Министерство образования

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Современные языки программирования

Отчет по лабораторной работе №1

«Использование языка программирования Kotlin»

Выполнил: Рубцова Е. С.

Студент группы 310902

Проверил: Усенко. Ф. В.

Минск 2024

Цель:выполнить разработку приложения с использованием языка программирования Kotlin.

Задание. 25. - согласно варианта задания написать программу (Вариант = номер в списке группы % 30 + 1);

- объекты и их взаимоотношения, имеющиеся в варианте задания, должны быть реализованы;

- функциональная часть приложения, представленная диаграммой последовательности, должна быть реализована;

- разбить функционал приложения на несколько пакетов придерживаясь логики.

Добавить класс Пакет и организовать возможность помещать туда товар. Добавить классу Тележка метод Выдать товар(). Добавить класс Молоко.

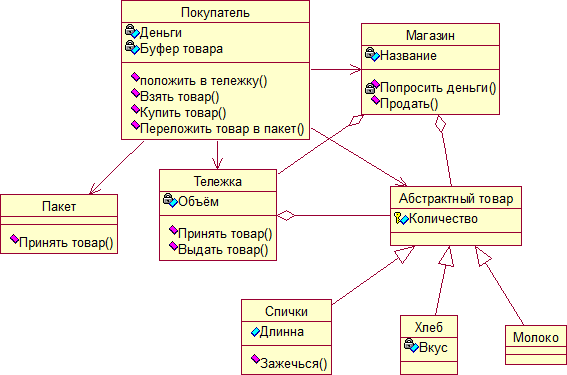


Рисунок 50 – Диаграмма классов

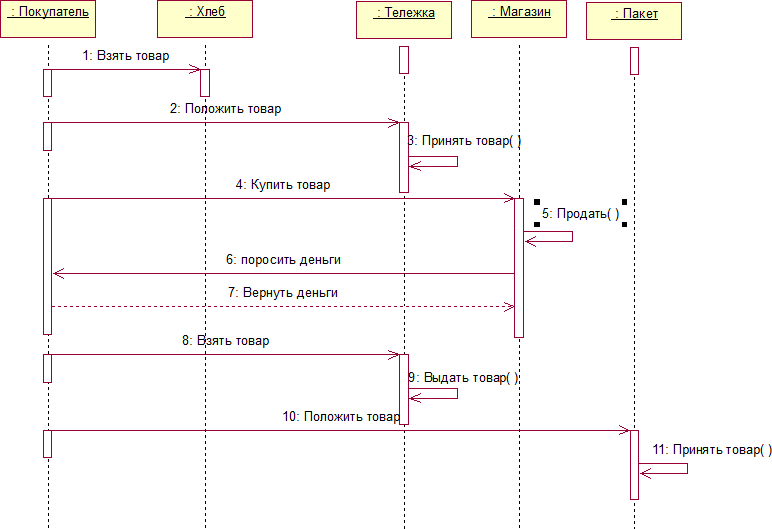


Рисунок 51 – Диаграмма последовательности

import java.util.Scanner  
abstract class Product {  
 abstract val productName: String  
 protected abstract val name: String  
 protected abstract val quantity: Int  
}  
class Bread(override val name: String = "Хлеб", override val quantity: Int = 1) : Product() {  
 private val taste: String = "Пшеничный"  
 override val productName: String get() = name  
 val productQuantity: Int get() = quantity  
}  
class Milk(override val name: String = "Молоко", override val quantity: Int = 1) : Product() {  
 override val productName: String get() = name  
 val productQuantity: Int get() = quantity  
}  
class Bag {  
 private val products = mutableListOf<Product>()  
 fun receiveProduct(product: Product) {  
 products.add(product)  
 println("Принят товар: ${product.productName}")  
 }  
 fun showProducts() {  
 println("Содержимое пакета:")  
 products.forEach { println(it.productName) }  
 }  
}  
class Cart {  
 private val products = mutableListOf<Product>()  
 fun receiveProduct(product: Product) {  
 products.add(product)  
 println("Товар добавлен в тележку: ${product.productName}")  
 }  
 fun issueProduct(): Product? {  
 return if (products.isNotEmpty()) {  
 val product = products.removeAt(products.size - 1)  
 println("Товар выдан: ${product.productName}")  
 product  
 } else {  
 println("Тележка пуста!")  
 null  
 }  
 }  
 fun showContents() {  
 println("Содержимое тележки:")  
 if (products.isEmpty()) {  
 println("Тележка пуста.")  
 } else {  
 products.forEach { println(it.productName) }  
 }  
 }  
 fun getProductCount(): Int {  
 return products.size  
 }  
}  
class Customer(var money: Int) {  
 private val cart = Cart()  
 private val bag = Bag()  
 fun takeProduct(product: Product) {  
 println("Покупатель взял товар: ${product.productName}")  
 putInCart(product)  
 }  
 private fun putInCart(product: Product) {  
 cart.receiveProduct(product)  
 }  
 fun transferToBag() {  
 val count = cart.getProductCount()  
 if (count == 0) {  
 println("Тележка пуста. Нет товаров для переложения.")  
 return  
 }  
 println("Перекладываем товары из тележки в пакет.")  
 for (i in 0 until count) {  
 cart.issueProduct()?.let { bag.receiveProduct(it) }  
 }  
 println("Все товары из тележки переложены в пакет.")  
 }  
 fun buyProduct(store: Store): Boolean {  
 println("Покупатель хочет купить товары.")  
 val count = cart.getProductCount()  
 if (count == 0) {  
 println("Тележка пуста. Нет товаров для покупки.")  
 return false  
 }  
 val totalCost = 100 \* count  
 return if (store.requestMoney(this, totalCost)) {  
 println("Покупка успешна.")  
 store.sellProduct()  
 } else {  
 println("Недостаточно денег для покупки.")  
 false  
 }  
 }  
 fun showBagContents() {  
 bag.showProducts()  
 }  
 fun showCartContents() {  
 cart.showContents()  
 }  
}  
class Store(val name: String) {  
 private var productAvailable = true  
 fun sellProduct(): Boolean {  
 return if (productAvailable) {  
 println("Магазин: Товар продан.")  
 true  
 } else {  
 println("Магазин: Товар отсутствует.")  
 false  
 }  
 }  
 private fun askForMoney(customer: Customer, totalCost: Int): Boolean {  
 if (customer.money >= totalCost) {  
 customer.money -= totalCost  
 println("Магазин: Получено $totalCost от покупателя.")  
 return true  
 } else {  
 println("Магазин: Недостаточно денег у покупателя.")  
 return false  
 }  
 }  
 fun requestMoney(customer: Customer, totalCost: Int): Boolean {  
 return askForMoney(customer, totalCost)  
 }  
}  
fun main() {  
 val store = Store("Магазин 1")  
 val bread = Bread()  
 val milk = Milk()  
 val customer = Customer(500)  
 while (true) {  
 println("\nВыберите действие:")  
 println("1. Выбрать товар")  
 println("2. Посмотреть содержимое тележки")  
 println("3. Купить товары из тележки")  
 println("4. Переложить товары из тележки в пакет")  
 println("5. Выйти")  
 var choose :Int = readLine()!!.toInt()  
 when (choose) {  
 1 -> {  
 println("Выберите товар (1 - Хлеб, 2 - Молоко):")  
 var x :Int = readLine()!!.toInt()  
 when (x) {  
 1 -> customer.takeProduct(bread)  
 2 -> customer.takeProduct(milk)  
 else -> println("Некорректный выбор товара.")  
 }  
 }  
 2 -> {  
 customer.showCartContents()  
 }  
 3 -> {  
 if (customer.buyProduct(store)) {  
 println("Покупка успешна.")  
 }  
 }  
 4 -> {  
 customer.transferToBag()  
 }  
 5 -> {  
 println("Выход из программы.")  
 return  
 }  
 else -> println("Некорректный выбор действия.")  
 }  
 }  
}

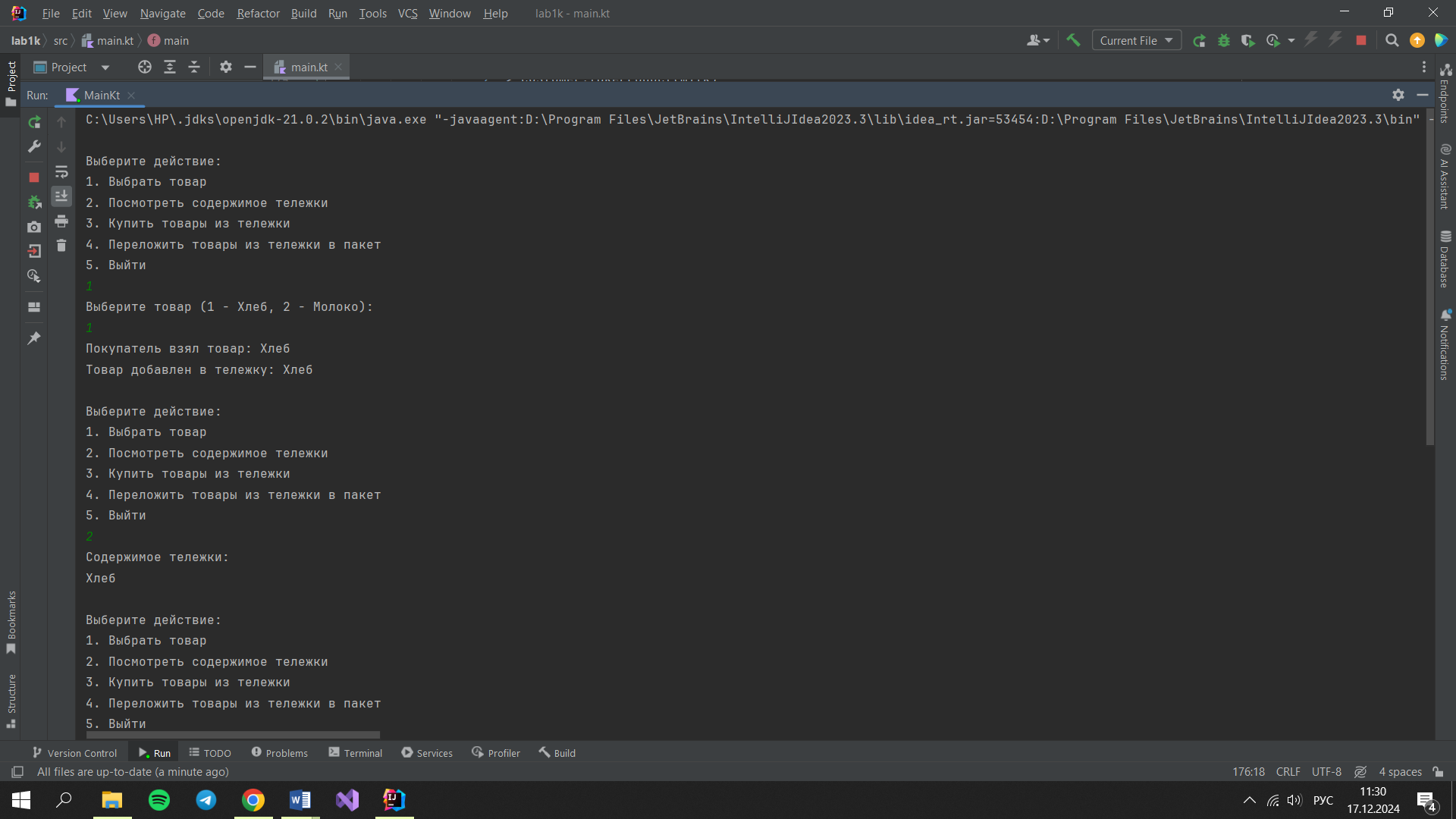


Рисунок 1 – Результат выполнения программы

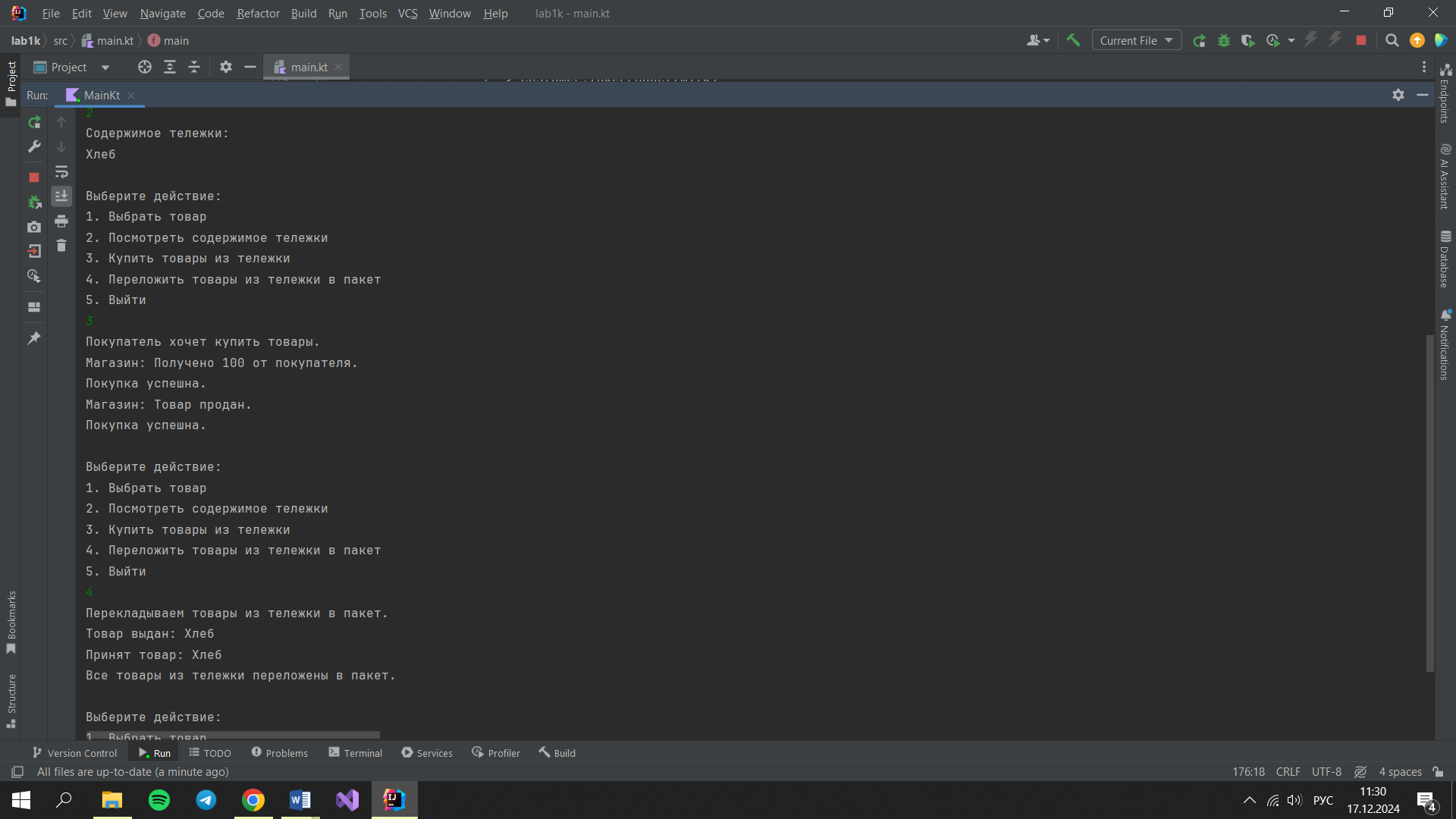


Рисунок 2 – Результат выполнения программы