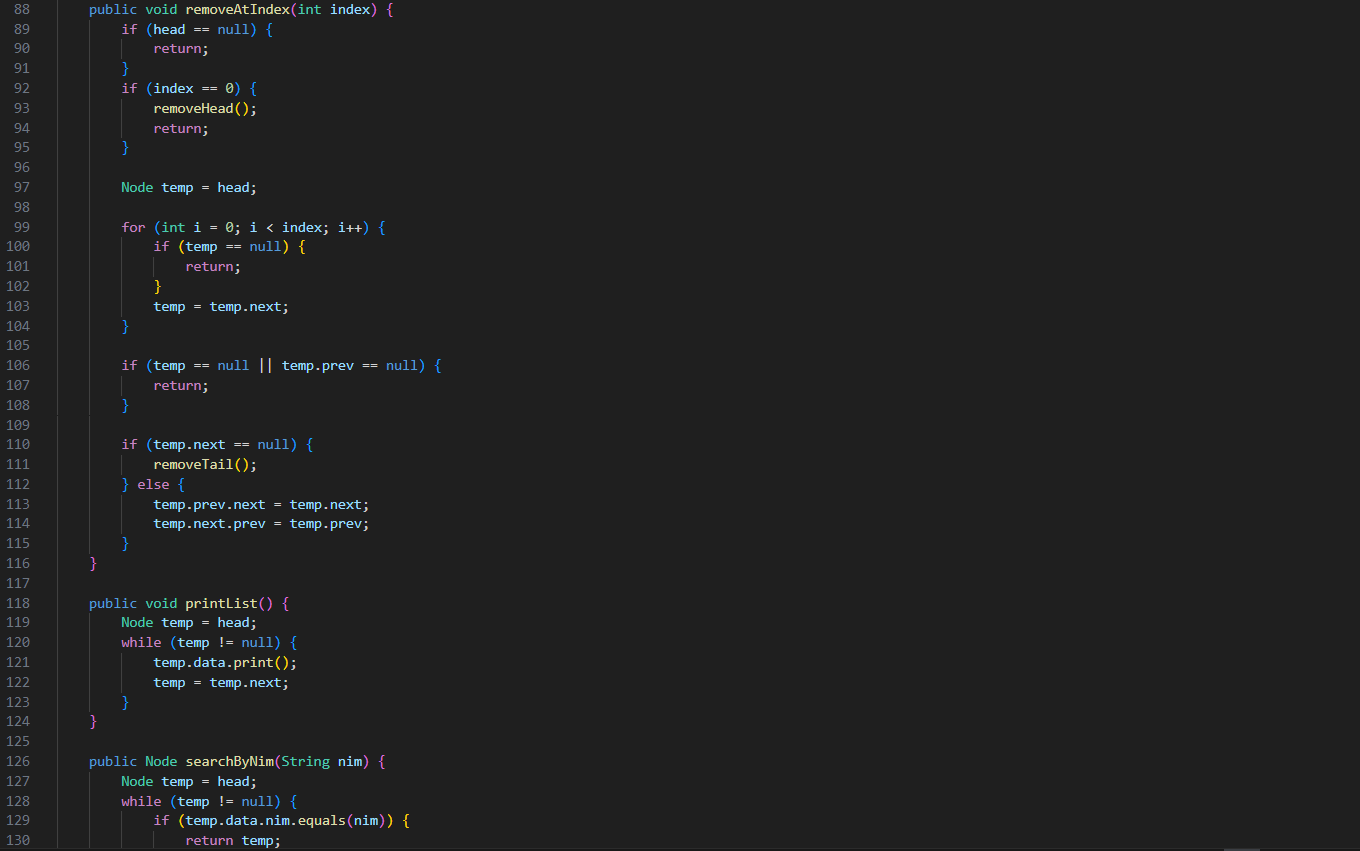
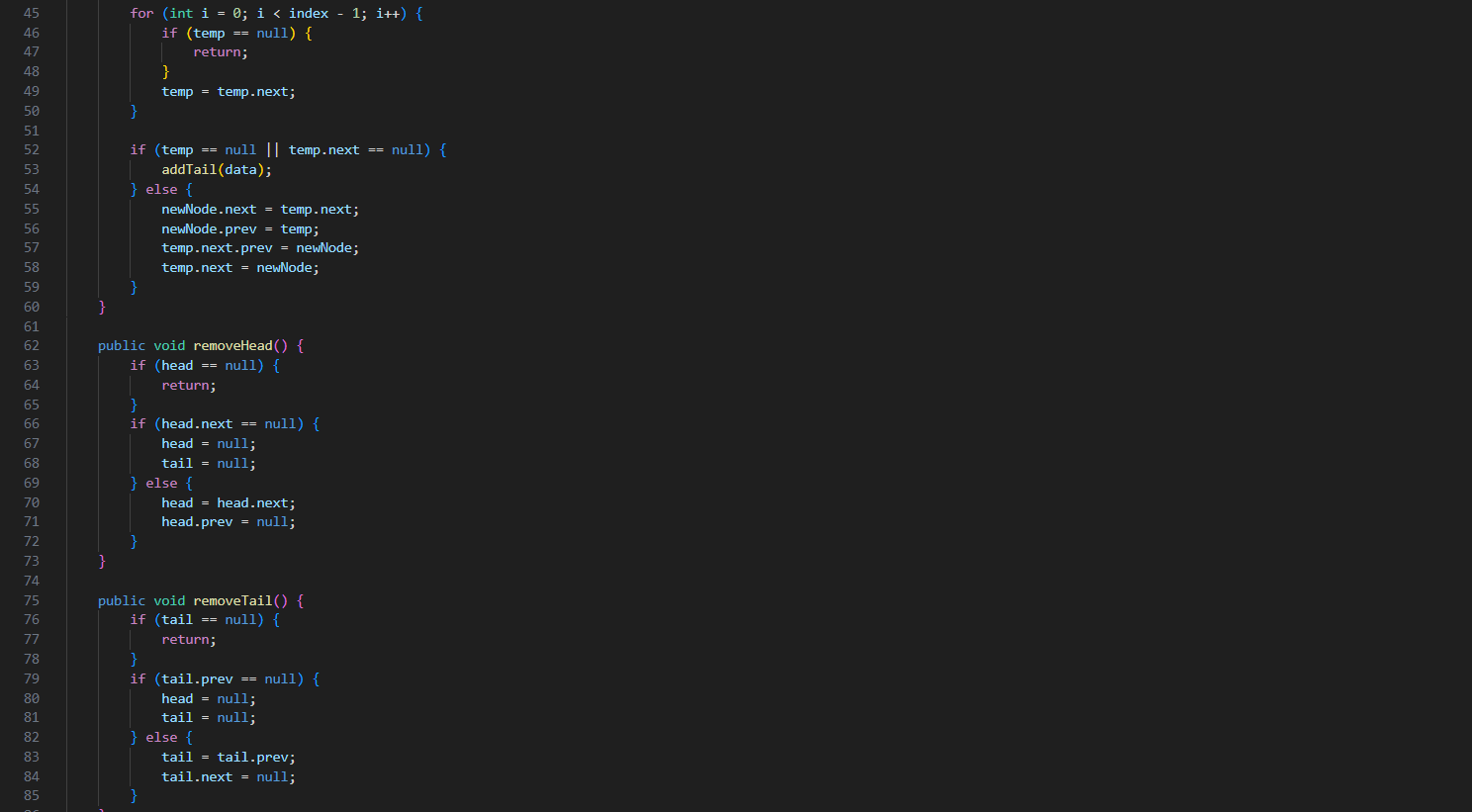
**Searching data**

****

**Sorting data**

****

****

****