# Міністерство Освіти і Науки України

# Національний технічний університет

«Харківський політехнічний інститут»

Кафедра «Програмної інженерії та інформаційних технологій управління»

**Лабораторна робота № 5**

Робота з масивами та рядками Java.

**Виконав:**

Студент групи КН-36А

Кулик В.В.

**Перевірив:** Смолін П.А.

Харків 2017

**Завдання**

**1.1 Індивідуальне завдання**

Спроектувати та реалізувати два класи відповідно до індивідуального завдання. У першому з класів повинен бути описаний масив елементів другого класу. Класи повинні мати конструктори, приватні поля та відкриті методи, зокрема методи доступу (сеттери та геттери).

Слід окремо здійснити тестування кожного з класів, після чого окремими методами першого класу реалізувати основне завдання. У функції main() першого з класів створити необхідний об'єкт та викликати для нього методи, які реалізують основне завдання. Вивести результати у консольне вікно.

Варіант завдання, який слід реалізувати у програмі, визначається залежно від номеру студента у списку груп.



**1.2 Ератосфенове решето**

### Заповнити масив із трьохсот цілих чисел послідовними додатними значеннями. Замінити всі значення, що не є простими числами, деяким від'ємним значенням. Для цього послідовно виключати всі числа - дільники інших чисел. Вивести на екран додатні значення, що залишилися, (прості числа).

### 1.3 Сортування вибором

Проініціалізувати одновимірний масив цілих чисел випадковими значеннями. Здійснити упорядкування масиву методом сортування вибором за таким алгоритмом:

* знаходимо індекс мінімального значення;
* здійснюємо обмін цього значення із значенням першої невідсортованої позиції (обмін не потрібний, якщо мінімальний елемент вже знаходиться на цій позиції);
* сортуємо решту масиву, виключивши з розгляду вже відсортовані елементи.

Результат вивести на екран.

**1.4 Знаходження чисел Фібоначчі**

Реалізувати функцію обчислення чисел Фібоначчі (до 92-го числа включно) з використанням допоміжного масиву (статичного поля). Параметр функції – номер числа Фібоначчі. Пошук чисел Фібоначчі здійснюється за таким правилом:

*F*(1) = *F*(2) = 1; *F*(*n*) = *F*(*n* - 2) + *F*(*n* - 1)

Під час першого виклику функції масив заповнюється до необхідного числа. Під час наступних викликів число або повертається з масиву, або обчислюється з використанням останніх двох чисел, що зберігаються у масиві з подальшим заповненням масиву. Використовувати тип **long** для представлення чисел.

Здійснити тестування функції для різних значень номерів, що вводяться у довільному порядку.

**1.5 Вирівнювання рядку**

Прочитати аргумент командного рядка і додати в нього пробіли так, щоб довжина рядка дорівнювала заданому числу.

**1.6 Абревіатура**

Увести з клавіатури рядок з декількох слів. Скласти рядок з перших літер слів з переведенням цих літер у верхній регістр.

**1.7 Квадратне рівняння**

Спроектувати клас для розв'язання квадратного рівняння. Слід передбачити аналіз усіх можливих комбінацій коефіцієнтів та відповідних результатів (два корені, один корінь для випадку виродження у лінійне рівняння, немає розв'язків, безмежна кількість коренів). Метод безпосереднього знаходження коренів (solve()) повинен повертати кількість коренів (-1 якщо безмежна кількість коренів). Геттери для x1 і x2 повинні повертати корені тільки, якщо було визначено вихідні дані та здійснено пошук коренів.

**Хід роботи**

**1.1 Індивідуальне завдання.**

*Код програми 1.1:*

**TIME**

package com.company;

import java.util.StringTokenizer;

public class Time {

private String name;

private int year;

private Station[] Stationes;

public Time(String name, int year, Station[] Stationes) {

this.name = name;

this.year = year;

this.Stationes = Stationes;

}

public int getyear() {

return year;

}

public void setyear(int year) {

this.year = year;

}

public Station[] getStationes() {

return Stationes;

}

public void setStationes(Station[] Stationes) {

this.Stationes = Stationes;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String maxtime() {

Station Station = Stationes[0];

for (int i = 1; i < Stationes.length; i++) {

if (Station.getpassangers() > Stationes[i].getpassangers()) {

Station = Stationes[i];

}

}

return Station.gettime();

}

public int getcount(Station[] Stationes) {

int sum = 0;

for (int i=0; i < Stationes.length; i++){

sum += Stationes[i].getpassangers();

}

return sum;

}

public int getcountwords(Station[] Stationes) {

int sum = 0;

for (int i=0; i < Stationes.length; i++){

sum += Stationes[i].getpassangers();

}

return sum;

}

public String Findminword(){

Station min = this.Stationes[0];

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

if (this.Stationes[i].getComments().split(" ").length < min.getComments().split(" ").length){

min = this.Stationes[i];

}

}

return min.getComments();

}

public static void main(String[] args) {

Station[] Stationes = {

new Station("10.25", 400, "Загружка метро не максимальная"),

new Station("12.35", 300, "Загрузка ниже среднего"),

new Station("14.30", 200, "Метро в данное время не загружено"),

new Station("16.25", 550, "Загруженность метро после обеда"),

new Station("8.30", 500, "Загруженность метро утром")

};

Time Time = new Time("Холодная гора", 1978, Stationes);

System.out.println("Время с наименьшим количеством пассажиров: " + Time.maxtime());

System.out.println("Суммароное количество пассажиров: " + Time.getcount(Stationes));

System.out.println("Минимальное количество слов в комментарии: " + Time.Findminword());

}

}

**STATION**

package com.company;

import java.util.StringTokenizer;

public class Station {

private String time;

private int passangers;

private String comments;

public Station(String time, int passangers, String comments) {

this.time = time;

this.passangers = passangers;

this.comments = comments;

}

public String getComments() {

return comments;

}

public void setComments(String comments) {

this.comments = comments;

}

public int getpassangers() {

return passangers;

}

public void setpassangers(int passangers) {

this.passangers = passangers;

}

public String gettime() {

return time;

}

public void settime(String time) {

this.time = time;

}

public boolean containsWord(String word) {

StringTokenizer st = new StringTokenizer(comments);

String s;

while (st.hasMoreTokens()) {

s = st.nextToken();

if (s.toUpperCase().equals(word.toUpperCase())) {

return true;

}

}

return false;

}

}

**1.2 Ератосфенове решето**

*Код програми 1.2:*

package lab5;  
  
public class resheto {  
 public static void main(String[] args) {  
 int[] a = new int[300];  
 for (int i = 1; i < a.length; i++) {  
 a[i] = i;  
 }  
 a[1] = -a[1];  
 int m = 2;  
 int j;  
 while (a.length > m) {  
 if (a[m] > 0) {  
 j = m \* 2;  
 while (j < a.length) {  
 if (a[j] > 0)  
 a[j] = -a[j];  
 j += m;  
 }  
 }  
 m += 1;  
 }  
 for (int z = 1; z < a.length; z++) {  
 if (a[z] > 0) {  
 System.*out*.println(a[z]);  
 }  
 }  
 }  
}

**1.3 Сортування вибором**

*Код програми 1.3:*

package lab5;  
  
import java.util.\*;  
  
public class sort {  
 public static void main(String[] args) {  
 int[] a = new int[15];  
 Random rand = new Random();  
 System.*out*.println("Исходный массив:");  
 for (int i = 0; i < a.length; i++) {  
 a[i] = rand.nextInt(20);  
 }  
 System.*out*.println(Arrays.*toString*(a));  
 int min, index;  
 System.*out*.println("Отсортированный массив:");  
 for (int j = 0; j < a.length; j++) {  
 min = a[j];  
 index = j;  
 for (int z = j; z < a.length; z++) {  
 if (min > a[z]) {  
 min = a[z];  
 index = z;  
 }  
 }  
 a[index] = a[j];  
 a[j] = min;  
 }  
 System.*out*.println(Arrays.*toString*(a));  
 }  
}

**1.4 Знаходження чисел Фібоначчі**

*Код програми 1.4:*

package lab5;  
  
public class fibonachi {  
 private static long[] *f* = new long[30];  
 private static int *last* = 1;  
  
 public static long factorial(int n) {  
 *f*[0] = *f*[1] = 1;  
 if (n > *last*) {  
 for (int i = *last* + 1; i <= n; i++) {  
 *f*[i] = *f*[i - 2] + *f*[i - 1];  
 }  
 *last* = n;  
 }  
 return *f*[n];  
 }  
  
 public static void main(String[] args) {  
 System.*out*.println(*factorial*(5));  
 System.*out*.println(*factorial*(1));  
 System.*out*.println(*factorial*(3));  
 System.*out*.println(*factorial*(6));  
 System.*out*.println(*factorial*(20));  
 }  
}

**1.5 Вирівнювання рядку**

*Код програми 1.5:*

package lab5;  
  
import java.util.Scanner;  
  
public class rydok {  
 public static void main(String[] args) {  
 Scanner p = new Scanner(System.*in*);  
 String s = new String(p.nextLine());  
 System.*out*.println(s.length());  
 while (s.length() < 30) {  
 s = s.concat(" ");  
 }  
 System.*out*.println(s);  
 System.*out*.println(s.length());  
 }  
}

**1.6 Абревіатура**

*Код програми 1.6:*

package lab5;  
  
import java.util.Scanner;  
  
public class abrevia {  
 public static void main(String[] args) {  
 Scanner s = new Scanner(System.*in*);  
 String s1 = new String(s.nextLine());  
 String s2 = new String();  
 s2 += s1.charAt(0);  
 for (int i = 1; i < s1.length(); i++) {  
  
 if (s1.charAt(i) == ' ') {  
 s2 += s1.charAt(i+1);  
 }  
 }  
 System.*out*.println(s2.toUpperCase());  
 }  
}

**1.7 Квадратне рівняння**

*Код програми 1.7:*

package lab5;  
  
import java.lang.Math;  
  
public class kvadrat\_urav {  
 private double a, b, c, x1, x2;  
  
 public double getA() {  
 return a;  
 }  
  
 public void setA(double a) {  
 this.a = a;  
 }  
  
 public double getB() {  
 return b;  
 }  
  
 public void setB(double b) {  
 this.b = b;  
 }  
  
 public double getC() {  
 return c;  
 }  
  
 public void setC(double c) {  
 this.c = c;  
 }  
  
 public double getX1() {  
 return x1;  
 }  
  
 public double getX2() {  
 return x2;  
 }  
  
 public int solve() {  
 double d = b \* b - 4 \* a \* c;  
 if (d < 0) {  
 return 0;  
 } else if (d == 0) {  
 if (b == a) {  
 return -1;  
 } else {  
 x1 = -b / 2 \* a;  
 return 1;  
 }  
 } else {  
 x1 = (-b + Math.*sqrt*(d)) / 2 \* a;  
 x2 = (-b - Math.*sqrt*(d)) / 2 \* a;  
 return 2;  
 }  
 }  
  
 public static void f(kvadrat\_urav e) {  
 if (e.solve() == 0) {  
 System.*out*.println("Уравнение корней не имеет");  
 } else if (e.solve() == -1) {  
 System.*out*.println("Уравнение имеет бесконечное множество корней");  
 } else if (e.solve() == 1) {  
 System.*out*.println("Уравнение имеет единственный корень\n" + "x = " + e.getX1());  
 } else {  
 System.*out*.println("Уравнение имеет два корня\n" + "x1 = " + e.getX1() + "\nx2 = " + e.getX2());  
 }  
 }  
  
 public static void main(String[] args) {  
 kvadrat\_urav e = new kvadrat\_urav();  
 e.setA(1);  
 e.setB(-2);  
 e.setC(3);  
 *f*(e);  
 e.setA(1);  
 e.setB(-8);  
 e.setC(12);  
 *f*(e);  
 e.setA(1);  
 e.setB(-6);  
 e.setC(9);  
 *f*(e);  
 e.setA(4);  
 e.setB(4);  
 e.setC(1);  
 *f*(e);  
 }  
}

***Висновок:***

*У даній роботі я за допомогою мови програмування Java та використовуючи попередній досвід я розробив різного роду програми, під час розробки яких я навчився використовувати масиви, рядки та класи в Java.*