

## Лабораторна робота №3

Виконав:  
ст. групи КН-107  
Щербань О. І.  
Прийняв:  
Старший викладач  
кафедри СШ  
Гасько Р.Т.

## Завдання 1. Зв'язний список

Код програми:

1.1) class Node

```
public class Node {
    private Node next;
    private Integer data;

    public Node() {
    }

    public Node getNext() {
        return next;
    }
    public void setNext(Node next) {
        this.next = next;
    }
    public Integer getData() {
        return data;
    }
    public void setData(Integer data) {
        this.data = data;
    }
}
```

1.2) class LinkedList

```
public class LinkedList {
    private Node tail;
    private Node head;
    private int size = 0;

    public LinkedList() {
    }

    public void add(Integer data) {
        Node newNode = new Node();
        newNode.setData(data);

        if (size == 0)
            head = newNode;
        else
            tail.setNext(newNode);

        tail = newNode;
        size++;
    }

    public Integer get(int index) {
        return findNodeByIndex(index).getData();
    }

    public boolean delete(int index) {
        if (findNodeByIndex(index) != null) {

            if (index != 0)
                findNodeByIndex(index - 1).setNext(findNodeByIndex(index+1));
            else
                head = head.getNext();

            size--;
            return true;
        }
    }
}
```

```

        return false;
    }

    public int size() {
        return size;
    }

    private Node findNodeByIndex(int index) {
        if (index < size && index >= 0) {
            Node curNode = head;
            int curIndex = 0;
            while (curIndex < index){
                if (curNode.getNext() != null) {
                    curNode = curNode.getNext();
                    curIndex++;
                }
            }
            return curNode;
        }
        else
            return null;
    }

    public String toString() {
        StringBuilder resultString = new StringBuilder("");

        for (int i = 0; i < size; i++) {
            resultString.append(get(i));

            if (i < size-1) resultString.append(", ");
        }
        resultString.append("]");
        return resultString.toString();
    }
}

```

Відповідь на Prometheus:

**ЗВЕРНІТЬ УВАГУ:** елементи списку повинні мати номери починаючи з нуля!

```

1 package com.tasks3.linkedlist;
2
3 public class LinkedList {
4     private Node tail;
5     private Node head;
6     private int size = 0;
7
8     public LinkedList() {
9
10    }
11
12    public void add(Integer data) {
13        Node newNode = new Node();
14        newNode.setData(data);
15
16        if (size == 0)

```

Правильно

Результати тесту

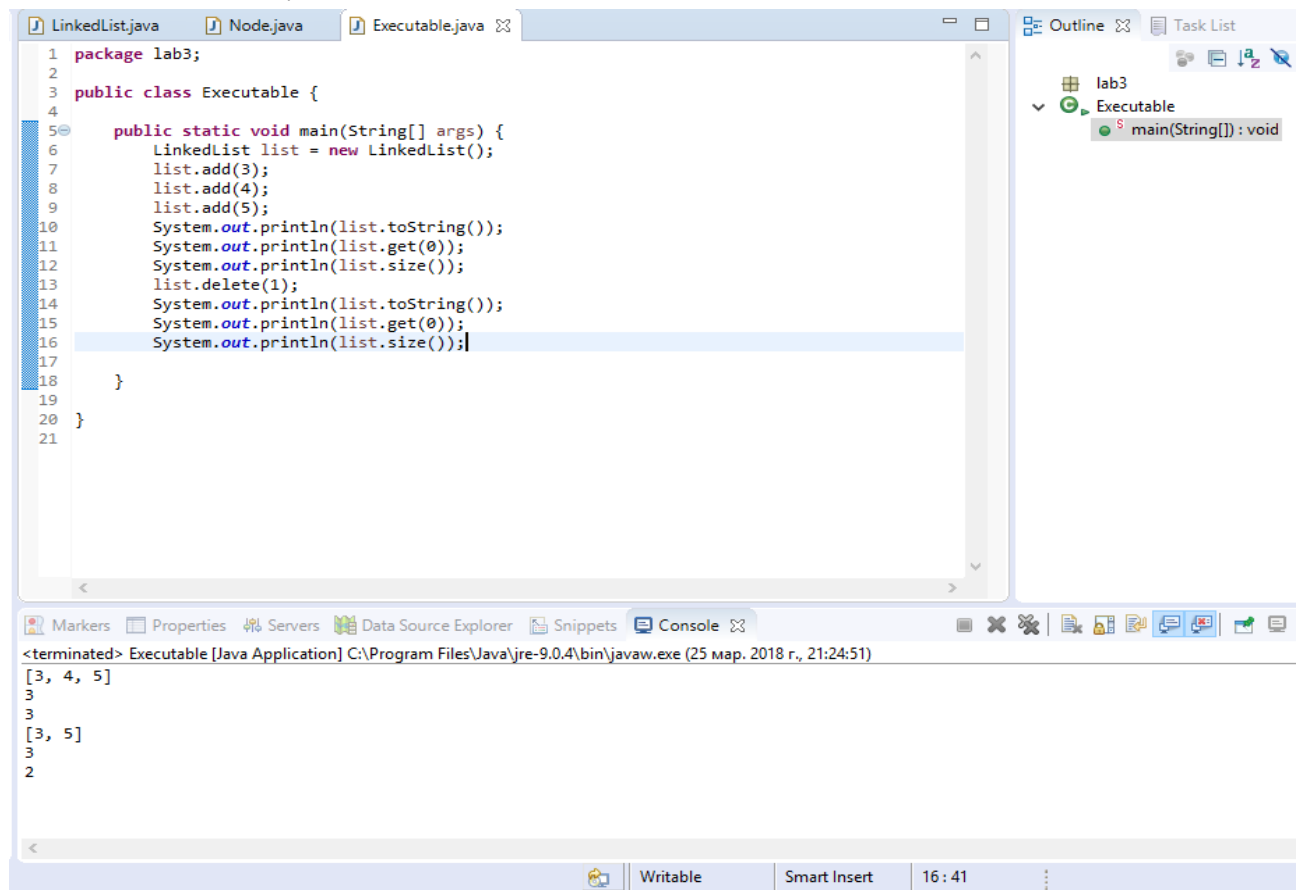
See full output	
ВІРНО	See full output

ПЕРЕВІРКА

ЗБЕРЕГТИ

ПОКАЗАТИ ВІДПОВІДЬ

## Виконання в IDE:



## Завдання 2. Колода карт

Код програми:

### 2.1) class Card

```
public class Card {
    private Rank rank;
    private Suit suit;

    public Card(Rank rank, Suit suit) {
        this.rank = rank;
        this.suit = suit;
    }

    public Rank getRank() {
        return rank;
    }

    public void setRank(Rank rank) {
        this.rank = rank;
    }

    public Suit getSuit() {
        return suit;
    }

    public void setSuit(Suit suit) {
        this.suit = suit;
    }
}
```

## 2.2) class Rank

```
public class Rank {
    public static final Rank ACE = new Rank("Ace");
    public static final Rank KING = new Rank("King");
    public static final Rank QUEEN = new Rank("Queen");
    public static final Rank JACK = new Rank("Jack");
    public static final Rank TEN = new Rank("10");
    public static final Rank NINE = new Rank("9");
    public static final Rank EIGHT = new Rank("8");
    public static final Rank SEVEN = new Rank("7");
    public static final Rank SIX = new Rank("6");

    public static Rank[] values = { ACE, KING, QUEEN, JACK, TEN, NINE, EIGHT, SEVEN, SIX };

    private String name;

    Rank(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }
}
```

## 2.3) class Suit

```
public class Suit {
    public static final Suit HEARTS = new Suit("HEARTS");
    public static final Suit DIAMONDS = new Suit("DIAMONDS");
    public static final Suit CLUBS = new Suit("CLUBS");
    public static final Suit SPADES = new Suit("SPADES");

    public static Suit[] values = { HEARTS, DIAMONDS, CLUBS, SPADES };

    private String name;

    Suit(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }
}
```

## 2.4) class Deck

```
public class Deck {
    Card[] deck;
    int index;
    public Deck(){
        index = 35;
        int i = -1;
        this.deck = new Card[Suit.values.length * Rank.values.length];
        for (Suit suit: Suit.values) {
            for (Rank rank: Rank.values) {
                deck[++i] = new Card(rank,suit);
            }
        }
    }
    public void shuffle() {
        Card tmpCard;
        int q = (int) (Math.random()*300+700);
        for (int n=0; n<q; n++){
            int l = (int) (Math.random()*32);
```

```

        int m = (int) (Math.random()*(36-1)+1);
        tmpCard = this.deck[m];
        this.deck[m] = this.deck[m-1];
        this.deck[m-1] = tmpCard;
    }
}

public void order() {
    index = 35;
    int i = -1;
    this.deck = new Card[Suit.values.length * Rank.values.length];
    for (Suit suit: Suit.values) {
        for (Rank rank: Rank.values) {
            deck[++i] = new Card(rank,suit);
        }
    }
}

public boolean hasNext() {
    return index > -1;
}

public Card drawOne() {
    if (index >= 0)
        return this.deck[index--];
    else return null;
}

public void deckPrn(){
    System.out.println("---Cards in deck-"+(index+1)+"---");
    for (int j=0; j<index+1; j++){
        System.out.print("|");
        System.out.print(this.deck[j].getSuit().getName());
        System.out.print(" ");
        System.out.println(this.deck[j].getRank().getName());
    }
}

public void cardPrn(Card crd){
    if (crd != null){
        System.out.print(crd.getSuit().getName());
        System.out.print(" ");
        System.out.println(crd.getRank().getName());
    }
    else System.out.println("Card is NULL");
}

public static void main(String[] args) {

    Deck dk1 = new Deck();

    dk1.shuffle();
    dk1.deckPrn();
}
}

```

## Відповідь на Prometheus:

```
    return name;
}
}
```

```
1 package com.tasks3.carddeck;
2
3 public class Deck {
4     Card[] deck;
5     int index;
6
7     // Constructor
8     public Deck(){
9         index = 35;
10        int i = -1;
11        this.deck = new Card[Suit.values.length * Rank.values.length];
12        for (Suit suit: Suit.values) {
13            for (Rank rank: Rank.values) {
14                deck[++i] = new Card(rank,suit);
15            }
16        }
17    }
18 }
```

Правильно

Результати тесту

See full output

ВІРНО

See full output

ПЕРЕВІРКА

ЗБЕРЕГТИ

ПОКАЗАТИ ВІДПОВІДЬ

## Виконання в IDE:

The screenshot shows an IDE with the following components:

- Editor:** Displays the `Deck.java` file. The code includes a package declaration, class definition, constructor, and methods for shuffling and ordering the deck. The `shuffle()` method uses a Fisher-Yates shuffle algorithm.
- Outline:** Shows the project structure with `lab3` and `Deck` class.
- Console:** Displays the output of the program, showing the cards in the deck after shuffling: HEARTS 6, DIAMONDS 8, CLUBS 9, DIAMONDS 6, SPADES 7, DIAMONDS 7, SPADES 9, SPADES Ace.

```
1 package lab3;
2
3 public class Deck {
4     Card[] deck;
5     int index;
6     public Deck(){
7         index = 35;
8         int i = -1;
9         this.deck = new Card[Suit.values.length * Rank.values.length];
10        for (Suit suit: Suit.values) {
11            for (Rank rank: Rank.values) {
12                deck[++i] = new Card(rank,suit);
13            }
14        }
15    }
16    public void shuffle() {
17        Card tmpCard;
18        int q = (int) (Math.random()*300+700);
19        for (int n=0; n<q; n++){
20            int l = (int) (Math.random()*32);
21            int m = (int) (Math.random()*(36-1)+1);
22            tmpCard = this.deck[m];
23            this.deck[m] = this.deck[l];
24            this.deck[l] = tmpCard;
25        }
26    }
27    public void order() {
28        index = 35;
29    }
30 }
```

Console Output:

```
<terminated> Deck [Java Application] C:\Program Files\Java\jre-9.0.4\bin\javaw.exe (25 map. 2018 r., 21:31:43)
---Cards in deck-36---
|HEARTS 6
|DIAMONDS 8
|CLUBS 9
|DIAMONDS 6
|SPADES 7
|DIAMONDS 7
|SPADES 9
|SPADES Ace
```

### Завдання 3. Числа Фібоначі

Код програми:

```
public class Fibonacci
{
    public long getNumber(int position){

        if (position == 1)
        {
            return 1;
        }
        else if (position > 0) {
            long c = 0;
            long a = 1;
            long b = 0;

            for (int i = 1; i < position; i++) {
                c = a + b;
                b = a;
                a = c;
            }

            return c;
        }
        else
        {
            return -1;
        }
    }
}
```

Відповідь на Prometheus:

```
//якщо число не можливо вираховати повернути -1
public long getNumber(int position){
}
}
```

```
1 package com.tasks3.fibonacci;
2
3 public class Fibonacci
4 {
5     public long getNumber(int position){
6
7         if (position == 1)
8         {
9             return 1;
10        }
11        else if (position > 0) {
12            long x = 0;
13            long prevX = 1;
14            long prev2X = 0;
15
16            for (int i = 1; i < position; i++) {
```

Правильно

#### Результати тесту

See full output	
ВІРНО	
See full output	

ПЕРЕВІРКА

ЗБЕРЕГТИ

ПОКАЗАТИ ВІДПОВІДЬ



## Виконання в IDE:

