June 2nd, 2020

Exercises for Programming Practice 2

Today's topic is Lisked Lists.

In Exercise 6 to 8, the classes "Node.java", "StackTest.java" and "QueueTest.java" are used. These can be obtained from Week 5 of Resources page, "Programming Practice 2" course, manaba+R.

> Note:
> ✓ Do not create "module-info.java", when you create a Java Project.
> ✓ Do not set Package name in the window "New Java Class".
> ✓ Do not use the title of exercise for "contents of the program".
> Think about "contents of the program" yourself.

The deadline for submitting the programs is 18:00 on June 4th, 2020.

Node class has only two variables: String "element" and Node "next" that are initialized in the constructor. These values can be referenced by two getter methods. One setter method is prepared to set "next".

```
public class Node {
    private String element;
    private Node next;

    // constructor of Node
    // e: element, n: next node (null if no next node)
    public Node(String e, Node n) {
        element = e;
        next = n;
    }

    // return element
    public String getElement() {
        return element;
    }

    // return next node
    Node getNext(){
        return next;
    }

    // set node to next
    public void setNext(Node n) {
        next = n;
    }
}
```

Fig. 1 Node class.

Exercise 6 (file name “Exercise6.java”)

Create method “public void showList(Node node)” to display elements of Linked List. The node given as an argument is the head of the Linked List. Display all element from head to tail with a space separator.

Create method showList(Node *node*) as follows:

> if *node* is null, output a newline.
> Otherwise, display the element of *node* and one space,
> and call showList(*node*.getNext()).

The outline of “Exercise6.java” is shown in Fig. 2.

The String array data is converted to a Linked List starting from “head” in the constructor. A Linked List is displayed using “showList” method to show how a Linked List is created as shown in Fig. 3.

```
/* comments */
public class Exercise6 {
    String[] data = {"AA", "BB", "CC", "DD"};
    Node head;
    Node node;

    public static void main(String[] args) {
        new Exercise6();
    }

    public Exercise6() {
        node = new Node(data[data.length - 1], null);
        showList(node);
        for(int i = data.length - 2; i >= 1; i--) {
            node = new Node(data[i], node);
            showList(node);
        }
        head = new Node(data[0], node);
        showList(head);
    }

    public void showList(Node node) {
        // Complete this part
    }
}
```

Fig. 2 The outline of the program “Exercise15.java”

```
DD
CC DD
BB CC DD
AA BB CC DD
```

Fig. 3 The output of “Exercise15.java”.

Exercise 7 (file name “LinkedStack.java”)

Create method “public void push(String e)” and “public String pop()”, and complete Java program “LinkedStack.java” to implement a stack with a singly linked list using class Node.

The outline of “LinkedStack.java” is shown in Fig. 4. Method “public void showList(Node node)” is the same method in “Exercise6.java” and is called from method “public void showList()” to display all element from head to tail with a space separator.

Use “StackTest.java” to verify that “LinkedStack.java” works correctly. The output of “StackTest.java” should be as shown in Fig. 5.

Submit “LinkedStack.java” to Exercise 7 of Assignments.

```
/* comments */
public class LinkedStack {
    Node head = null;

    // to display a singly linked list
    public void showList() {
        showList(head);
    }

    public void showList(Node node) {
        // Complete this part
    }

    public void push(String e) {
        // Complete this part
    }

    public String pop() {
        // Complete this part
    }
}
```

Fig. 4 The outline of the program “LinkedStack.java”

```
ghi def abe
The poped element is ghi
def abe
jkl def abe
The poped element is jkl
def abe
The poped element is def
abe
The poped element is abe

The poped element is null
```

Fig. 5 The output of “StackTest.java”.

Exercise 8 (file name “LinkedQueue.java”)

Create method “public String dequeue()” and “public void enqueue(String e)”, and complete Java program “LinkedQueue.java” to implement a queue with a singly linked list using class Node.

The outline of “LinkedQueue.java” is shown in Fig. 6. Method “public void showList(Node node)” is the same method in “Exercise6.java” and is called from method “public void showList()” to display all element from head to tail with a space separator.

Use “QueueTest.java” to verify that “LinkedQueue.java” works correctly. The output of “QueueTest.java” should be as shown in Fig. 7.

Submit “LinkedQueue.java” to Exercise 8 of Assignments.

```
/* comments */
public class LinkedQueue {
    Node head = null;
    Node tail = null;
    int size = 0;

    // to display a singly linked list
    public void showList() {
        showList(head);
    }

    public void showList(Node node) {
        // Complete this part
    }

    public String dequeue() {
        // Complete this part
    }

    public void enqueue(String e) {
        // Complete this part
    }
}
```

Fig. 6 The outline of the program “LinkedQueue.java”

```
abe def ghi
The dequeued element is abe
def ghi
def ghi jkl
The dequeued element is def
ghi jkl
The dequeued element is ghi
jkl
The dequeued element is jkl

The dequeued element is null
```

Fig. 7 The output of “QueueTest.java”.