Міністерство освіти і науки України

Національний Технічний Університет

«Харківський Політехнічний Інститут»

Кафедра «Стратегічного Управління»

Лабораторна робота № 5

«Робота з масивами та рядками Java. Створення класів»

Перевірила: ас. кафедри СК

Вільхівская О. В.

Виконав:

Харків 2018

Лабораторна робота № 5

Тема: Робота з масивами та рядками Java. Створення класів

Завдання

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 групи.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Перший клас | | Другий клас | | Основне завдання:  знайти та вивести наступі дані |
| Сутність | Обов'язкові поля | Сутність | Обов'язкові поля |  |
| 2 | Навчальний курс | Назва, наявність іспиту | Практичне заняття | Дата, тема, кількість студентів | Середня кількість студентів, заняття з максимальною кількістю студентів, список тем з певним словом у назві |

1. 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).

3. Сортування вибором (задача підвищеної складності)

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.

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.

5. Вирівнювання рядку

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

6. Абревіатура

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

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. Індивідуальне завдання

Клас Topic

|  |
| --- |
| **package** task1;  **import** java.util.StringTokenizer;  **public** **class** Topic {  **private** **int**[] date;  **private** String subject;  **private** **int** students;  **public** Topic(**int**[] date, String subject, **int** students) {  **this**.date = date;  **this**.subject = subject;  **this**.students = students;  }  **public** **int**[] getDate() {  **return** date;  }  **public** **void** setDate(**int**[] date) {  **this**.date = date;  }  **public** String getSubject() {  **return** subject;  }  **public** **void** setSubject(String subject) {  **this**.subject = subject;  }  **public** **int** getAmount() {  **return** students;  }  **public** **void** setAmount(**int** amount) {  **this**.students = amount;  }  **public** **void** fullInfo() {  System.***out***.println("Date: " + date[0] + "." + date[1] + "." + date[2] + " Subject: " + subject + " Students: " + students);  }  **public** **boolean** containsWord(String word) {  StringTokenizer st = **new** StringTokenizer(subject);  String s;  **while** (st.hasMoreTokens()) {  s = st.nextToken();  **if** (s.toLowerCase().equals(word.toLowerCase())) {  **return** **true**;  }  }  **return** **false**;  }  } |

Клас Subject

|  |
| --- |
| **package** task1;  **import** java.util.StringTokenizer;  **public** **class** Subject {  **private** Topic[] Practices;  **public** Subject(Topic[] practices, String name, **boolean** exam) {  **super**();  Practices = practices;  **this**.name = name;  **this**.exam = exam;  }  **private** String name;  **private** **boolean** exam;  **public** **double** getAverage() {  **double** average=0;  **for** (Topic practice: Practices) {  average+=practice.getAmount();  }  **return** average/=Practices.length;  }  **public** Topic getMax() {  Topic practice = Practices[0];  **for** (**int** i=1; i<Practices.length; i++) {  **if**(Practices[i].getAmount()>practice.getAmount()) {  practice = Practices[i];  }  }  **return** practice;  }  **public** **void** findWord(String word) {  System.***out***.println("Info about practice containing word " + word);  **for** (Topic practice: Practices) {  **if** (practice.containsWord(word)) {  practice.fullInfo();  }  }  }  **public** **static** **void** main(String[] args) {  Topic[] Practices = {  **new** Topic(**new** **int** []{10,1,2018}, "Deep First Search", 5),  **new** Topic(**new** **int**[]{17,1,2018}, "Bredth First Search", 6),  };  Subject sub = **new** Subject(Practices, "Algorithms & Data Structures", **true**);    System.***out***.println("Students average amount: " + sub.getAverage());  System.***out***.println("Max amount of students practice: " + sub.getMax().getAmount() + " (" + sub.getMax().getSubject() + " topic)");  sub.findWord("Search");  }  } |

Приклад виконання програми:

|  |
| --- |
| Students average amount: 5.5  Max amount of students practice: 6 (Bredth First Search topic)  Info about practice containing word Search  Date: 10.1.2018 Subject: Deep First Search Students: 5  Date: 17.1.2018 Subject: Bredth First Search Students: 6 |

2. Ератосфенове решето

|  |
| --- |
| package Sieve;  import java.util.Scanner;  import java.util.Arrays;  public class Sieve {    static int n=300;  static int[] primes = new int[n];    public static void main(String[] args) {  for (int i=0; i<n; ++i) {  primes[i]=(i);  }  for (int i = 2; i < primes.length; ++i) {  if ((i<n) && (primes[i]!=0) ){  for (int j = 2; i \* j < primes.length; ++j) {  primes[i \* j] = 0;  }  }  }  for (int i:primes) if (i>0) System.out.print(i + " ");  }  } |

Приклад виконання програми:

|  |
| --- |
| 1 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211 223 227 229 233 239 241 251 257 263 269 271 277 281 283 293 |

1. Сортування вибором (задача підвищеної складності)

|  |
| --- |
| **package** Sorting;  **import** java.util.Arrays;  **public** **class** Sorting {  **public** **static** **void** sorting(**int**[] a) {  **int** min;  **int** counter;  **for** (counter=0; counter<a.length; counter++) {  min = counter;  **for** (**int** i=counter; i<a.length; i++) {  **if** (a[i]<a[min]) min = i;  }  **int** temp = a[min];  a[min] = a[counter];  a[counter] = temp;  }  }  **public** **static** **void** main(String[] args) {  **int** n=10;  **int**[] a = **new** **int**[n];  System.***out***.println("Unsorted array: ");  **for** (**int** i = 0; i < n; i++) {  a[i] = (**int**) (Math.*random*() \* 100);  System.***out***.print(a[i] + " ");  }  System.***out***.println();  *sorting*(a);  System.***out***.println("Sorted array: \n" + Arrays.*toString*(a));  }  } |

Приклад виконання програми:

|  |
| --- |
| Unsorted array:  4 49 91 89 92 83 68 21 80 79  Sorted array:  [4, 21, 49, 68, 79, 80, 83, 89, 91, 92] |

1. Знаходження чисел Фібоначчі

|  |
| --- |
| **package** Fibonacci;  **public** **class** Fibonacci {  **private** **static** **long**[] *arr* = **new** **long**[100];  **private** **static** **int** *size* = 0;  **public** **static** **long** fib(**int** n) {  **if** (n<*size*) {  **return** *arr*[n];  }  *generate*(n);  **return** *arr*[n];  }    **public** **static** **void** generate (**int** n) {  **if** (*size*==0) {  *arr*[0]=1;  *arr*[1]=1;  *size* = 2;  **do** {  *arr*[*size*]=(*fib*(*size*-2)+*fib*(*size*-1));  System.***out***.print(*arr*[*size*]+ " ");  *size*++;  } **while**(*fib*(*size*-2)+*fib*(*size*-1)<92);  }  **else** {  **do** {  *arr*[*size*]=(*fib*(*size*-2)+*fib*(*size*-1));  System.***out***.print(*arr*[*size*]+ " ");  *size*++;  } **while**(*size*<=n);  }  System.***out***.println();  }  **public** **static** **void** main(String[] args) {  *fib*(0);  System.***out***.println(*fib*(2));  System.***out***.println(*fib*(10));  System.***out***.println(*fib*(15));  System.***out***.println(*fib*(20));  }  } |

Приклад виконання програми:

|  |
| --- |
| 2 3 5 8 13 21 34 55 89  2  89  144 233 377 610 987  987  1597 2584 4181 6765 10946  10946 |

5. Вирівнювання рядку

|  |
| --- |
| **package** String;  **import** java.util.Scanner;  **public** **class** String {  **static** java.lang.String *input*;  **public** **static** **void** add(**int** n) {  **while** (*input*.length()<n) {  *input* += ".";  }  }  **public** **static** **void** main(java.lang.String[] args) {  Scanner s = **new** Scanner(System.***in***);  *input* = s.nextLine();  *add*(5);  System.***out***.println(*input*);  }  } |

Приклад виконання програми:

|  |
| --- |
| 5  5.... |

6. Абревіатура

|  |
| --- |
| **package** Abbreviation;  **import** java.util.Scanner;  **import** java.util.StringTokenizer;  **public** **class** abbreviation {  **private** **static** String *input*;  **private** **static** String *result*="";  **public** **static** **void** abbr() {  StringTokenizer st = **new** StringTokenizer(*input*);  **while** (st.hasMoreTokens()) {  *result* += st.nextToken().substring(0,1).toUpperCase();  }  }  **public** **static** **void** main(String[] args) {  Scanner s = **new** Scanner(System.***in***);  *input* = s.nextLine();  *abbr*();  System.***out***.print(*result*);  }  } |

Приклад виконання програми:

|  |
| --- |
| asd qw Q s  AQQS |

7. Квадратне рівняння

|  |
| --- |
| **package** Equation;  **public** **class** Equation {  **private** **static** **double** *x1*;  **private** **static** **double** *x2*;  **public** **static** **int** solve(**int** a, **int** b, **int** c) {  **if** (a!=0) {  **double** d = b\*b - 4\*a\*c;  **if** (d<0) **return** 0;  **else**  {  *x1* = (-b + Math.*sqrt*(d)) / 2 / a;  *x2* = (-b - Math.*sqrt*(d)) / 2 / a;  **if** (*x1* == *x2*) **return** 1;  **return** 2;  }  }  **else** {  **if**(b!=0) {  *x1* = -c/b;  **return** 1;  }  **else** {  **if** (c!=0) **return** 0;  **return** -1;  }  }  }  **public** **static** **double** getX1() {  **return** *x1*;  }  **public** **static** **double** getX2() {  **return** *x2*;  }  **public** **static** **void** main(String[] args) {  **int** a=1;  **int** b=5;  **int** c=4;  *solve*(a,b,c);  System.***out***.print(*getX1*() + " " + *getX2*());  }  } |

Приклад виконання програми:

|  |
| --- |
| -1.0 -4.0 |

Висновок

При виконанні цієї лабораторної я більш суттєво ознайомився з механізмом роботи класів, детільніше ознайомився з рядками — обєктами класу java.lang.String.