### ЗВІТ

До лабораторної роботи № 1

**З дисципліни:** *“Кросплатформене програмування”*

**На тему:**”*Вивчення мови Java*”

**Лектор:**

доц. каф. ПЗ

Мельник Н.Б.

**Виконав:**

ст. гр. ПЗ-32

Сомін Д. С.

**Прийняв:**

доц. каф. ПЗ

Шкраб Р. Р.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2020 р.

∑= \_\_\_\_\_ .

Львів – 2020

Мета роботи : Вивчити методи та інструменти ООП у Java та UML.

Тема роботи : Створення програми по сортуванню товарів.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Класи – це конструкції спеціального виду, які дозволяють об'єднати ряд змінних різних типів в одне ціле. Крім власне даних, класи зазвичай включають підпрограми (в термінології java - методи) і можуть включати блоки (сукупність інструкцій між фігурними дужками {}) та інші класи (внутрішні класи).

ЗАВДАННЯ

Магазин світла. Реалізувати ієрархію товарів, присутніх в магазині. Реалізувати пошук товарів певної категорії.

Реалізувати можливість сортування знайденого товару за двома типами параметрів (на вибір, реалізовано як два окремі методи).

Реалізація сортування має передбачати можливість сортувати як за спаданням, так і за зростанням.

ХІД РОБОТИ

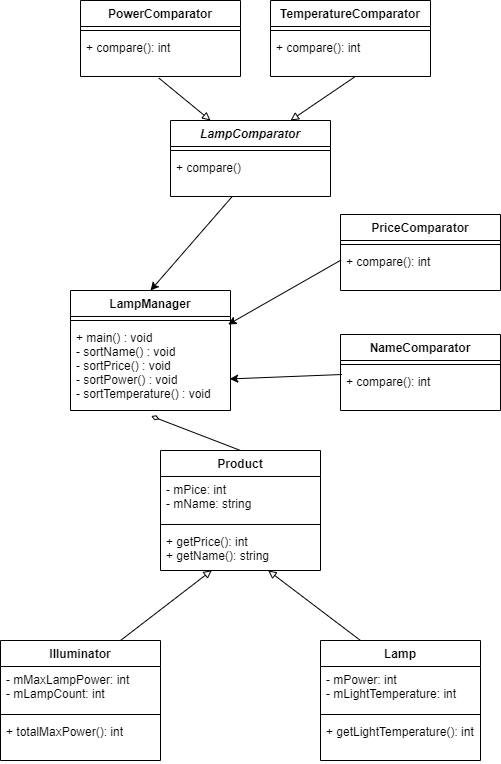


Рис. 1. Діаграма класів

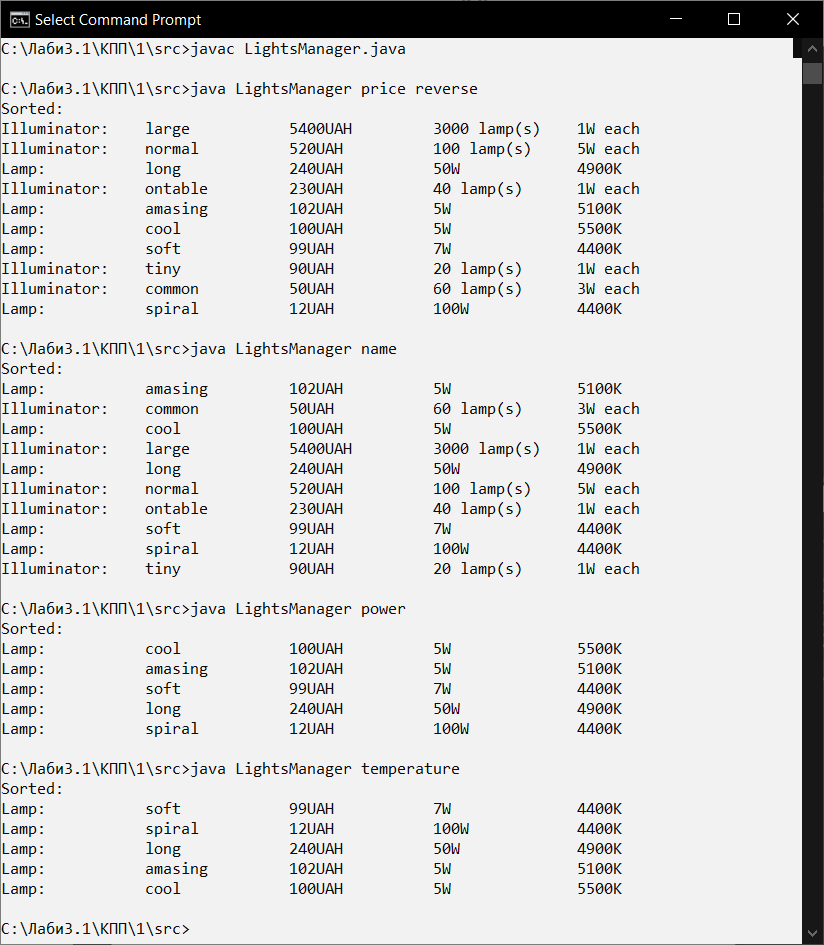


Рис. 2. Вивід програми

Код програми

Клас LightsManager:

public class LightsManager {

public static void main(String[] args)

{

final Lamp lamps[] =

{

new Lamp(100, "cool", 5, 5500),

new Lamp(102, "amasing", 5, 5100),

new Lamp(99, "soft", 7, 4400),

new Lamp(12, "spiral", 100, 4400),

new Lamp(240, "long", 50, 4900)

};

final Illuminator illuminators[] =

{

new Illuminator(5400, "large", 3000, 1),

new Illuminator(520, "normal", 100, 5),

new Illuminator(50, "common", 60, 3),

new Illuminator(230, "ontable", 40, 1),

new Illuminator(90, "tiny", 20, 1),

};

Product sortArr[] = new Product[lamps.length + illuminators.length];

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

{

sortArr[i] = new Lamp(lamps[i].mPrice,

lamps[i].mName,

lamps[i].mPower,

lamps[i].mLightTemperature);

}

for (int i = lamps.length; i < sortArr.length; ++i)

{

sortArr[i] = new Illuminator(illuminators[i - lamps.length].mPrice,

illuminators[i - lamps.length].mName,

illuminators[i - lamps.length].mLampCount,

illuminators[i - lamps.length].mMaxLampPower);

}

if (args.length < 1)

{

System.out.println("ERROR");

}

else if (args[0].equals("price"))

{

sortPrice(sortArr, (args.length > 1) && args[1].equals("reverse"));

System.out.println("Sorted:");

for (Product product : sortArr)

{

product.print();

}

}

else if (args[0].equals("name"))

{

sortName(sortArr);

System.out.println("Sorted:");

for (Product product : sortArr)

{

product.print();

}

}

else if (args[0].equals("temperature"))

{

sortTemperature(lamps);

System.out.println("Sorted:");

for (Lamp lamp : lamps)

{

lamp.print();

}

}

else if (args[0].equals("power"))

{

sortPower(lamps);

System.out.println("Sorted:");

for (Lamp lamp : lamps)

{

lamp.print();

}

}

else

{

System.out.println("ERROR");

}

}

private static void sortPrice(Product[] arr, boolean isReverse)

{

int reverseMultiplier = isReverse ? -1 : 1;

for (int i = 0; i < arr.length - 1; ++i)

{

for (int j = 0; j < arr.length - i - 1; ++j)

{

if (PriceComparator.compare(arr[j], arr[j + 1]) \* reverseMultiplier > 0)

{

Product temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

private static void sortName(Product[] arr)

{

LightsManager manager = new LightsManager();

NameComparator cmp = manager.new NameComparator();

for (int i = 0; i < arr.length - 1; ++i)

{

for (int j = 0; j < arr.length - i - 1; ++j)

{

if (cmp.compare(arr[j], arr[j + 1]) > 0)

{

Product temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

private static void sortTemperature(Lamp[] arr)

{

for (int i = 0; i < arr.length - 1; ++i)

{

for (int j = 0; j < arr.length - i - 1; ++j)

{

if (LightTemperatureComparator.compare(arr[j], arr[j + 1]) > 0)

{

Lamp temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

private static void sortPower(Lamp[] arr)

{

for (int i = 0; i < arr.length - 1; ++i)

{

for (int j = 0; j < arr.length - i - 1; ++j)

{

if (lampPowerComparator.compare(arr[j], arr[j + 1]) > 0)

{

Lamp temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

private interface LampComparator

{

public int compare(final Lamp l1, final Lamp l2);

}

private static class PriceComparator

{

public static int compare(final Product p1, final Product p2)

{

return p1.getPrice() - p2.getPrice();

}

}

private class NameComparator

{

public int compare(final Product p1, final Product p2)

{

return p1.getName().compareTo(p2.getName());

}

}

private static LampComparator LightTemperatureComparator = new LampComparator()

{

@Override

public int compare(final Lamp l1, final Lamp l2)

{

return l1.getLightTemperature() - l2.getLightTemperature();

}

};

private static LampComparator lampPowerComparator =

(final Lamp l1, final Lamp l2) ->

{

return l1.getPower() - l2.getPower();

};

}

Клас Product:

public abstract class Product

{

public Product(int price, String name)

{

mPrice = price;

mName = name;

}

public int getPrice() { return mPrice; }

public String getName() { return mName; }

public abstract void print();

protected int mPrice;

protected String mName;

}

Клас Lamp:

public class Lamp extends Product

{

public Lamp(int price, String name, int power, int lightTemperature)

{

super(price, name);

mPower = power;

mLightTemperature = lightTemperature;

}

public int getPower() { return mPower; }

public int getLightTemperature() { return mLightTemperature; }

@Override

public void print()

{

System.out.print("Lamp:\t\t" + mName + "\t\t"

+ mPrice + "UAH\t\t"

+ mPower + "W\t\t"

+ mLightTemperature + "K\n");

}

protected int mPower;

protected int mLightTemperature;

}

Клас Illuminator:

public class Illuminator extends Product

{

public Illuminator(

int price,

String name,

int maxLampPower,

int lampCount)

{

super(price, name);

mMaxLampPower = maxLampPower;

mLampCount = lampCount;

}

public int getLampCount() { return mLampCount; }

public int getMaxLampPower() { return mMaxLampPower; }

public int totalPower() { return mLampCount \* mMaxLampPower; }

@Override

public void print()

{

System.out.print("Illuminator:\t" + mName + "\t\t"

+ mPrice + "UAH\t\t"

+ mLampCount + " lamp(s)\t"

+ mMaxLampPower + "W each\n");

}

protected int mMaxLampPower;

protected int mLampCount;

}

ВИСНОВКИ

На лабораторній роботі я познайомився з Java та застосував знання UML на практиці. Я дізнався про класи в Java, статичні класи, вкладені класи та анонімні класи. Також застосував лямбда вираз. Для реалізації програми мені знадобилось встановити Java та інструмента розробника для мови Java, та навчитись компілювати та запускати програму.