Activity 7.1

Ramzy Izza Wardhana - 21/472698/PA/20322

Linked List

Node.java

```java
package linklist;
public class Node {
    int data;
    Node next;

    //constructor for initialization
    Node(int data){
        this.data = data;
    }

    //print data
    public void displayLink(){
        System.out.print("{" + this.data + "} ");
    }
}
```

LinkListInit.java

```java
package linklist;
public class LinkListInit {
    Node first;

    //Constructor
    LinkListInit(){
        first = null;
    }

    //check whether list is empty or not
    public boolean isEmpty() {
        return (first == null);
    }

    //insert data from the front of the list
    public void insertFirst(int data){
        Node newNode = new Node(data);
        newNode.next = first;
        first = newNode;
    }
```

```
    //insert data from the end of the list
    public void insertLast(int data){
        if(first == null)
            insertFirst(data);
        else {
            Node temp = first;
            while(temp.next != null){
                temp = temp.next;
            }
            temp.next = new Node(data);
        }
    }

    //delete the first data
    public void deleteFirst(){
        Node temp = first;
        first = first.next;
    }

    //delete the last data
    public void deleteLast(){
        Node temp = first;
        while(temp.next.next != null){
            temp = temp.next;
        }
        temp.next = null;
    }

    //print the list
    public void displayList() {
        System.out.print("List (first --> last): ");
        Node current = first;
        while(current != null){
            current.displayLink();
            current = current.next;
        }
        System.out.print("");
    }
}
```

TestListInit.java

```
package linklist;
import java.util.Scanner;
```

```java
public class TestLinkList {
    public static void main(String[] args) {
        LinkListInit thelist1 = new LinkListInit();
        LinkListInit thelist2 = new LinkListInit();

        //defining the size of the list
        Scanner in = new Scanner (System.in);
        int nodeNum1;
        int nodeNum2;
        int tempNum;

        System.out.print("First list size: ");
        nodeNum1 = in.nextInt();

        for(int i = 0; i < nodeNum1; i++){
            System.out.print("Insert Number: ");
            tempNum = in.nextInt();
            thelist1.insertLast(tempNum);
        }
        thelist1.displayList();

        System.out.print("\nSecond list size: ");
        nodeNum2 = in.nextInt();

        for(int i = 0; i < nodeNum2; i++){
            System.out.print("Insert Number: ");
            tempNum = in.nextInt();
            thelist2.insertFirst(tempNum);
        }
        thelist2.displayList();

        //deleting elements of the list
        System.out.println("\nDeleting the first node of the first list");
        thelist1.deleteFirst();
        thelist1.displayList();

        System.out.println("\n\nDeleting the last node of the second list");
        thelist2.deleteLast();
        thelist2.displayList();
    }
}
```

Output

```
First list size: 3
Insert Number: 32
Insert Number: 65
Insert Number: 76
List (first --> last): {32} {65} {76}
Second list size: 4
Insert Number: 54
Insert Number: 9
Insert Number: 78
Insert Number: 4
List (first --> last): {4} {78} {9} {54}
Deleting the first node of the first list
List (first --> last): {65} {76}

Deleting the last node of the second list
List (first --> last): {4} {78} {9}
PS C:\Users\themi\Downloads\java-prak-asd\sixth-meet\activity>
```

Stack

Stack.java

```java
package stack;
import java.io.IOException;
import java.util.Scanner;
import java.util.Arrays;

class StackInit{
    private final int maxSize;
    private int[] stackArray;
    private int top;

    public StackInit(int s){
        maxSize = s;
        stackArray = new int[maxSize];
        top = -1;
    }

    public void push(int j){
        stackArray[++top] = j;
    }

    public double pop(){
        return stackArray[top--];
    }
```

```
    public boolean isEmpty(){
        return(top == -1);
    }

    public void printStack(){
        System.out.println(Arrays.toString(stackArray));
    }
}

public class Stack {
    public static void main(String[] args) throws IOException{
        int stackSize;
        int stackNum;
        Scanner in = new Scanner(System.in);

        System.out.print("How many integer? ");
        stackSize = in.nextInt();

        StackInit theStack = new StackInit(stackSize);

        for(int i = 0; i < stackSize; i++){
            System.out.print("Enter Number: ");
            stackNum = in.nextInt();
            theStack.push(stackNum);
        }
        theStack.printStack();

        while(!theStack.isEmpty()){
            double value = theStack.pop();
            System.out.print(value);
            System.out.print(" ");
        }

        System.out.println("");
    }
}
```

Output

```
How many integer? 4
Enter Number: 1
Enter Number: 2
Enter Number: 6
Enter Number: 7
[1, 2, 6, 7]
7.0 6.0 2.0 1.0
PS C:\Users\themi\Downloads\java-prak-asd\sixth-meet\activity>
```

Queue

Queue.java

```java
package queue;
import java.io.IOException;
import java.util.Scanner;
import java.util.Arrays;

class QueueInit{
    private int maxSize;
    private int[] queueArray;
    private int front;
    private int rear;
    private int nItems;

    public QueueInit(int s){
        maxSize = s;
        queueArray = new int[maxSize];
        front = 0;
        rear = -1;
        nItems = 0;
    }

    public void enqueue(int j){
        if (rear == maxSize - 1)
            rear = -1;
        queueArray[++rear] = j;
        nItems++;
    }

    public int dequeue(){
        int temp = queueArray[front++];
        if(front == maxSize)
            front = 0;
        nItems--;
        return temp;
    }
```

```
    public boolean isEmpty(){
        return(nItems == 0);
    }

    public boolean isFull(){
        return (nItems == maxSize);
    }

    public void printQueue(){
        System.out.println(Arrays.toString(queueArray));
    }
}

public class Queue {
    public static void main(String[] args) throws IOException {
        int queueSize;
        int numTemp;
        int numChoice = 0;

        Scanner in = new Scanner(System.in);
        System.out.print("Enter queue size: ");
        queueSize = in.nextInt();

        QueueInit theQueue = new QueueInit(queueSize);

        while(numChoice != 3){
            System.out.println("\n 1: Enqueue \t 2: Dequeue \t 3: End");
            System.out.print("Enter command: ");
            numChoice = in.nextInt();
            if(numChoice == 1) {
                if(theQueue.isFull())
                    System.out.println("Queue is full");
                else{
                    System.out.print("Enter number: ");
                    numTemp = in.nextInt();
                    theQueue.enqueue(numTemp);
                }
            }
            else if(numChoice == 2){
                if(theQueue.isEmpty())
                    System.out.println("Queue is Empty");
                else{
                    numTemp = theQueue.dequeue();
                    System.out.println("Dequeue number: " + numTemp);
```

```
                }
            }
            else if(numChoice != 3){
                System.out.println("Wrong Command!");
            }
        }
    }
}
```

Output

```
Enter queue size: 5

 1: Enqueue      2: Dequeue      3: End
Enter command: 1
Enter number: 23

 1: Enqueue      2: Dequeue      3: End
Enter command: 1
Enter number: 45

 1: Enqueue      2: Dequeue      3: End
Enter command: 2
Dequeue number: 23

 1: Enqueue      2: Dequeue      3: End
Enter command: 2
Dequeue number: 45

 1: Enqueue      2: Dequeue      3: End
Enter command: 3
PS C:\Users\themi\Downloads\java-prak-asd\sixth-meet\activity>
```