Міністерство освіти і науки України

Національний університет «Львівська політехніка»



**Лабораторна робота №1**

на тему:

«Короткий огляд мови Java»

з курсу:

«Об’єктно-орієнтоване програмування»

Виконала:

ст. гр. КН-110

Трояновська Олеся

Прийняв:

Гасько Р.Т

Львів – 2018р.

**Тема:** виконати завдання 3 тижня курсу а саме cтворіть клас що описує зв'язний список. Створіть класи для зберігання колоди з 36 карт.

**Мета:** освоїти основи синтаксису Java.

Приклади програм

1)

|  |
| --- |
| public class LinkedList { |
|  | private Node head; |
|  | private Node tail; |
|  | /\* Конструктор без аргументів \*/ |
|  | public LinkedList() {} |
|  | /\* Додати елемент в кінець списку \*/ |
|  | public void add(Integer data) { |
|  |  |
|  | Node node = new Node(); |
|  | // новий элемент |
|  | node.setData(data); |
|  | if(tail == null) |
|  | { |
|  | head = node; |
|  | tail = node; |
|  | } |
|  | else { |
|  | tail.setNext(node); |
|  | tail = node; |
|  | } |
|  |  |
|  | } |
|  | /\* Отримати елемент по індексу, повертає null якщо такий елемент недоступний \*/ |
|  | public Integer get(int index) { |
|  | int count=0; |
|  | Node t=head; |
|  | while (t!=null && index<size()) |
|  | { |
|  | if(index==count) |
|  | return t.getData(); |
|  | else{ count++; |
|  | t=t.getNext();} |
|  |  |
|  | } |
|  | return null; |
|  |  |
|  | } |
|  | /\* Вилучення елементу за індексом, повертає true у разі успіху або false в іншому випадку \*/ |
|  | public boolean delete(int index) { |
|  | int count=0; |
|  | Node t=head; |
|  | if (head == null) |
|  | return false; |
|  | while (t!=null && index<size()) { |
|  | if (index == count) { |
|  |  |
|  |  |
|  | // Store head node |
|  | Node temp = head; |
|  |  |
|  | if (index == 0) |
|  | { |
|  | head = temp.getNext(); // Change head |
|  | return true; |
|  | } |
|  |  |
|  | // Find previous node of the node to be deleted |
|  | for (int i = 0; temp != null && i < index - 1; i++) |
|  | temp = temp.getNext(); |
|  |  |
|  | // If position is more than number of ndoes |
|  | if (temp == null || temp.getNext() == null) |
|  | return false; |
|  |  |
|  | // Node temp->next is the node to be deleted |
|  | // Store pointer to the next of node to be deleted |
|  | Node next = temp.getNext().getNext(); |
|  |  |
|  | temp.setNext(next); |
|  | return true;// Unlink the deleted node from list |
|  | } else { |
|  | count++; |
|  | t = t.getNext(); |
|  | } |
|  | } |
|  |  |
|  |  |
|  |  |
|  | return false; |
|  | } |
|  | /\*Поверта розмір списку: якщо елементів в списку нема то повертає 0 (нуль)\*/ |
|  | public int size() { |
|  | int counter=0; |
|  | Node t = head; //получаем ссылку на первый элемент |
|  | while (t != null) //пока элемент существуе |
|  | { |
|  | t = t.getNext(); |
|  | counter++;//и переключаемся на следующий |
|  | } |
|  | return counter; |
|  | } |
|  |  |
|  |  |
|  | public static void main(String[] args) { |
|  | // write your code here |
|  | LinkedList t=new LinkedList(); |
|  | t.add(12); |
|  | t.add(13); |
|  | t.add(14); |
|  | t.add(111); |
|  | t.add(15); |
|  | int a=t.size(); |
|  |  |
|  | } |
|  | } |

2) package com.tasks3.carddeck;

import java.util.\*;

public class Deck {

public Card[] deck = new Card[36];

private int size = 0;

public void shuffle() {

Random rand = new Random(System.currentTimeMillis());

for (int i = 0; i < this.deck.length; i++)

{

int randRank = rand.nextInt(Rank.values.length);

int randSuit = rand.nextInt(Suit.values.length);

this.deck[i] = new Card(Rank.values[randRank], Suit.values[randSuit]);

}

this.size = 36;

//Collections.shuffle(deck);

}

public void order() {

for (int suit = 0, place = 0; suit < 4; suit++)

{

for (int rank = 0; rank < 9; rank++, place++)

{

this.deck[place] = new Card(Rank.values[rank], Suit.values[suit]);

}

}

this.size = 36;

}

public boolean hasNext() {

return (this.size > 0);

}

public Card drawOne() {

if (size <= 0)

return null;

Card randCard = this.deck[this.size - 1];

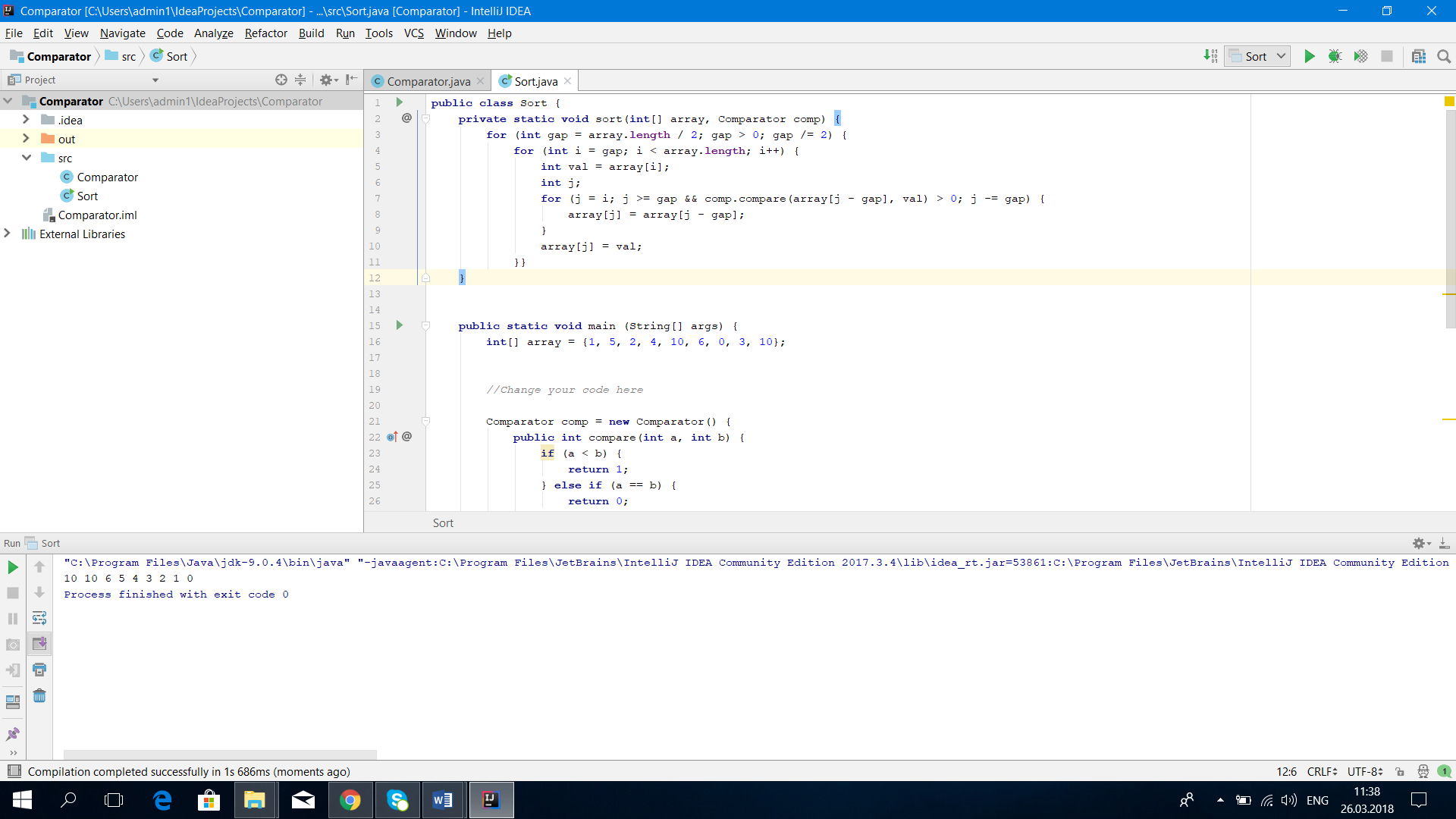
this.deck[this.size - 1] = null;

this.size--;

return randCard;

}

}



3) package com.tasks3.fibonacci;

public class Fibonacci

{

public long getNumber(int position){

//PUT YOUR CODE HERE

if(position==1||position==2)

return 1;

else if(position>2)

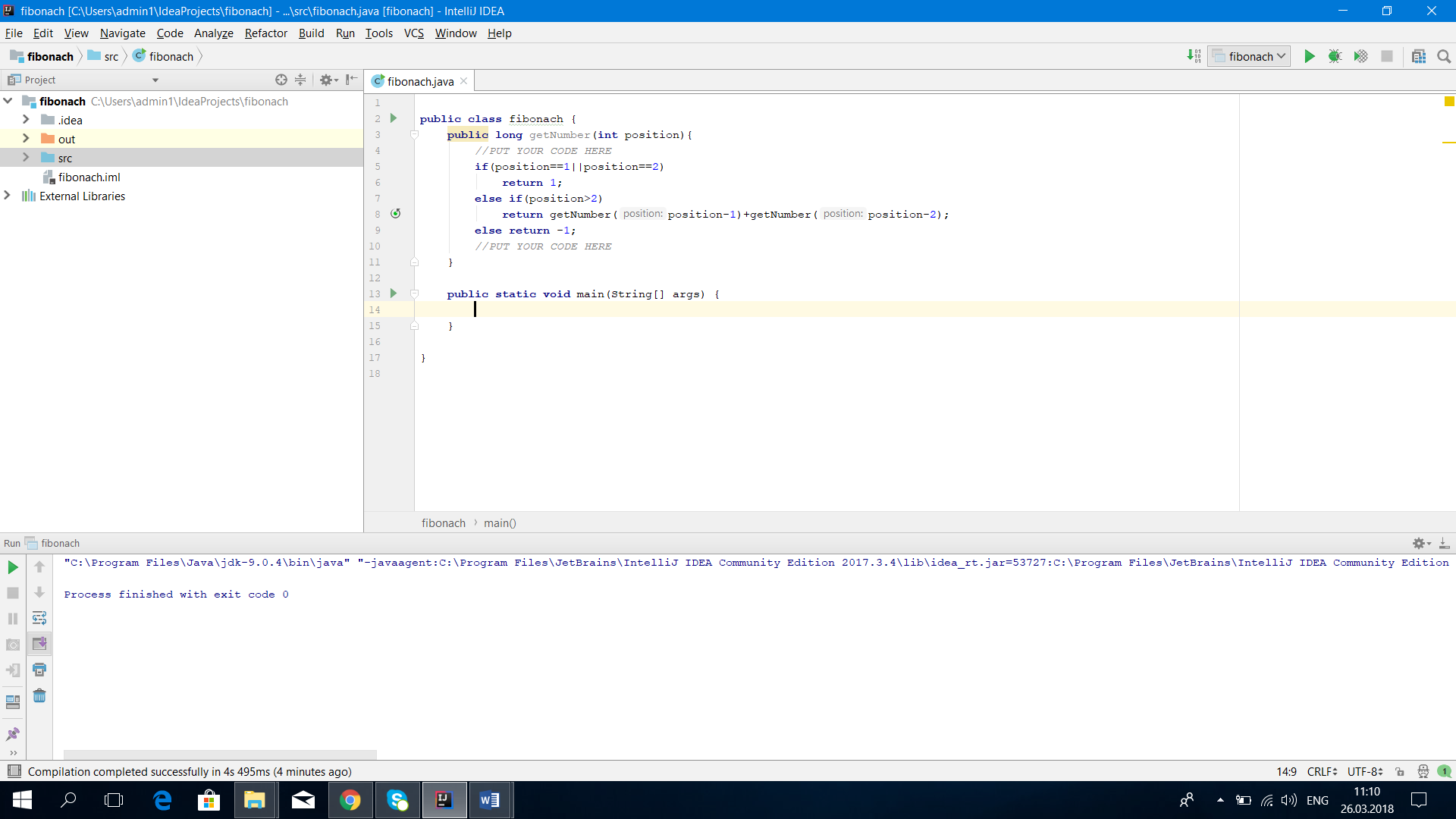
return getNumber(position-1)+getNumber(position-2);

else return -1;

//PUT YOUR CODE HERE

}

}



Висновок: навчилися створювати клас який об’єднує зв’язний список.