Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра «Електронних обчислювальних машин»



Звіт

з лабораторної роботи № 3

з дисципліни: «Кросплатформенні засоби програмування»

на тему: «Класи та пакети»

**Виконав:**

студент групи КІ-35

Скіпар І.В.

**Прийняв:**

доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2022

**Мета роботи:** ознайомитися з процесом розробки класів та пакетів мовою Java.

**Завдання (варіант № 10)**

1. Написати та налагодити програму на мові Java, що реалізує у вигляді класу предметну область згідно варіанту. Програма має задовольняти наступним вимогам:

* програма має розміщуватися в пакеті Група.Прізвище.Lab3;
* клас має містити мінімум 3 поля, що є об’єктами класів, які описують складові частини предметної області;
* клас має містити кілька конструкторів та мінімум 10 методів;
* для тестування і демонстрації роботи розробленого класу розробити клас-драйвер;
* методи класу мають вести протокол своєї діяльності, що записується у файл;
* розробити механізм коректного завершення роботи з файлом (не надіятися на метод finalize());
* програма має володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.

1. Автоматично згенерувати документацію до розробленого пакету.
2. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації.
3. Дати відповідь на контрольні запитання:
4. Синтаксис визначення класу.
5. Синтаксис визначення методу.
6. Синтаксис оголошення поля.
7. Як оголосити та ініціалізувати константне поле?
8. Які є способи ініціалізації полів?
9. Синтаксис визначення конструктора.
10. Синтаксис оголошення пакету.
11. Як підключити до програми класи, що визначені в зовнішніх пакетах?
12. В чому суть статичного імпорту пакетів?
13. Які вимоги ставляться до файлів і каталогів при використанні пакетів?

**Текст програми**

House.java

*/\*\*  
 \* Skipar package  
 \*/*package KI35.Skipar.Lab3;  
  
import java.io.FileNotFoundException;  
import java.io.PrintWriter;  
  
*/\*\*  
 \* Class <code>House</code> implement house  
 \** ***@author*** *Ілля  
 \** ***@version*** *1.0  
 \*/*public class House {  
  
 private int floors;  
 private String street;  
 private int citizens;  
 private int number;  
 private PrintWriter fout;  
  
 */\*\*  
 \* Constructor  
 \** ***@throws*** *FileNotFoundException Not found log-file  
 \*/* public House() throws FileNotFoundException {  
 fout = new PrintWriter("Log.txt");  
 floors = 0;  
 street = "Unknown";  
 citizens = 0;  
 number = 0;  
 }  
  
 */\*\*  
 \* Constructor  
 \** ***@param*** *street the street on which the house is located  
 \** ***@param*** *number house number  
 \** ***@param*** *floors count of floors in the house  
 \** ***@param*** *citizens count of citizens in the house  
 \** ***@throws*** *FileNotFoundException Not found log-file  
 \*/* public House(String street, int number, int floors, int citizens) throws FileNotFoundException {  
 this.number = number;  
 this.street = street;  
 this.floors = floors;  
 this.citizens = citizens;  
 fout = new PrintWriter("Log.txt");  
 }  
  
 */\*\*  
 \* Method returns house's current street name  
 \** ***@return*** *house's current street name  
 \*/* public String getStreet() {  
 return street;  
 }  
  
 */\*\*  
 \* Method set or change house's current street name  
 \** ***@param*** *street the street on which the house is located  
 \*/* public void setStreet(String street) {  
 if (this.street.equals("Unknown"))  
 fout.print("House is on " + street + " street\n");  
 else fout.print("Street name changed to " + street + "\n");  
 this.street = street;  
 fout.flush();  
 }  
  
 */\*\*  
 \* Method return count of floors in the house  
 \** ***@return*** *count of house floors  
 \*/* public int getFloors() {  
 return floors;  
 }  
  
 */\*\*  
 \* Method set or change count of floors in the house  
 \** ***@param*** *floors count of floors in the house  
 \*/* public void setFloors(int floors) {  
 if (this.floors != 0 && this.floors>floors )  
 fout.print( this.floors-floors + (this.floors-floors!=1?" floors were":" floor was") + " destroyed\n");  
 else fout.print( floors-this.floors + (floors-this.floors!=1?" floors were":" floor was") + " built\n");  
 this.floors = floors;  
 fout.flush();  
 }  
  
 */\*\*  
 \* Method return count of citizens in the house  
 \** ***@return*** *count of citizens in the house  
 \*/* public int getCitizens() {  
 return citizens;  
 }  
  
 */\*\*  
 \* Method set or change count of citizens in the house  
 \** ***@param*** *citizens count of citizens in the house  
 \*/* public void setCitizens(int citizens) {  
 if (this.citizens<citizens)  
 fout.print(citizens-this.citizens + " new citizens in the house\n");  
 else fout.print(this.citizens-citizens + " citizens leave the house\n");  
 this.citizens = citizens;  
 fout.flush();  
 }  
  
 */\*\*  
 \* Method return house number  
 \** ***@return*** *house number  
 \*/* public int getNumber() {  
 return number;  
 }  
  
 */\*\*  
 \* Method set or change current house number  
 \** ***@param*** *number house number  
 \*/* public void setNumber(int number) {  
 if (this.number == 0)  
 fout.print("House is at " + number + " " + street + " street\n");  
 else fout.print("House number changed to " + number + "\n");  
 this.number = number;  
 fout.flush();  
 }  
  
 */\*\*  
 \* Method return average number of citizens on the floor  
 \** ***@return*** *average number of citizens on the floor  
 \*/* public float citizenOnFloor (){  
 return (float) this.citizens/this.floors;  
 }  
 */\*\*  
 \* Method close log-file  
 \*/* public void closeFile () {  
 fout.close();  
 }  
  
}

HouseApp.java

*/\*\*  
 \* Skipar package  
 \*/*package KI35.Skipar.Lab3;  
  
import java.io.IOException;  
import static java.lang.System.*out*;  
  
*/\*\*  
 \* House Application class implements main method for House  
 class possibilities demonstration  
 \** ***@author*** *Ілля  
 \** ***@version*** *1.0  
 \*/*public class HouseApp {  
  
 */\*\*  
 \** ***@param*** *args args  
 \** ***@throws*** *IOException i/o exception  
 \*/* public static void main(String[] args) throws IOException {  
 House my\_house = new House("Bandera", 7,12, 156);  
  
 *out*.print(my\_house.getStreet() + "\n");  
 my\_house.setStreet("Chornovola");  
  
 my\_house.setFloors(13);  
 *out*.print(my\_house.getFloors() + "\n");  
  
 my\_house.setCitizens(166);  
 *out*.print(my\_house.citizenOnFloor() + "\n");  
  
 my\_house.setFloors(2);  
 *out*.print(my\_house.getFloors() + "\n");  
  
 my\_house.setCitizens(17);  
 *out*.print(my\_house.citizenOnFloor() + " " + my\_house.getCitizens() + "\n");  
  
 my\_house.setNumber(9);  
 *out*.print(my\_house.getNumber() + "\n");  
  
 my\_house.closeFile();  
 }  
  
}

**Результат виконання програми**

Консоль

Bandera

13

12.769231

2

8.5 17

9

Log.txt

Street name changed to Chornovola

1 floor was built

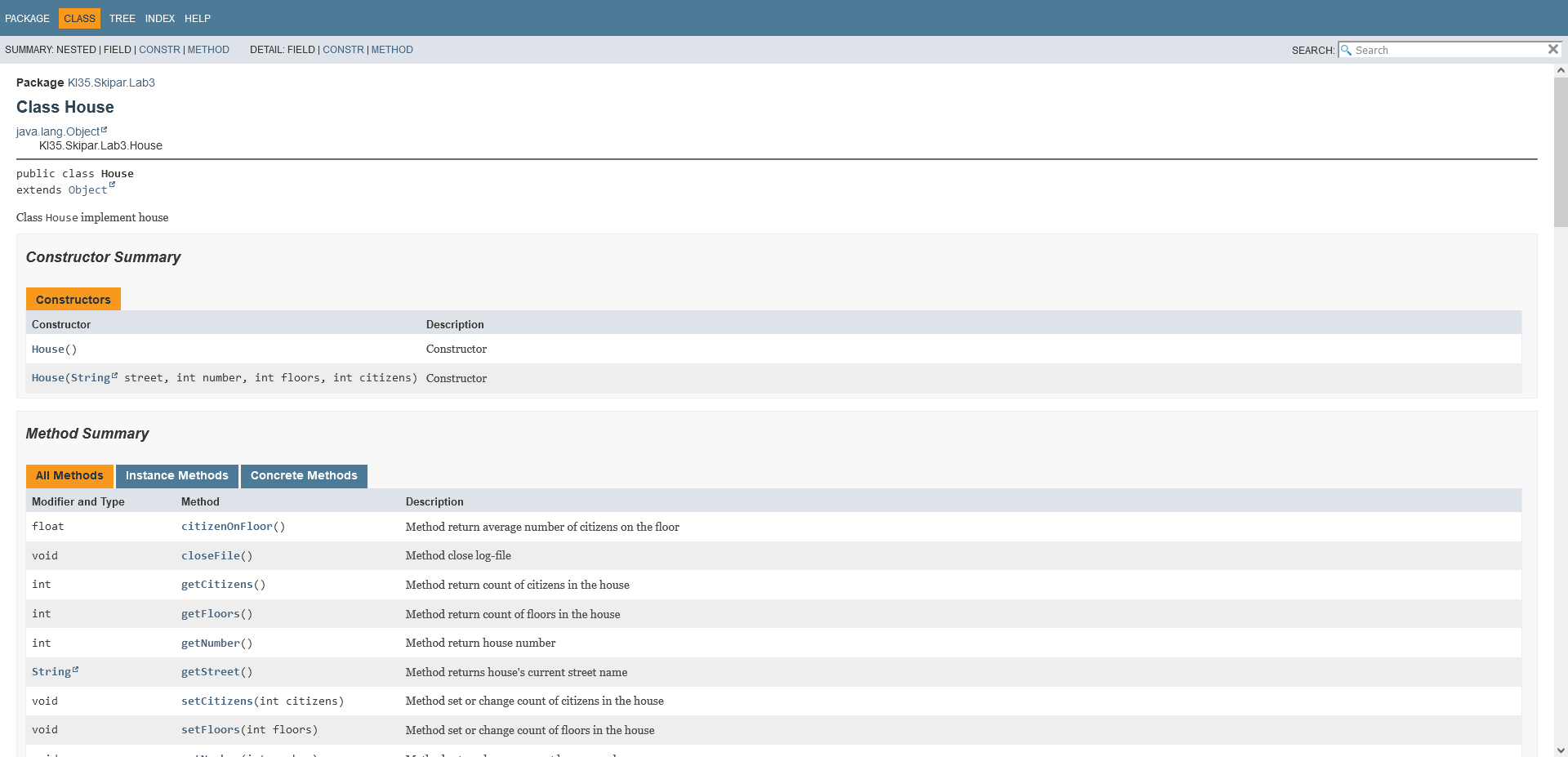
10 new citizens in the house

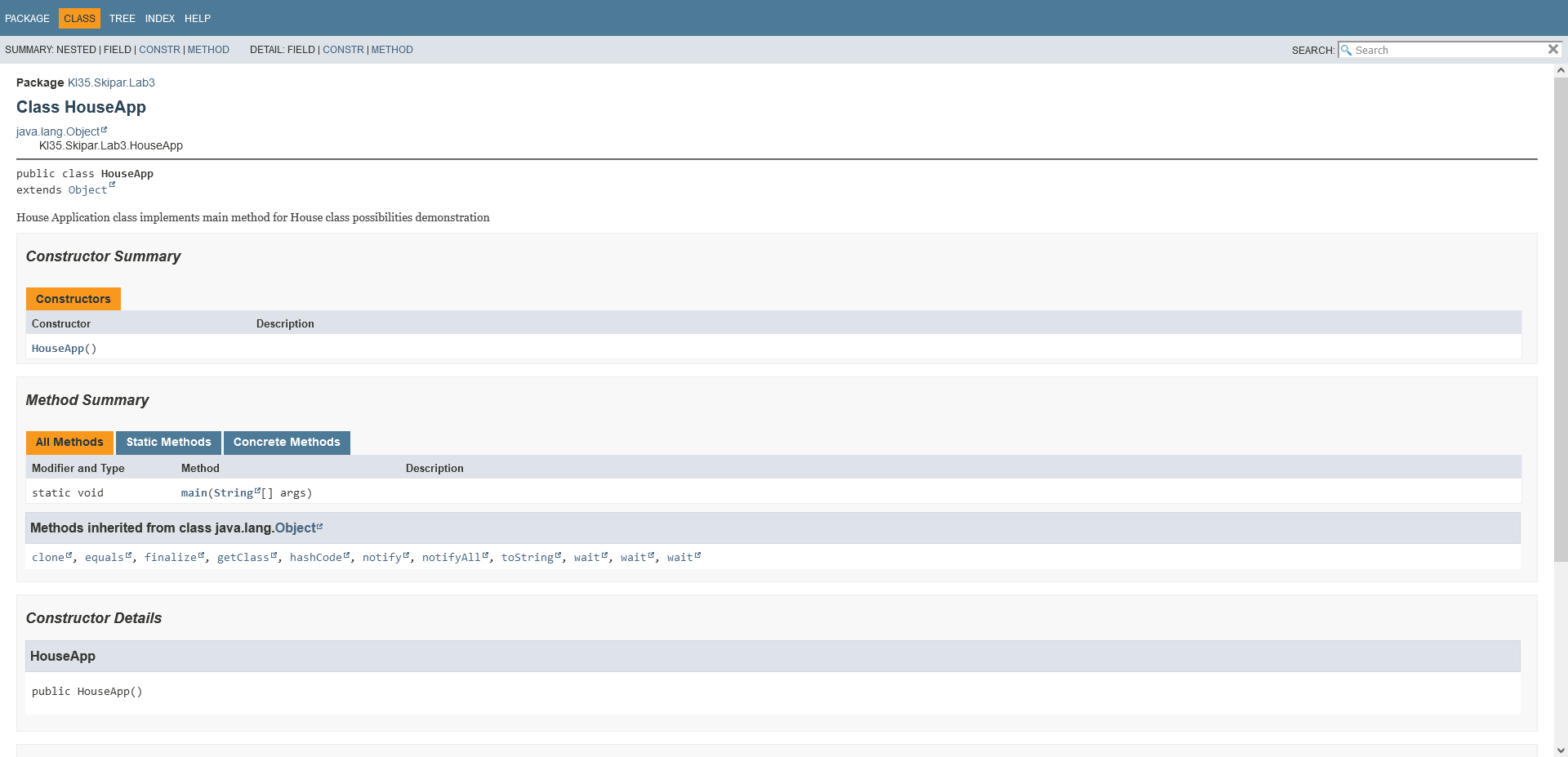
11 floors were destroyed

149 citizens leave the house

House number changed to 9

**Фрагмент згенерованої документації**

[](doc/KI35/Skipar/Lab3/House.html)

[](doc/KI35/Skipar/Lab3/HouseApp.html)

**Відповіді на контрольні запитання**

1. Спершу пишемо специфікатор доступу (public, private, protected), потім ключове слово «class», назву класу і у фігурних дужках його реалізацію.
2. Метод визначається як функція. Специфікатор доступу, тип, назва методу, в дужках параметри які приймає метод, можливі exceptions, реалізація методу.
3. Специфікатор доступу, тип, назва поля.
4. Константне поле можна ініціалізувати за допомогою ключового слова «static».
5. Є три способи ініціалізувати поле: у конструкторі, явно при оголошенні, у блоці ініціалізації
6. Конструктор визначаємо в такому порядку: специфікатор доступу, назва класу, в круглих дужках параметри які приймає цей конструктор, у фігурних дужках тіло конструктора.
7. Пакети оголошуються за допомогою ключового слова package, тобто «package назва пакету{.назва підпакету»}.
8. Підключити до програми класи, що визначені в зовнішніх пакетах можна за допомогою ключового слова import.
9. Суть статичного імпорту пакетів полягає в тому що він дозволяє не вживати явно назву класу при звертанні до статичного поля або методу класу.
10. Використання пакетів вимагає, щоб файли і каталоги проекту та їх ієрархія були строго структурованими. Так назви пакету і його підпакетів мають співпадати з назвами каталогів, де вони розміщуються.

**Висновок**

В результаті виконання лабораторної роботи я навчився працювати з класами та потоками в Java. Частина коду автоматично згенерована середовищем IntelliJ IDEA. Пакет створений в лабораторній роботі може використовуватись в подальшому для обліку будинків в містах та кількості жителів в них.