Data Structure and Algorithm Practicum
Final Exam 2nd Semester

Name
Dicha Zelianivan Arkana

NIM
2241720002

Class
1i

Department
Information Technology

Study Program
D4 Informatics Engineering

1 Questions

a) `toArray():` method in akan menghasilkan sebuah variable array,
dimana datanya berasal dari data di dalam objek linked list yang sudah ada.

```java
int[] toArray() {
    // prepare the array
    int[] result = new int[this.size()];

    // iterate through the linked list
    Node2P tmp = head;
    for (int i = 0; i < result.length; i++) {
        // place each of the node inside the array and then advance to the next node
        result[i] = tmp.data;
        tmp = tmp.next;
    }

    // return the filled array
    return result;
}
```

b) `sublist(int start, int end):` method ini digunakan untuk mengembalikan list baru yang mengambil sebagian dari data yang sudah ada di list dari posisi `start` sampai posisi `end`

```java
DLL sublist(int start, int end) {
    // prepare the linked sub-list container
    DLL subLinkedList = new DLL();

    // keep track of the current index
    int i = 0;
    // walk through the linked list
    Node2P tmp = head;
    while (tmp != null) {
        // after reaching certain index, add them to the sub-list
        // we don't use the existing `get(int index)` method since it will
        // always iterate from the start everytime we want to get an item
        // (which would make the time complexity to be O(n^2)) while using this
        // method we'll only have to do a single pass (which will make it O(n))
        // since the list is within a range
        if (i >= start && i <= end) {
            subLinkedList.addLast(tmp.data);
        }
        // increment the counter and advance to the next node
        i++;
        tmp = tmp.next;
    }

    // return the filled sub linked list
    return subLinkedList;
}
```

c) `addAll(DLL list)`: method ini digunakan untuk menambahkan data yang ada di `list` ke dalam list yang sudah ada

```
void addAll(DLL list) {
    // iterate through every node in the list in the parameter and
    // then add it to the list of this class
    Node2P tmp = list.head;
    while (tmp != null) {
        // add each item from the passed list to the last of the existing list
        this.addLast(tmp.data);
        tmp = tmp.next;
    }
}
```

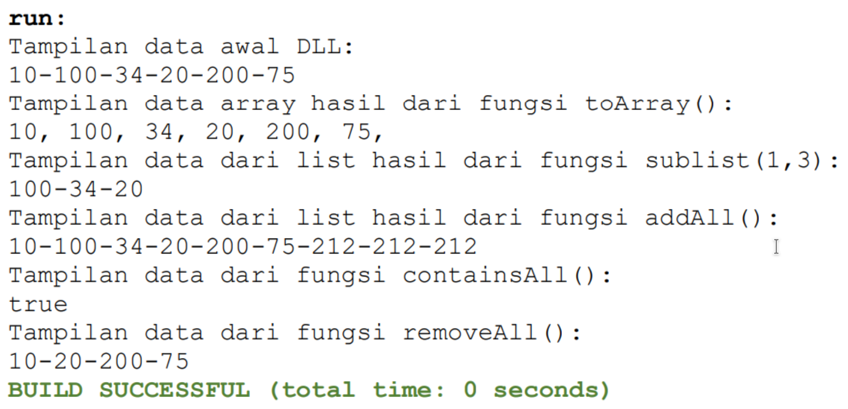
d) `containsAll(Dll list)`: method ini akan mengecek apakah semua data yang ada di dalam `list`, ada di dalam list yang sudah ada

```
boolean containsAll(DLL list) {
    // walk through every nodes
    Node2P tmp = list.head;
    while (tmp != null) {
        // create a flag that marks if the item of the list contained in the class' list
        boolean contains = false;

        // walk through the list passed from the parameter
        Node2P innerTmp = head;
        while (innerTmp != null) {
            // check if the data is the same in the list, if it does then
            // mark the flag as true
            if (innerTmp.data == tmp.data) {
                contains = true;
            }
            innerTmp = innerTmp.next;
        }

        // if at the end of the loop the flag remains false, it means that
        // there is no item from the passed list in the class' list, so we'll
        // just return false here and stop the loop, no need to check the rest
        if (!contains) return false;
        tmp = tmp.next;
    }

    // returns true when the above loop passes
    return true;
}
```

e) `removeAll(DLL list)`: method ini akan menghapus data dari dalam list yang sudah ada berdasarkan nilai yang ada di dalam `list`

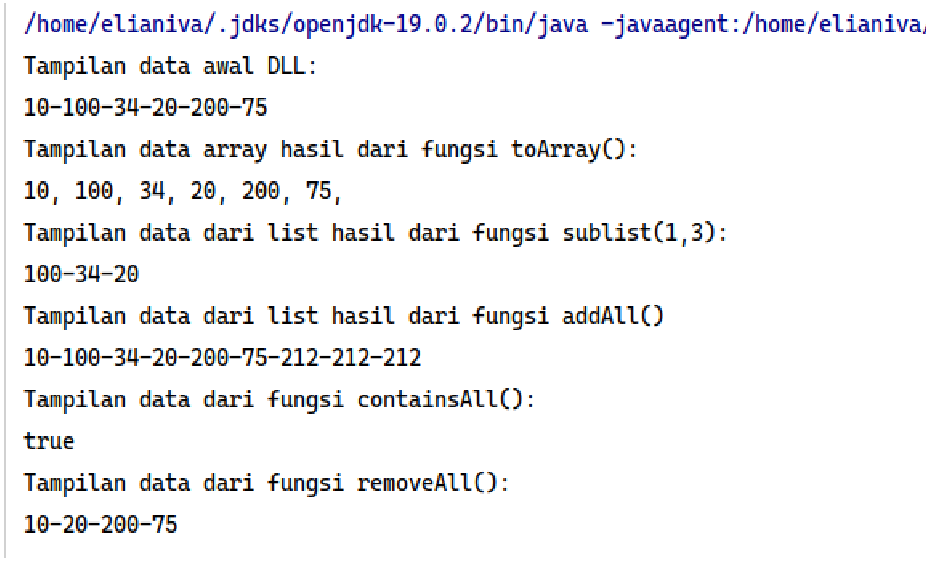
```
void removeAll(DLL list) {
     // iterate through the passed list
     Node2P tmp = list.head;
     while (tmp != null) {
         // remove the item from the list
         // this would imply O(n^2) since we need to iterate through the inner list
         // to find the data that we want to delete based on the list that is passed
         this.deleteByData(tmp.data);
         tmp = tmp.next;
     }
 }
```

2 Output



```
run:
Tampilan data awal DLL:
10-100-34-20-200-75
Tampilan data array hasil dari fungsi toArray():
10, 100, 34, 20, 200, 75,
Tampilan data dari list hasil dari fungsi sublist(1,3):
100-34-20
Tampilan data dari list hasil dari fungsi addAll():
10-100-34-20-200-75-212-212-212
Tampilan data dari fungsi containsAll():
true
Tampilan data dari fungsi removeAll():
10-20-200-75
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 1: The expected output of the program



```
/home/elianiva/.jdks/openjdk-19.0.2/bin/java -javaagent:/home/elianiva,
Tampilan data awal DLL:
10-100-34-20-200-75
Tampilan data array hasil dari fungsi toArray():
10, 100, 34, 20, 200, 75,
Tampilan data dari list hasil dari fungsi sublist(1,3):
100-34-20
Tampilan data dari list hasil dari fungsi addAll()
10-100-34-20-200-75-212-212-212
Tampilan data dari fungsi containsAll():
true
Tampilan data dari fungsi removeAll():
10-20-200-75
```

Figure 2: The actual output of the program