Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №7**

**Дисциплина: Программирование для мобильных платформ**

Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.Д. Воробьев

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Т.А. Приходько

**Тема работы:** Разработка консольного приложения по теме синхронизации

**Постановка задачи:**

В порт грузовиками доставляются грузы одинаковыми по грузоподъемности грузовиками – 10т, которые сгружаются на 5 платформ. К платформам причаливают грузовые суда разного тоннажа  (100, 120, 150т), загружаются до тех пор, пока не наберут груза по предельной возможности, затем освобождают место другому судну для загрузки.

**Описание решения:**

1. Класс Ship: Этот класс представляет корабль, расширяет класс Thread и синхронно обращается к разделяемому ресурсу, чтобы избежать гонки между потоками. В рамках программы создаётся несколько инстансов этого класса.
2. Класс Truck: Этот класс представляет грузовика, расширяет класс Thread и синхронно обращается к разделяемому ресурсу, чтобы избежать гонки между потоками. В рамках программы создаётся несколько инстансов этого класса.
3. Класс Platform: Этот класс представляет платформу, к которой будет осуществляться синхронный доступ.
4. Класс Main: В этом классе конфигурируется количество грузовиков и кораблей. Затем производится запуск потоков.

На рисунке 1 представлена UML-диаграмма классов.

Изображение выглядит как текст, снимок экрана, диаграмма

Автоматически созданное описание

Рисунок 1 – UML-диаграмма

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2 – Консольный вывод программы.

**Листинг программ**

import classes.Platform;  
import classes.Truck;  
import classes.Ship;  
  
import java.util.concurrent.CompletableFuture;  
  
public class Main {  
 public static void main(String[] args) {  
  
 Truck.*delay* = 2000;  
 Ship.*delay* = 2000;  
  
 Platform[] platforms = new Platform[5];  
 platforms[0] = new Platform();  
 platforms[1] = new Platform();  
 platforms[2] = new Platform();  
 platforms[3] = new Platform();  
 platforms[4] = new Platform();  
  
 Truck[] trucks = new Truck[3];  
 trucks[0] = new Truck(platforms, "Грузовик 1");  
 trucks[1] = new Truck(platforms, "Грузовик 2");  
 trucks[2] = new Truck(platforms, "Грузовик 3");  
  
 for (Truck truck : trucks) {  
 truck.start();  
 }  
  
  
 Ship[] ships = new Ship[3];  
 ships[0] = new Ship(100, "Корабль 1", platforms);  
 ships[1] = new Ship(120, "Корабль 2", platforms);  
 ships[2] = new Ship(150, "Корабль 3", platforms);  
  
  
 for (Ship ship : ships) {  
 ship.start();  
 }  
  
 }  
}

package classes;  
  
import java.util.Random;  
import java.util.concurrent.CompletableFuture;  
  
  
public class Truck extends Thread {  
 private int capacity;  
 private Platform[] platforms;  
 static public int *delay* = 500;  
 private String name;  
  
 public Truck(Platform[] platforms, String name) {  
 this.platforms = platforms;  
 this.capacity = 10;  
 this.name = name;  
 }  
  
 private int getRandomNumber(int min, int max) {  
 return (int) ((Math.*random*() \* (max - min)) + min);  
 }  
  
 @Override  
 public void run() {  
 super.run();  
 while (true) {  
 int index = getRandomNumber(0, platforms.length);  
 synchronized (platforms[index]) {  
 if (platforms[index].sutedByTruck) {  
 continue;  
 }  
 }  
 try {  
 Thread.*sleep*(*delay*);  
 } catch (InterruptedException e) {  
 throw new RuntimeException(e);  
 }  
 synchronized (platforms[index]) {  
 platforms[index].sutedByTruck = true;  
 platforms[index].currentLoad += this.capacity;  
 System.*out*.println("Грузовик " + this.name + " отгрузил " + this.capacity + " на платформу " + (index + 1));  
 System.*out*.println("На платформе " + (index + 1) + " уже " + this.platforms[index].currentLoad + " тонн");  
 platforms[index].sutedByTruck = false;  
 }  
 }  
 }  
}

package classes;  
  
public class Ship extends Thread {  
 private int tonnage;  
 private int currentTonnage;  
 private final Platform[] platforms;  
 private String name;  
 static public int *delay* = 2000;  
  
  
 public Ship(int tonnage, String name, Platform[] platforms) {  
 this.platforms = platforms;  
 this.tonnage = tonnage;  
 this.name = name;  
 }  
  
 private int getRandomNumber(int min, int max) {  
 return (int) ((Math.*random*() \* (max - min)) + min);  
 }  
  
 @Override  
 public void run() {  
 super.run();  
 while (true) {  
 int index = getRandomNumber(0, platforms.length);  
 synchronized (platforms[index]) {  
 if (platforms[index].sutedByShip) {  
 continue;  
 }  
 }  
 while (this.currentTonnage < this.tonnage) {  
 synchronized (platforms[index]) {  
 platforms[index].sutedByShip = true;  
 if (platforms[index].currentLoad == 0) {  
 try {  
 Thread.*sleep*(*delay*);  
 } catch (InterruptedException e) {  
 throw new RuntimeException(e);  
 }  
 } else {  
 if (platforms[index].currentLoad - (tonnage - currentTonnage) > 0) {  
 platforms[index].currentLoad -= tonnage - currentTonnage;  
 currentTonnage += tonnage - currentTonnage;  
 } else {  
 currentTonnage += platforms[index].currentLoad;  
 platforms[index].currentLoad = 0;  
 }  
 System.*out*.println("Тоннаж коробля " + this.name + ": " + this.currentTonnage);  
 }  
 }  
 }  
 synchronized (platforms[index]) {  
 platforms[index].sutedByShip = false;  
 System.*out*.println("Корабль " + this.name + " загружен");  
 this.currentTonnage = 0;  
 }  
 }  
 }  
}

package classes;  
  
public class Platform {  
  
 public int currentLoad;  
 public boolean sutedByShip;  
 public boolean sutedByTruck;  
  
 public Platform() {  
 currentLoad = 0;  
 sutedByShip = false;  
 sutedByTruck = false;  
 }  
}