**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

# по дисциплине «Введение в разработку программного обеспечения»

на тему: «Развитие представлений о разработке программ. Объектно-ориентированное программирование в Java»

Выполнил: студент гр. ИТП-21

Бурковский И.Д.

Принял: ассистент

Башаримов Ю.С.

Гомель 2024

Цель работы: Разработать UML-диаграмму иерархии классов, согласно варианта (таблица 1). При описании иерархии использовать наследование и композицию. На основе UML-диаграммы разработать иерархию классов на языке Java.

**Ход работы**

**Задание.** Создать иерархию классов для учёта покупок хот-догов (HunterDog, MasterDog и Berlinka); При этом все компоненты могут продаваться отдельно. Продать каждый вид хот-дога. Подсчитать общую сумму всех заказов. Подсчитать количество заказов. Подсчитать среднюю стоимость заказов.

Код программам находится в приложении А.

Результат выполнения задания представлен на рисунке 1.

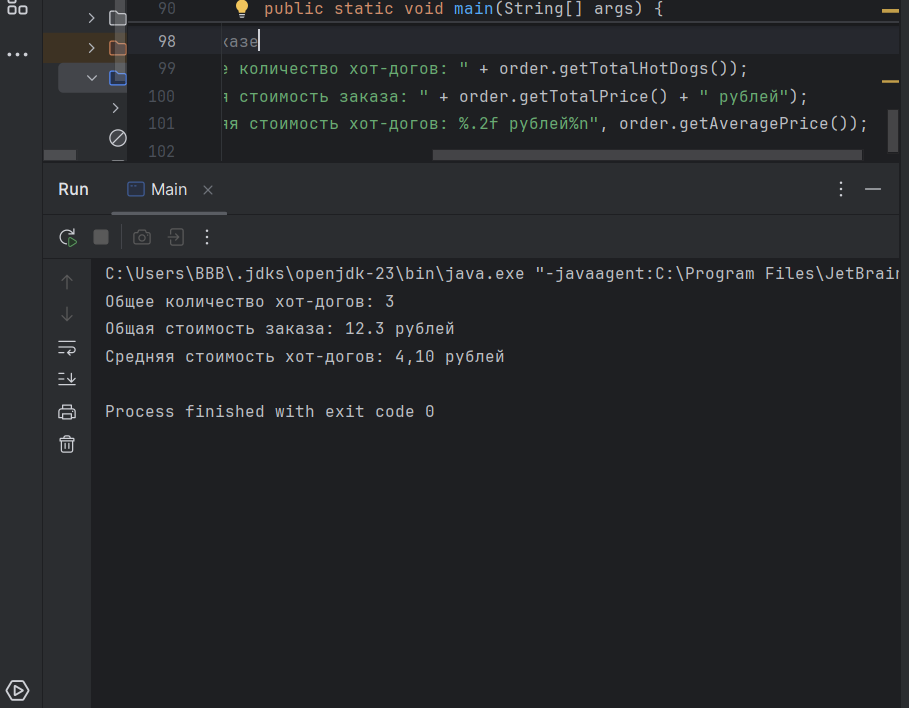


Рисунок 1 – Результат выполнения программы

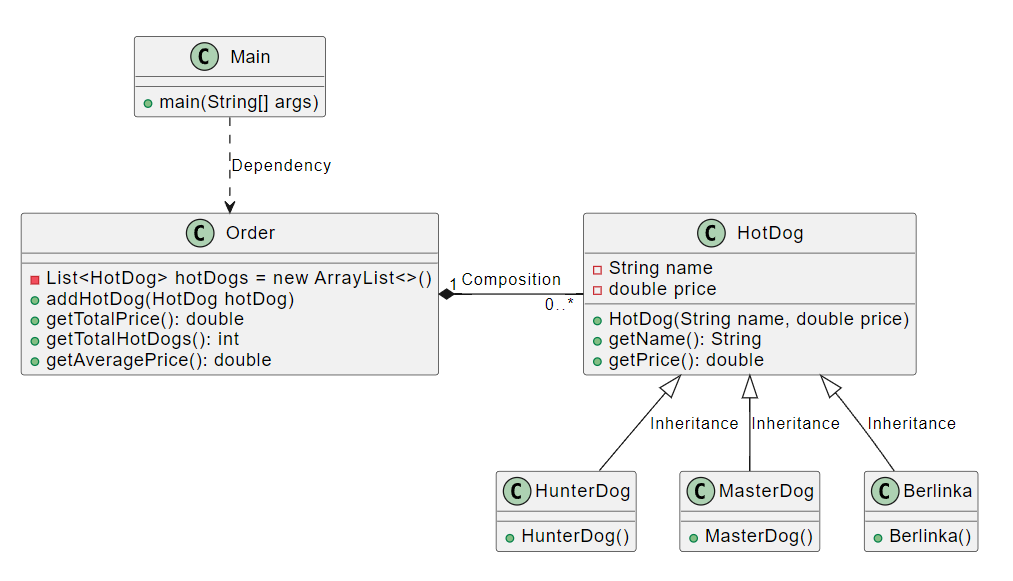


Рисунок 2 – UML-диаграммa

Вывод: На основе UML-диаграммы создана иерархия классов на языке Java, где различные элементы наследуются от общего базового класса, а класс заказа использует композицию для хранения объектов этих элементов. Такой подход обеспечил структурированное и гибкое проектирование системы.

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Текст программы**

**Main.java:**

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Представляет базовый хот-дог с названием и ценой.

\*/

class HotDog {

private String name;

private double price;

public HotDog(String name, double price) {

this.name = name;

this.price = price;

}

public String getName() {

return name;

}

public double getPrice() {

return price;

}

}

/\*\*

\* Класс, представляющий хот-дог типа HunterDog.

\*/

class HunterDog extends HotDog {

public HunterDog() {

super("HunterDog", 3.50);

}

}

/\*\*

\* Класс, представляющий хот-дог типа MasterDog.

\*/

class MasterDog extends HotDog {

public MasterDog() {

super("MasterDog", 4.00);

}

}

/\*\*

\* Класс, представляющий хот-дог типа Berlinka.

\*/

class Berlinka extends HotDog {

public Berlinka() {

super("Berlinka", 4.80);

}

}

/\*\*

\* Класс для управления заказами хот-догов, расчета общей и средней стоимости.

\*/

class Order {

private List<HotDog> hotDogs = new ArrayList<>();

// Метод для добавления хот-дога в заказ

public void addHotDog(HotDog hotDog) {

hotDogs.add(hotDog);

}

// Метод для расчета общей стоимости заказа

public double getTotalPrice() {

double total = 0;

for (HotDog hotDog : hotDogs) {

total += hotDog.getPrice();

}

return total;

}

// Метод для получения общего количества хот-догов в заказе

public int getTotalHotDogs() {

return hotDogs.size();

}

// Метод для расчета средней стоимости хот-догов в заказе

public double getAveragePrice() {

if (hotDogs.isEmpty()) {

return 0;

}

return getTotalPrice() / hotDogs.size();

}

}

/\*\*

\* Главный класс для демонстрации работы системы заказов хот-догов.

\*/

public class Main {

public static void main(String[] args) {

Order order = new Order();

// Добавление хот-догов в заказ

order.addHotDog(new HunterDog());

order.addHotDog(new MasterDog());

order.addHotDog(new Berlinka());

// Вывод информации о заказе

System.out.println("Общее количество хот-догов: " + order.getTotalHotDogs());

System.out.println("Общая стоимость заказа: " + order.getTotalPrice() + " рублей");

System.out.printf("Средняя стоимость хот-догов: %.2f рублей%n", order.getAveragePrice());

}

}