Міністерство освіти та науки України

Національний університет «Львівська політехніка»

Кафедра «Електронних обчислювальних машин»



**ЗВІТ**

**з лабораторної роботи № 7**

**з дисципліни:** «Кросплатформенні засоби програмування»

**на тему:** «ПАРАМЕТРИЗОВАНЕ ПРОГРАМУВАННЯ»

Виконав: ст. гр. КІ-35

Гоцій М.М.

Прийняв:

доцент кафедри ЕОМ

Іванов Ю.С

Львів-2022

**Мета роботи:** оволодіти навиками параметризованого програмування мовою Java.

**ЗАВДАННЯ**

1. Створити параметризований клас, що реалізує предметну область задану варіантом. Клас має містити мінімум 4 методи опрацювання даних включаючи розміщення та виймання елементів. Парні варіанти реалізують пошук мінімального елементу, непарні – максимального. Написати на мові Java та налагодити програму-драйвер для розробленого класу, яка мстить мінімум 2 різні класи екземпляри яких розмішуються у 9 екземплярі розробленого класу-контейнеру. Програма має розміщуватися в пакеті Група.Прізвище.Lab6 та володіти коментарями, які дозволять автоматично згенерувати документацію до розробленого пакету.

2. Автоматично згенерувати документацію до розробленого пакету.

3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації.

4. Дати відповідь на контрольні запитання.

**Варіант 11:**

****

**Текст програми:**

|  |
| --- |
| ***Лістинг Bakery***  package KI35.Hotsii.Lab7;  */\*\*  \* Class <code>Bakery</code> describes Bakery  \*/* public class Bakery implements Shop {  private String name;  private int productPrice;   */\*\*  \* Constructor  \** ***@param*** *name - name value  \** ***@param*** *price - price value  \*/* public Bakery(String name, int price) {  this.name = name;  this.productPrice = price;  }   */\*\*  \* This method return name  \** ***@return*** *name  \*/* public String getName() {  return name;  }   */\*\*  \* This method sets name  \** ***@param*** *name - name value  \*/* public void setName(String name) {  this.name = name;  }   */\*\*  \* This method sets price  \** ***@param*** *price - name price  \*/* public void setProductPrice(int price) {  this.productPrice = price;  }   @Override  public void print() {  System.*out*.printf("Name: %s; Product price: %s;%n", name, productPrice);  }   @Override  public int getPrice() {  return productPrice;  }   @Override  public int compareTo(Shop shop) {  return productPrice - shop.getPrice();  } }  ***Лістинг ClothingShop***  */\*\** package KI35.Hotsii.Lab7;  */\*\*  \* Class <code>ClothingShop</code> describes clothing shop  \*/* public class ClothingShop implements Shop {  private String name;  private int shoesPrice;   */\*\*  \* Constructor  \** ***@param*** *name - name value  \** ***@param*** *price - price value  \*/* public ClothingShop(String name, int price) {  this.name = name;  this.shoesPrice = price;  }   */\*\*  \* This method return name  \** ***@return*** *name  \*/* public String getName() {  return name;  }   */\*\*  \* This method sets name  \** ***@param*** *name - name value  \*/* public void setName(String name) {  this.name = name;  }   */\*\*  \* This method sets price  \** ***@param*** *price - name price  \*/* public void setShoesPrice(int price) {  this.shoesPrice = price;  }   @Override  public void print() {  System.out.printf("Name: %s; Shoes price: %s;%n", name, shoesPrice);  }   @Override  public int getPrice() {  return shoesPrice;  }   @Override  public int compareTo(Shop shop) {  return shoesPrice - shop.getPrice();  } }  ***Лістинг Mall***  package KI35.Hotsii.Lab7;  import java.util.ArrayList; import java.util.List;  */\*\*  \* Class <code>Mall</code> describes mall  \** ***@param*** <*T*>  *\*/* public class Mall<T extends Shop> {  private final List<T> shops = new ArrayList<>();   */\*\*  \*  \** ***@param*** *shop - shop  \*/* public void addShop(T shop) {  shops.add(shop);  }   */\*\*  \*  \** ***@param*** *index - index  \*/* public void deleteShop(int index) {  shops.remove(index);  }   */\*\*  \* This method returns shop with the highest price  \** ***@return*** *<code>Shop</code>shop with the highest price  \*/* public T findMax() {  if (shops.isEmpty()) return null;   T shopWithHighestPrice = shops.get(0);  for (T shop: shops) {  if (shopWithHighestPrice.compareTo(shop) < 0) {  shopWithHighestPrice = shop;  }  }   return shopWithHighestPrice;  } }  ***Лістинг Shop***  package lab7; package KI35.Hotsii.Lab7;  */\*\*  \* Shop interface  \*/* public interface Shop extends Comparable<Shop>{  */\*\*  \* Method prints object's data  \*/* void print();  */\*\*  \*  \* Method returns the price of the product  \*/* int getPrice(); }  ***Лістинг Main***  package lab7; import KI35.Hotsii.Lab7.Bakery; import KI35.Hotsii.Lab7.ClothingShop; import KI35.Hotsii.Lab7.Mall; import KI35.Hotsii.Lab7.Shop;  */\*\*  \* Class <code>Accountant</code> Implements driver for Accountant class  \** ***@author*** *Mykyta Hotsii  \** ***@version*** *1.0  \*/* public class Main {  */\*\*  \** ***@param*** *args  \*/* public static void main(String[] args) {  Mall<Shop> hdd = new Mall<>();  hdd.addShop(new ClothingShop("Windows", 852124325));  hdd.addShop(new ClothingShop("Weekend" , 32568741));  hdd.addShop(new Bakery("People" , 2547814));  hdd.addShop(new Bakery("JRE" , 67894257));  Shop res = hdd.findMax();  System.*out*.print("The greatest data on HDD is: \n");  res.print();  } } |

**Результат роботи :**

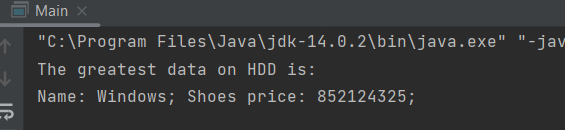
****

Рис.1 Результат роботи програми

**Відповіді на контрольні запитання:**

1. Дайте визначення терміну «параметризоване програмування».

Параметризоване програмування є аналогом шаблонів у С++. Воно полягає у написанні коду, що можна багаторазово застосовувати з об’єктами різних класів. Користувачів параметризованого програмування можна поділити на 3 рівні кваліфікації:

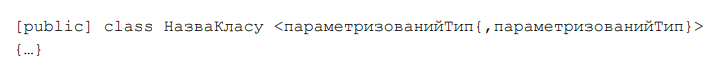
1. ті, що користуються готовими класами;

2. ті, що користуються готовими класами і вміють виправляти помилки, що виникають при їх використанні;

3. ті, що пишуть власні параметризовані класи. Для успішного застосування параметризованого програмування слід навчитися розуміти помилки, що генерує середовище при компіляції програми, що можуть стосуватися, наприклад, неоднозначності визначення спільного суперкласу для всіх переданих об’єктів. З іншої сторони необхідно передбачити захист від присвоєння об’єктів параметризованого класу, що містять об’єкти підкласу об’єктам параметризованого класу, що містять об’єкти суперкласу і дозволити зворотні присвоєння. Для вирішення цієї проблеми у мові Java введено так звані підстановочні типи. Це далеко не всі «підводні камені», що виникають при застосуванні параметризованого програмування.

1. Розкрийте синтаксис визначення простого параметризованого класу.

Параметризований клас – це клас з однією або більше змінними типу. Синтаксис оголошення параметризованого класу:



**Висновок:**

Під час виконання роботи я оволодів навиками параметризованого програмування мовою Java.