МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Криворізький національний університет

Кафедра моделювання та програмного забезпечення

ЛАБОРАТОРНА РОБОТА №7

З дисципліни «Програмування на основі Java технологій»

Тема: «Використання утилітарних пакетів (Java.Lang та Java.Util) у мові Java»

Виконав студент групи ІПЗ-21-2

Губарєв Р.В.

Перевірив викладач

Котов І.А.

Гриценко А.М.

Карабут Н.О.

Кривий Ріг

2022

1. **Основні відомості про пакети Java.Lang та Java.Util**

Пакет java.lang є основним пакетом, засоби якого використовуються при розробці програм на мові Java. Пакет містить найбільш широко використовувані інтерфейси та класи, без яких неможливо написати програму на Java.

Цей пакет автоматично імпортується у всі програми. Тому, щоб доступитись до засобів пакету, не обов’язково включати цей пакет в програму з допомогою рядка

**import** java.lang.\*;

Пакет включає засоби для розширення можливостей примітивних типів даних, складових елементів мови, роботи з рядками, потоками та інше.

Пакет java.lang містить наступні класи

* Boolean – клас-обгортка (клас-оболонка) для типу boolean;
* Byte – клас-обгортка для типу byte;
* Character – клас-обгортка для типу char;
* Character.Subset – визначає специфічні множини символів набору Unicode;
* Class – інкапсулює стан часу виконання класу чи інтерфейсу;
* ClassLoader – визначає об’єкт, що відповідає за порядок завантаження класів;
* ClassValue – використовується для зв’язку значення з типом;
* Compiler – забезпечує створення середовищ в яких байт-код компілюється у виконавчий код;
* Double – клас-обгортка для типу double;
* Enum – клас, що служить суперкласом для всіх зчислень (enum) у програмі;
* Float – клас-обгортка для типу float;
* InheritableThreadLocal – призначений для створення локальних змінних потоків виконання, які можуть успадковуватись;
* Integer – клас-обгортка для типу int;
* Long – клас-обгортка для типу long;
* Math – містить функції та константи для проведення математичних обчислень над числовими типами;

І ще багато інших класів.

Пакет java.util містить широкий спектр засобів, що дають потужні функціональні можливості для розробки програм мовою Java. Однією з цих можливостей є керування та організація роботи з наборами (групами) об’єктів, які ще називаються колекціями. У мові Java колекції були впроваджені починаючи з версії J2SE 1.2. Колекції утворюють так званий каркас, який називається Java Collection Framework і представляє собою складну ієрархію класів та інтерфейсів.  
У каркасі колекцій реалізовано набір стандартних операцій над різними відомими видами груп об’єктів, до яких належать:

* черга;
* множина;
* список;
* масив;
* хеш-таблиця;
* дерево.

1. **Основні відомості про консольну та файлову систему введення-виводу в мові Java**

У мові програмування Java ввід/вивід інформації базується на понятті потоку. Потік – це абстрактне поняття, яке символізує джерело або приймач даних, що може передавати або отримувати деяку інформацію. Будь-який потік приховує операції над даними, що виконуються на нижчих рівнях безпосередньо в пристроях вводу/виводу.

Відповідно до призначення потоків, класифікуються і класи в мові Java. Одні класи реалізують операції вводу, інші реалізують операції виводу. Щоб використовувати класи потоків вводу/виводу потрібно імпортувати пакет java.io

У мові Java розрізняють два види потоків:

* *байтові потоки*. Це аналог потоків двійкових даних, які дозволяють компактно зберігати інформацію;
* *символьні потоки*. Це потоки, що представлені зручним способом (для людей) кодування інформації у вигляді зрозумілих текстових символів. У багатьох мовах програмування символьні потоки асоціюються з текстовим форматом представлення інформації.

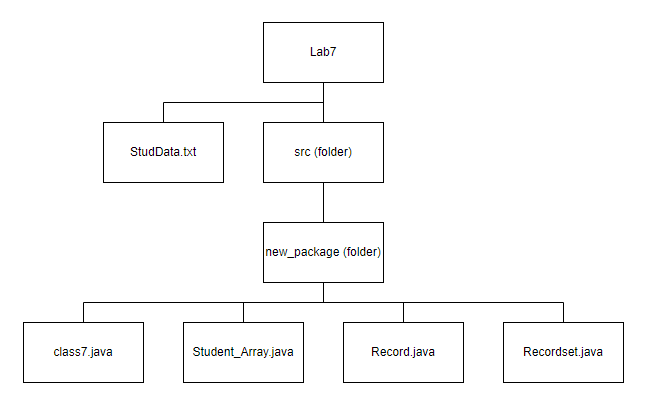
Класи, що реалізують байтові потоки вводу успадковані від абстрактного класу InputStream:

* InputStream – абстрактний клас, що описує потік вводу. Даний клас є базовим для всіх інших класів системи вводу;
* BufferedInputStream – клас, що описує буферизований потік вводу;
* ByteArrayInputStream – клас, що описує потік вводу, який читає байти з масиву;
* DataInputStream –клас, що реалізує методи для читання даних стандартних типів, визначених у Java (int, double, float і т.д.);
* FileInputStream – клас, що реалізує потік вводу, який читає дані з файлу;
* FilterInputStream – це є реалізація абстрактного класу InputStream;
* ObjectInputStream – клас, що реалізує потік вводу об’єктів;
* PipedInputStream – клас, що відповідає каналу вводу;
* PushbackInputStream – клас, що відповідає потоку вводу, який підтримує повернення одного байту назад в потік вводу;
* SequenceInputStream – клас, що реалізує потік вводу, який складається з двох або більше потоків вводу, дані з яких читаються по черзі.

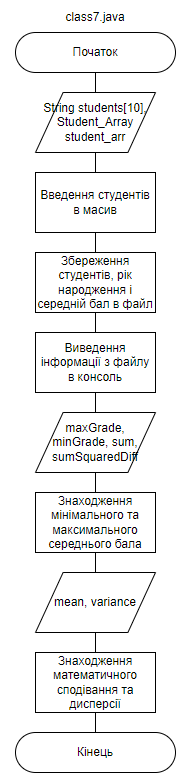
Класи, що реалізують байтові потоки виводу успадковані від абстрактного класу OutputStream:

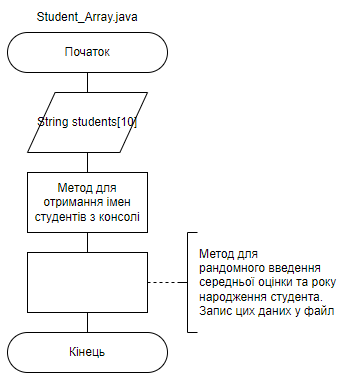
* OutputStream – абстрактний клас, що описує потік виводу. Усі інші класи системи виводу є підкласами класу OutputStream;
* BufferedOutputStream – клас, що імплементує буферизований потік виводу;
* ByteArrayOutputStream – клас, що реалізує потік виводу, який записує байти в масив;
* DataOutputStream – клас, що реалізує потік виводу, який містить методи для читання даних стандартних типів, визначених у Java (int, float, double тощо);
* FileOutputStream – клас, що відповідає потоку виводу, який записує дані у файл;
* FilterOutputStream – клас, який реалізує абстрактний клас OutputStream;
* ObjectOutputStream – клас, що відповідає потоку виводу об’єктів;
* PipedOutputStream – клас, що асоціюється з каналом виводу;
* PrintStream – клас, що представляє собою потік виводу, який містить методи print() та println().

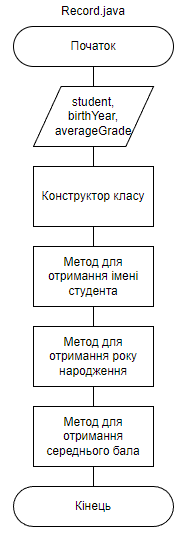
1. **Розгорнуту структуру програмного проекту у вигляді деревоподібної схеми**

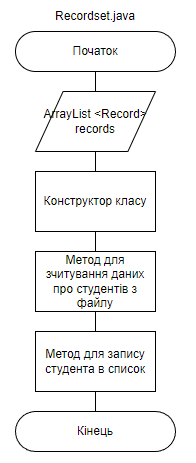
****

1. **Блок-схеми алгоритмів роботи методів класів**

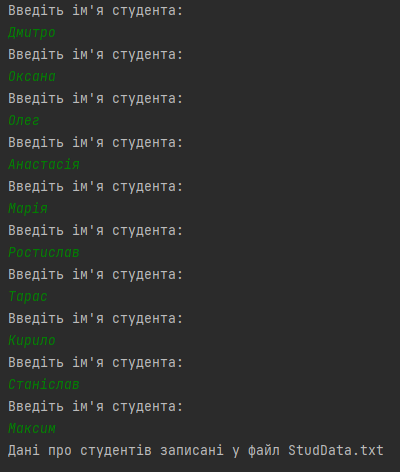


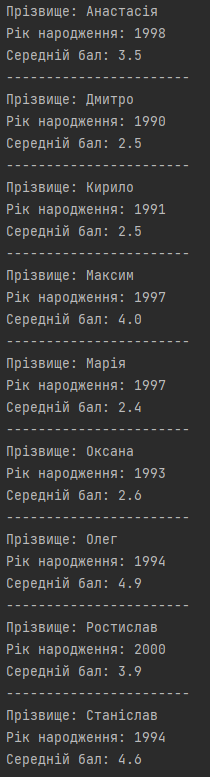


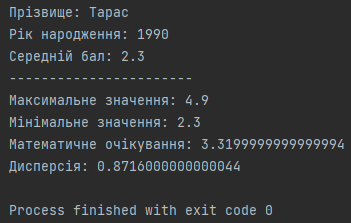


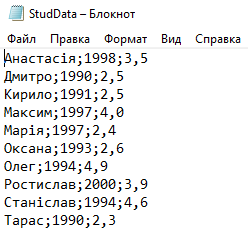


1. **Скріншот екрану програми з результатом роботи програми**









1. **Текст вихідних кодів програми**

|  |
| --- |
| **class7.java**  package new\_package;  import java.util.\*;  public class class7 {  public static void main(String[] args) {  String[] students = new String[10];  Student\_Array student\_arr = new Student\_Array();   student\_arr.getStudent(students);  student\_arr.*writeStudents*(students);   Recordset recordset = new Recordset();  recordset.loadRecords("StudData.txt");  ArrayList<Record> records = recordset.getRecords();   // Виведення завантажених елементів типу Record в консоль  for (Record record : records) {  System.*out*.println("Прізвище: " + record.getStudent());  System.*out*.println("Рік народження: " + record.getBirthYear());  System.*out*.println("Середній бал: " + record.getAverageGrade());  System.*out*.println("-----------------------");  }   // Розрахунок статистичних показників  double maxGrade = Double.*MIN\_VALUE*;  double minGrade = Double.*MAX\_VALUE*;  double sum = 0;  double sumSquaredDiff = 0;   for (Record record : records) {  double grade = record.getAverageGrade();   if (grade > maxGrade) {  maxGrade = grade;  }   if (grade < minGrade) {  minGrade = grade;  }   sum += grade;  sumSquaredDiff += Math.*pow*(grade, 2);  }   double mean = sum / records.size();  double variance = (sumSquaredDiff / records.size()) - Math.*pow*(mean, 2);   System.*out*.println("Максимальне значення: " + maxGrade);  System.*out*.println("Мінімальне значення: " + minGrade);  System.*out*.println("Математичне очікування: " + mean);  System.*out*.println("Дисперсія: " + variance);  } } |
| **Student\_Array.java**  package new\_package;  import java.util.\*; import java.lang.\*; import java.io.\*;  public class Student\_Array {  String[] students = new String[10];   public void getStudent(String[] student){  Scanner in = new Scanner(System.*in*);   for (int i = 0; i < 10; i++){  System.*out*.println("Введіть ім'я студента: ");  student[i] = in.next();  }   Arrays.*sort*(student);  }   public static void writeStudents(String[] student) {  int[] birthYears = new int[10];  double[] averageGrades = new double[10];   Random random = new Random();   // Заповнення масивів з випадковими даними  for (int i = 0; i < student.length; i++) {  birthYears[i] = random.nextInt(11) + 1990; // діапазон від 1990 до 2000  averageGrades[i] = 2.0 + (random.nextDouble() \* 3.0); // діапазон від 2.0 до 5.0  }   // Запис даних студентів у файл  try (PrintWriter writer = new PrintWriter(new FileWriter("StudData.txt"))) {  for (int i = 0; i < student.length; i++) {  String formattedGrade = String.*format*("%.1f", averageGrades[i]);  writer.println(student[i] + ";" + birthYears[i] + ";" + formattedGrade);  }  System.*out*.println("Дані про студентів записані у файл StudData.txt\n");  } catch (IOException e) {  System.*out*.println("Виникла помилка при записі у файл: " + e.getMessage() + "\n");  }  } } |
| **Record.java**  package new\_package;  public class Record {  private String student;  private int birthYear;  private double averageGrade;   public Record(String student, int birthYear, double averageGrade) {  this.student = student;  this.birthYear = birthYear;  this.averageGrade = averageGrade;  }   public String getStudent() {  return student;  }   public int getBirthYear() {  return birthYear;  }   public double getAverageGrade() {  return averageGrade;  } } |
| **Recordset.java**  package new\_package;  import java.io.\*; import java.util.\*;  public class Recordset {  private ArrayList<Record> records;   public Recordset() {  records = new ArrayList<>();  }   public void loadRecords(String fileName) {  try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {  String line;  while ((line = reader.readLine()) != null) {  String[] data = line.split(";");  String student = data[0];  int birthYear = Integer.*parseInt*(data[1]);  double averageGrade = Double.*parseDouble*(data[2].replace(",", ".")); // Заміна коми на крапку   Record record = new Record(student, birthYear, averageGrade);  records.add(record);  }  } catch (IOException e) {  System.*out*.println("Виникла помилка при читанні з файлу: " + e.getMessage());  } catch (NumberFormatException e) {  System.*out*.println("Виникла помилка при перетворенні рядка на число: " + e.getMessage());  }  }   public ArrayList<Record> getRecords() {  return records;  } } |

1. **Короткі висновки**

В цій лабораторній роботі я навчився працювати з такими пакетами, як Java.Lang і Java.Util у мові програмування Java

1. **Список використаних джерел**

* <https://www.bestprog.net/uk/2021/09/26/java-the-java-lang-package-general-information-ua/>
* <https://www.bestprog.net/uk/2022/03/23/java-introduction-to-java-collections-framework-ua/>