МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
Національний аерокосмічний університет ім. М.Є. Жуковського
“Харківський авіаційний інститут”

Кафедра комп’ютерних систем, мереж і кібербезпеки

Лабораторна робота № 6

з дисципліни “Технології програмування”

# ПОТОКИ ВВЕДЕННЯ-ВИВЕДЕННЯ. РОБОТА З ФАЙЛАМИ. ЗАСТОСУВАННЯ КОЛЕКЦІЙ

XAI.503.525i1.22o.125, ПЗ

Виконав студент гр. 525і1 Проценко Д.І.
(№ групи) (П.І.Б.)

(підпис, дата)

Перевірив ст. викладач каф.503
(науковий ступінь, вчене звання, посада)

Здоровець Ю. В.

(підпис, дата) (П.І.Б.)

Харків 2022

**Тема роботи:** Потоки введення-виведення. Робота з файлами. Застосування Колекцій

**Мета роботи:** Вивчення основ застосування класів пакета java.io для організації введення-виведення даних в додатках на мові Java; освоєння принципів роботи з текстовими і бінарними файлами; вивчення та освоєння механізму серіалізації-десеріалізації об'єктів класів; набуття практичних навичок розробки програм на мові Java у середовищі IntelliJ IDEA, в яких передбачено роботу з потоками та файлами; набуиия практичних навичок застосовування колекцій; залучити до самостійної діяльності та прийняття рішень при формуванні програмного коду.

# Завдання 1.

## Частина 1. Умови завдання

**Завдання:**

Написати програму мовою Java відповідно до заданого варіанту.

- передбачити в програмі виняткові ситуації;
- Організувати перегляд вмісту файлу.
- Організувати читання і обробку даних з файлу відповідно до індивідуального завдання.
- Зберегти отримані результати в новий текстовий файл.
- Вивести інформацію про файл (розмір, дата створення, права доступу та ін.).
- Зробити висновки.

Рис. 1 Умови завдання

**Варіанти завдань**

Дано файл[1], що містить відомості про іграшки, указується назва іграшки (лялька, кубики, м'яч, конструктор), її вартість і вікові межі дітей, для яких іграшка призначена (наприклад, для дітей від двох до п'яти років). Крім того, для ляльки зазначено її розмір у сантиметрах, для кубиків – їхня кількість у наборі, для м'яча – його вага у грамах, для конструктора – кількість конструкцій, які з нього можна збудувати згідно інструкції.

У новому файлі необхідно одержати наступні відомості:

**Варіант 1.**

Перелік іграшок, ціна яких не перевищує вказану і які підходять дітям 5 років у порядку зростання ціни

Рис. 2 Персональне завдання

## Частина 2. Текст програми

```
import jdk.jshell.spi.ExecutionControl;

import java.io.*;
import java.util.*;
import java.util.stream.Collectors;



public class Main {

    public static void main(String[] args) throws IOException {
        //считываем данные из файла и добавляем в список
        List <Toy> toys  = readToysFromFile("toys.txt");

        System.out.println("\nINPUT FILE:\n");
        printToys(toys);

        //Фильтруем список по заданым критериям(необх. возраст 5, цена не больше
50) и сортируем по цене
        List <Toy> filtersToys = toys.stream()
                .filter(toy -> toy.getPrice() <= 50 &&
toy.getAgeRange().contains(5))
                .sorted(Comparator.comparing(Toy::getPrice))
                .collect(Collectors.toList());

        System.out.println("\nOUTPUT FILE:\n");
        printToys(filtersToys);

        //вывод полученого списка в новый файл
        WriteToysToFile(filtersToys,"filtered_toys.txt");
    }

    public static List<Toy> readToysFromFile(String filename) throws IOException{
        //считывание данных из файла и парсинг их в список объектов Toy
        List<Toy> toys = new ArrayList<>();
        try(BufferedReader reader = new BufferedReader(new FileReader(filename)))
{
            String line;
            while((line = reader.readLine()) != null)
            {
                String[] parts = line.split(",");
                String name = parts[0];
                int price = Integer.parseInt(parts[1]);
                int minAge = Integer.parseInt(parts[2]);
                int maxAge = Integer.parseInt(parts[3]);
                AgeRange ageRange = new AgeRange(minAge, maxAge);
                Toy toy;
                if(name.equals("doll")){
                    int size = Integer.parseInt(parts[4]);
                    toy = new Doll(name,price,ageRange,size);
                }
                else if(name.equals("ball")){
                    int weight = Integer.parseInt(parts[4]);
                    toy = new Ball(name,price,ageRange,weight);
                } else if (name.equals("blocks")) {
                    int numBlocks = Integer.parseInt(parts[4]);
                    toy = new Blocks(name, price, ageRange, numBlocks);
                } else if (name.equals("constructor")) {
                    int numConstructions = Integer.parseInt(parts[4]);
                    toy = new Constructor(name, price, ageRange,
numConstructions);
```

```
                }
                else{
                    throw new IllegalArgumentException("Invalid name toy: " +
name);
                }
                try{
                    toys.add(toy);
                }catch (IllegalArgumentException e)
                {
                    System.err.println("Error parcing toy data: "+
e.getMessage());
                }
            }
        }
        return  toys;
    }

    public static void printToys(List<Toy> toys) {
        for (Toy toy : toys) {
            System.out.print(toy.getName() + "," + toy.getPrice() + "," +
                    toy.getAgeRange().getMinAge() + "," +
toy.getAgeRange().getMaxAge());
            if (toy instanceof Doll) {
                System.out.print("," + ((Doll) toy).getSize() + "\n");
            } else if (toy instanceof Ball) {
                System.out.print("," + + ((Ball) toy).getWeight() + "\n");
            } else if (toy instanceof Blocks) {
                System.out.print("," + ((Blocks) toy).getNumBlocks() + "\n");
            } else if (toy instanceof Constructor) {
                System.out.print("," + ((Constructor) toy).getNumConstructions() +
"\n");
            }
            //System.out.println();
        }
    }

    private static void WriteToysToFile(List<Toy> toys, String filename) throws
IOException{
        //Вывод полученого списка в файл
        try(BufferedWriter writer = new BufferedWriter(new FileWriter(filename))){
            for (Toy toy: toys)
            {
                writer.write(toy.getName() + "," + toy.getPrice() + "," +
                        toy.getAgeRange().getMinAge() + "," +
toy.getAgeRange().getMaxAge());
                if(toy instanceof Doll)
                {
                    writer.write("," + ((Doll)toy).getSize());
                } else if (toy instanceof Ball) {
                    writer.write("," + ((Ball) toy).getWeight());
                } else if (toy instanceof Blocks) {
                    writer.write("," + ((Blocks) toy).getNumBlocks());
                } else if (toy instanceof Constructor) {
                    writer.write("," + ((Constructor) toy).getNumConstructions());
                }
                writer.newLine();
            }
        }
    }
}

class Toy
{
    private String name;
```

```
    private int price;

    private AgeRange ageRange;

    public Toy(String name, int price, AgeRange ageRange) {
        this.name = name;
        this.price = price;
        this.ageRange = ageRange;
    }

    public String getName() {
        return name;
    }

    public int getPrice() {
        return price;
    }

    public AgeRange getAgeRange(){
        return ageRange;
    }
}

class Doll extends Toy
{
    private int size;


    public Doll(String name, int price, AgeRange ageRange, int size) {
        super(name, price, ageRange);
        this.size = size;
    }

    public int getSize() {
        return size;
    }
}

class Blocks extends Toy{
    private  int numBlocks;

    public Blocks(String name, int price, AgeRange ageRange, int numBlocks) {
        super(name, price, ageRange);
        this.numBlocks = numBlocks;
    }

    public int getNumBlocks() {
        return numBlocks;
    }
}

class Ball extends Toy{
    private int weight;

    public Ball(String name, int price, AgeRange ageRange, int weight) {
        super(name, price, ageRange);
        this.weight = weight;
    }

    public int getWeight() {
        return weight;
    }
}
```

```
class Constructor extends Toy{
    private int numConstructions;

    public Constructor(String name, int price, AgeRange ageRange, int
numConsructions) {
        super(name, price, ageRange);
        this.numConstructions = numConsructions;
    }

    public int getNumConstructions() {
        return numConstructions;
    }
}
class AgeRange{
    private int minAge;

    private int maxAge;

    public AgeRange(int minAge, int maxAge){
        this.minAge = minAge;
        this.maxAge = maxAge;
    }

    public int getMinAge() {
        return minAge;
    }

    public int getMaxAge() {
        return maxAge;
    }
    public boolean contains(int Age)
    {
        return Age >= minAge && Age <= maxAge;
    }
}
```

## Частина 3. Діаграма структури класів

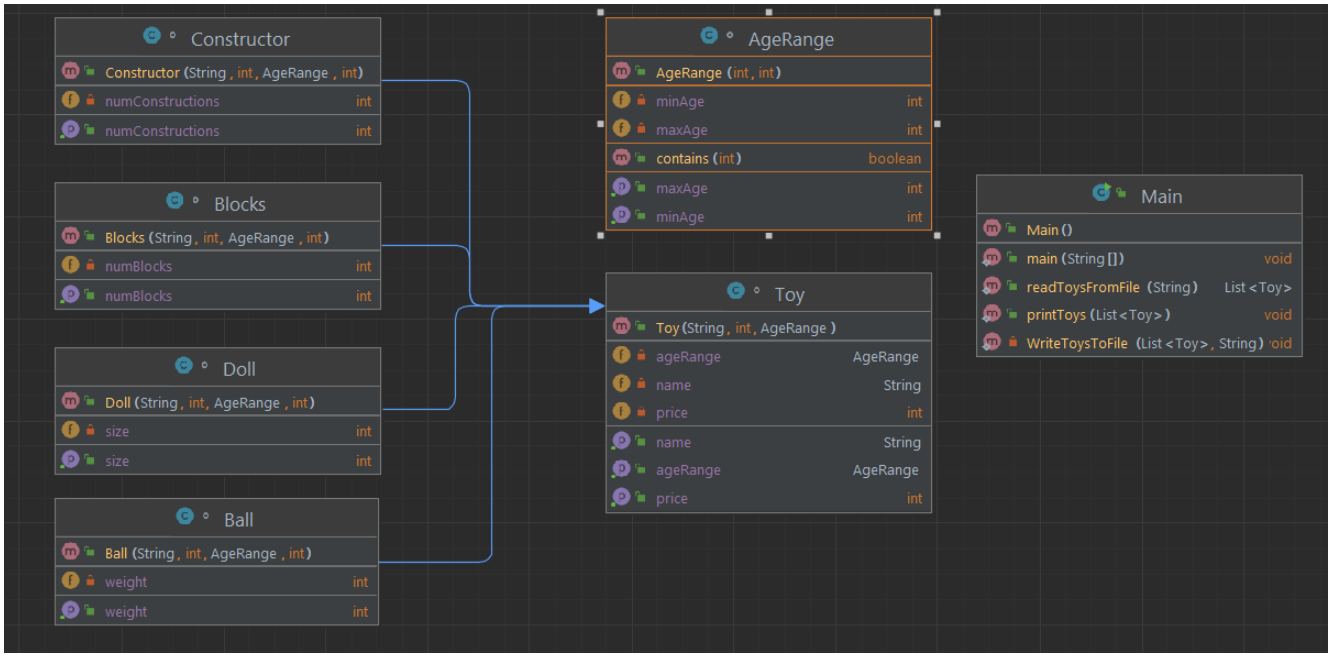


Рис. 3 Діаграма классів

## Частина 4. Скріншоти виконання завдання

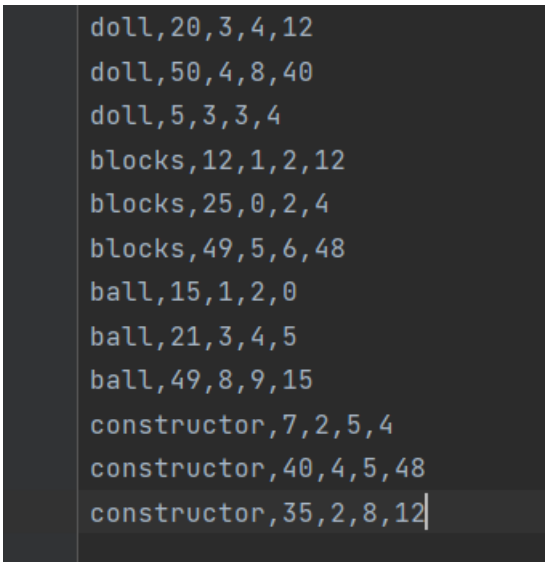


Рис. 4 Скріншот виконання завдання (1.1 частина)

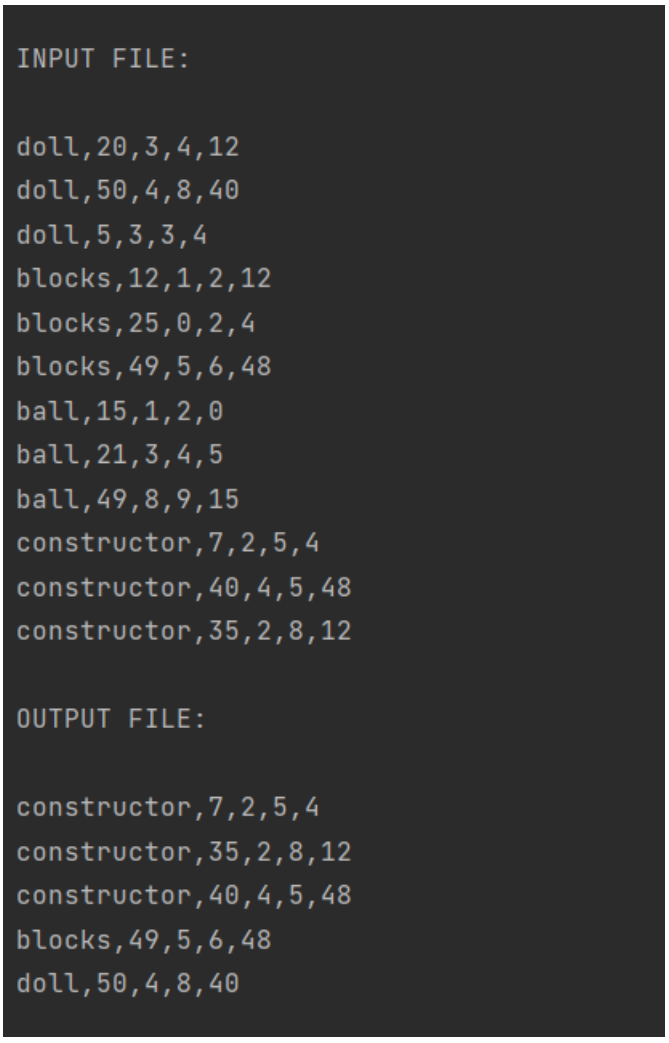


Рис. 5 Скріншот виконання завдання (1.2 частина)

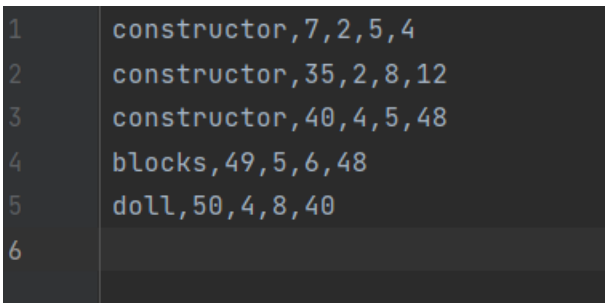


Рис. 6 Скріншот виконання завдання (1.3 частина)

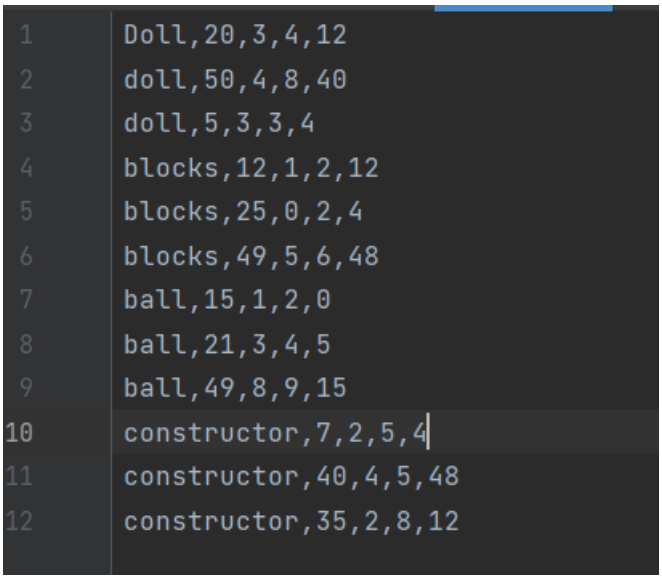


Рис. 7 Скріншот виконання завдання (2.1 частина)

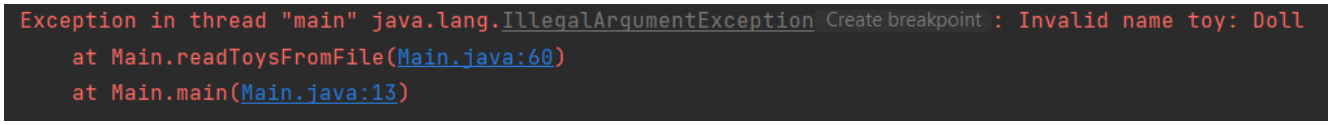


Рис. 8 Скріншот виконання завдання (2.2 частина)

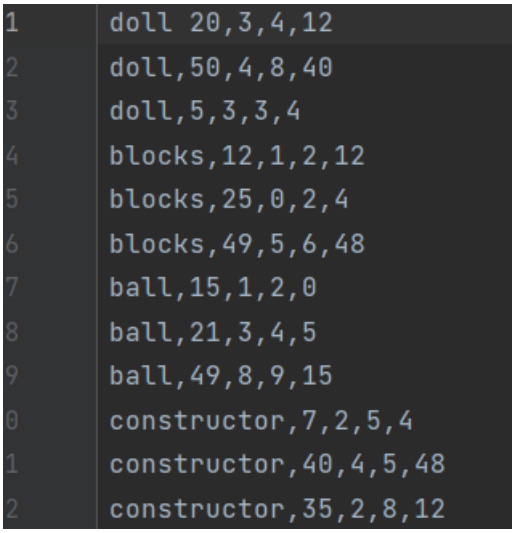


Рис. 9 Скріншот виконання завдання (3.1 частина)

```
Exception in thread "main" java.lang.IllegalArgumentException Create breakpoint : Invalid name toy: doll 20
    at Main.readToysFromFile(Main.java:60)
    at Main.main(Main.java:13)
```

Рис. 10 Скріншот виконання завдання (3.2 частина)

```
1   doll,doll,3,4,12
2   doll,50,4,8,40
3   doll,5,3,3,4
4   blocks,12,1,2,12
5   blocks,25,0,2,4
6   blocks,49,5,6,48
7   ball,15,1,2,0
8   ball,21,3,4,5
9   ball,49,8,9,15
10  constructor,7,2,5,4
11  constructor,40,4,5,48
12  constructor,35,2,8,12
```

Рис.11 Скріншот виконання завдання (4.1 частина)

```
Exception in thread "main" java.lang.NumberFormatException Create breakpoint : For input string: "doll"
    at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:67)
    at java.base/java.lang.Integer.parseInt(Integer.java:668)
    at java.base/java.lang.Integer.parseInt(Integer.java:784)
    at Main.readToysFromFile(Main.java:40)
    at Main.main(Main.java:13)
```

Рис. 12 Скріншот виконання завдання (4.2 частина)

## Висновки

Вивчив основи застосування класів пакета java.io для організації введення-виведення даних в додатках на мові Java; освоїв принципи роботи з текстовими і бінарними файлами; вивчив та освоїв механізми серіалізації-десеріалізації об'єктів класів; набув практичних навичок розробки програм на мові Java у середовищі IntelliJ IDEA, в яких передбачено роботу з потоками та файлами; набув практичних навичок застосовування колекцій; залучився до самостійної діяльності та прийняття рішень при формуванні програмного коду.

**Використані джерела**


1. Лекції з “Технології програмування”в
2. https://www.youtube.com/watch?v=ZspkReG8L2E