Министерство образования Калининградской области

государственное бюджетное учреждение Калининградской области

профессиональная образовательная организация

«Колледж информационных технологий и строительства»

(ГБУ КО ПОО «КИТиС»)

**Отчет по учебной практике**

УП.01 Разработка модулей программного обеспечения для компьютерных систем

по ПМ.01 Разработка модулей программного обеспечения для компьютерных систем

Специальность 09.02.07 «Информационные системы и программирование»

Сроки прохождения практики:

с «17» ноября 2022 г. по «07» декабря 2022 г.

Место практики ГБУ КО ПОО «КИТиС»

|  |  |
| --- | --- |
| Выполнил: | студент 4 курса,  группы ИСп 19-1  Арипджанова Малика Талатовна  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |
| Проверила: | Большакова-Стрекалова Анна Викторовна  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

Калининград, 2022

Содержание

|  |  |
| --- | --- |
| Введение | 3 |
| 1 Решение задач | 4 |
| 1.1 Задача 1 | 4 |
| 1.2 Задача 2 | 4 |
| 1.3 Задача 3 | 8 |
| 1.4 Задача 4 | 12 |
| 1.5 Задача 5 | 18 |
| 1.6 Задача 7 | 21 |
| 1.7 Задача 9 | 22 |
| 1.8 Меню с задачами | 28 |
| 1.9 Задача 10 | 30 |
| 2 Разработка АИС «Поликлиника» |  |
| 2.1 ER-диаграмма |  |
| 2.2 Техническое задание |  |
| 2.3 Разработка структурной и функциональной схемы АИС |  |
| 2.4 Проектирование и реализация интерфейса |  |
| 2.5 Разработка встроенной системы справочного руководства |  |
| 2.6 Проведение тестирования |  |
| 3 Разработка мобильного приложения |  |
|  |  |
| 4 Разработка сайта |  |
|  |  |
| 5 Руководство пользователя |  |
|  |  |
| Заключение |  |
|  |  |
| Список использованных источников |  |
|  |  |
| Приложение |  |

Введение

1. Решение задач
   1. Задача 1

Вывести числа от 1 до k в виде матрицы N x N слева направо и сверху вниз.

import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.print("N = ");  
 int N = in.nextInt();  
 int[][] matrix = new int[N][N];  
 int count = 1;  
 for (int i = 0; i < matrix.length; i++) {  
 for (int j = 0, value = i; j < matrix[i].length; j++, value += N) {  
 matrix[i][j] = count++;  
 }  
 }  
 for (int i = 0; i < matrix.length; i++) {  
 for (int j = 0; j < matrix[i].length; j++) {  
 System.*out*.print(matrix[i][j] + "\t");  
 }  
 System.*out*.println();  
 }  
 }  
}

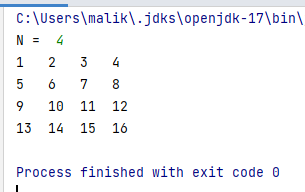


Рисунок 1 – Результат задачи 1

* 1. Задача 2

Customer: id, Фамилия, Имя, Отчество, Адрес, Номер кредитной карточки, Номер банковского счета.

Создать массив объектов. Вывести:

a) список покупателей в алфавитном порядке;

b) список покупателей, у которых номер кредитной карточки находится в заданном интервале.

import java.util.ArrayList;  
import java.util.Collections;  
import java.util.Comparator;  
import java.util.List;  
public class Main {  
 public static void main(String[] args) {  
 Shop market = new Shop("Theia");  
 market.addCustomerToShopList(new Customer("Арипджанова", "Малика", "Талатовна", 758\_265\_999, 6996));  
 market.addCustomerToShopList(new Customer("Дукмас", "Злата", "Сергеевна", 249\_134\_694, 1008));  
 market.addCustomerToShopList(new Customer("Лебедев", "Николай", "Андреевич", 100\_699\_809, 6740));  
 market.addCustomerToShopList(new Customer("Ахраров", "Алексей", "Станиславович", 368\_427\_418, 1258));  
 market.addCustomerToShopList(new Customer("Бызова", "Екатерина", "Алексеевна", 999\_025\_589, 9731));  
 System.*out*.println("\nСортировка по имени:");  
 List<Customer> listSortByName = market.getListName();  
 for (Customer c : listSortByName) {  
 System.*out*.println(c);  
 }  
 System.*out*.println("\nСортировка по фамилии:");  
 List<Customer> listSortBySurname = market.getListSurname();  
 for (Customer c : listSortBySurname) {  
 System.*out*.println(c);  
 }  
 System.*out*.println("\nСортировка по кредитной карте:");  
 List<Customer> diapasonCardumber = market.getListDiapasonCardNumber(0000000000,696969696);  
 for (Customer c : diapasonCardumber) {  
 System.*out*.println(c);  
 }  
 }  
}  
class Shop {  
 private String ShopName;  
 private ArrayList<Customer> customerList = new ArrayList<>();  
 Shop(String ShopName) {  
 this.ShopName = ShopName;  
 }  
 void addCustomerToShopList(Customer customer) {  
 customerList.add(customer);  
 }  
 List<Customer> getListName() {  
 List<Customer> list = new ArrayList<>(customerList);  
 Collections.*sort*(list, new Comparator<Customer>() {  
 @Override  
 public int compare(Customer o1, Customer o2) {  
 return o1.getName().compareToIgnoreCase(o2.getName());  
 }  
 });  
 return list;  
 }  
 List<Customer> getListSurname() {  
 List<Customer> list = new ArrayList<>(customerList);  
 Collections.*sort*(list, new Comparator<Customer>() {  
 @Override  
 public int compare(Customer o1, Customer o2) {  
 return o1.getSurname().compareToIgnoreCase(o2.getSurname());  
 }  
 });  
 return list;  
 }  
 List<Customer> getListDiapasonCardNumber(int diapasonStart, int diapasonEnd) {  
 List<Customer> list = new ArrayList<>();  
 for (Customer c : customerList) {  
 if (c.getCardNumber() >= diapasonStart && c.getCardNumber() <= diapasonEnd) {  
 list.add(c);  
 }  
 }  
 return list;  
 }  
}  
class Customer {  
 {  
 *idGenertor*++;  
 }  
 private static int *idGenertor* = 0;  
 private int id = *idGenertor*;  
 private String surname;  
 private String name;  
 private String patronymic;  
 private int CardNumber;  
 private int BankNumber;  
 Customer(String surname, String name, String patronymic, int CardNumber, int BankNumber) {  
 this.name = name;  
 this.surname = surname;  
 this.patronymic = patronymic;  
 this.CardNumber = CardNumber;  
 this.BankNumber = BankNumber;  
 }  
 public String getName() {  
 return name;  
 }  
 public String getSurname() {  
 return surname;  
 }  
 public String getPatronymic() {  
 return patronymic;  
 }  
 public int getCardNumber() {  
 return CardNumber;  
 }  
 public int getBankNumber() {  
 return BankNumber;  
 }  
 public String toString() {  
 return String.*format*("ID: %d\t Имя: %s\t Фамилия: %s\t Отчество %s \t Номер карты %d\t Номер банка %d",  
 id, surname, name, patronymic, CardNumber, BankNumber);  
 }  
}

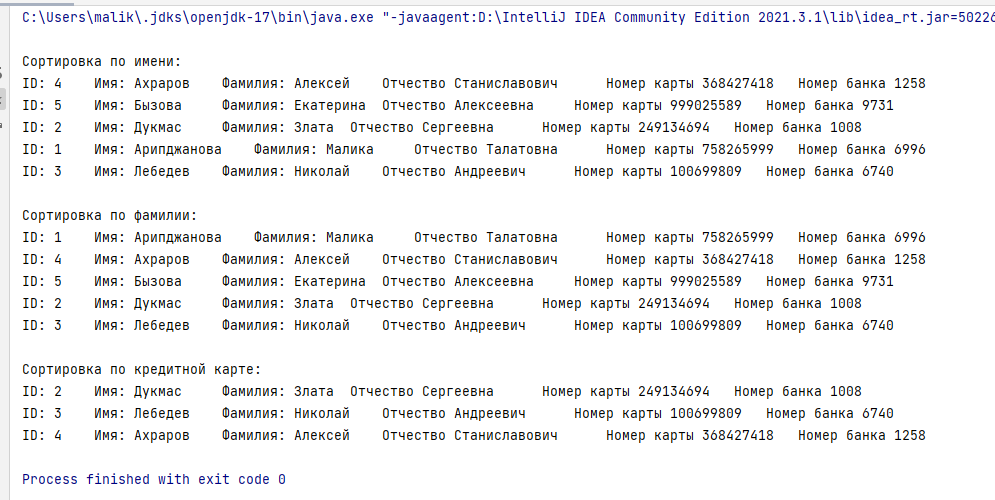


Рисунок 2 – Результат задачи 2

* 1. Задача 3

Создать объект класса Автомобиль, используя классы Колесо, Двигатель. Методы: ехать, заправляться, менять колесо, вывести на консоль марку автомобиля.

public class Main {  
 public static void main(String[] args) {  
 Car audi = new Car("Audi R8");  
 System.*out*.printf("Создан автомобиль %s. Test drive:%n", audi.getModelName());  
 audi.run();  
 Engine engine = new Engine(950, 6.485);  
 audi.setEngine(engine);  
 System.*out*.printf("%nДвигательная мощность %d л.с. Объем %.3f литров. Test drive:%n",  
 engine.getPower(), engine.getVolume());  
 audi.run();  
 int whellDiameter = 50;  
 audi.setWheels(new Wheel(whellDiameter), new Wheel(whellDiameter), new Wheel(whellDiameter), new Wheel(whellDiameter));  
 System.*out*.printf("%nУстановка колес. Test drive:\n");  
 audi.run();  
 audi.getWheels().get(2).breakWheel();  
 System.*out*.println("\nSTOP! Что-то сломалось.\nTest drive:");  
 audi.run();  
 audi.changeWheel();  
 System.*out*.println("\nTest drive:");  
 audi.run();  
 System.*out*.println("...");  
 audi.setFuel(false);  
 audi.run();  
 audi.refuel();  
 audi.run();  
 System.*out*.println("\nМодель автомобиля: " + audi.getModelName());  
 }  
}

import java.util.ArrayList;  
import java.util.Arrays;  
class Car {  
 private String modelName;  
 private Engine engine;  
 private ArrayList<Wheel> wheels;  
 private boolean isFuel;  
 Car(String modelName) {  
 this.modelName = modelName;  
 this.wheels = new ArrayList<>(10);  
 this.isFuel = true;  
 }  
 void setEngine(Engine engine) {  
 this.engine = engine;  
 }  
 ArrayList<Wheel> getWheels() {  
 return wheels;  
 }  
 void setWheels(Wheel w1, Wheel w2, Wheel w3, Wheel w4) {  
 this.wheels = new ArrayList<>(Arrays.*asList*(w1, w2, w3, w4));  
 }  
 String getModelName() {  
 return modelName;  
 }  
 void changeWheel() {  
 for(int i = 0; i < 4; i++) {  
 if (!wheels.get(i).isGood()) {  
 System.*out*.printf("№ %d изменен.\n", i + 1);  
 int wheelDia = this.getWheels().get(i).getDiameter();  
 this.getWheels().set(i, new Wheel(wheelDia));  
 }  
 }  
 }  
 void setFuel(boolean fuel) {  
 this.isFuel = fuel;  
 }  
 void refuel() {  
 this.isFuel = true;  
 System.*out*.println("Машина заправлена.");  
 }  
 void run() {  
 int wheelCounter = 0;  
 String message = "";  
 boolean isEngine = false;  
 for (Wheel wheel : wheels) {  
 if (wheel != null) {  
 wheelCounter++;  
 }  
 }  
 if (wheelCounter == 0) {  
 message = "- колеса не установлены. Пожалуйста, установите 4 колеса.\n";  
 } else if (wheelCounter == 1) {  
 message = "- установлено только 1 колесо. Пожалуйста, установите еще 3 колеса\n";  
 } else if (wheelCounter == 2 || wheelCounter == 3) {  
 message = String.*format*("- установлено только %d колеса. Пожалуйста, установите еще %d колеса\n", wheelCounter, 4 - wheelCounter);  
 }  
 System.*out*.print(message);  
 if (engine == null) {  
 System.*out*.println("- двигатель не установлен. Пожалуйста, установите двигатель.");  
 } else {  
 isEngine = true;  
 }  
 if (isEngine && wheelCounter == 4) {  
 boolean ready = true;  
 for (int i = 0; i < 4; i++) {  
 if (!wheels.get(i).isGood()) {  
 System.*out*.printf("- №%d колесо сломано. Пожалуйста замените колесо.%n", i + 1);  
 ready = false;  
 }  
 }  
 if (!isFuel) {  
 System.*out*.println("- кончилось топливо. Необходимо заправиться.");  
 } else if (ready) {  
 System.*out*.println("- машина выезжает на дорогу!");  
 }  
 }  
 }  
}

public class Wheel {  
 private int diameter;  
 private boolean isGood;  
 Wheel(int diameter) {  
 this.diameter = diameter;  
 this.isGood = true;  
 }  
 int getDiameter() {  
 return diameter;  
 }  
 boolean isGood() {  
 return isGood;  
 }  
 public void setGoodWhell() {  
 isGood = true;  
 }  
 void breakWheel() {  
 this.isGood = false;  
 }  
}

public class Engine {  
 private int power;  
 private double volume;  
 Engine(int power, double volume) {  
 this.power = power;  
 this.volume = volume;  
 }  
 int getPower() {  
 return power;  
 }  
 public void setPower(int power) {  
 this.power = power;  
 }  
 double getVolume() {  
 return volume;  
 }  
 public void setVolume(float volume) {  
 this.volume = volume;  
 }  
}

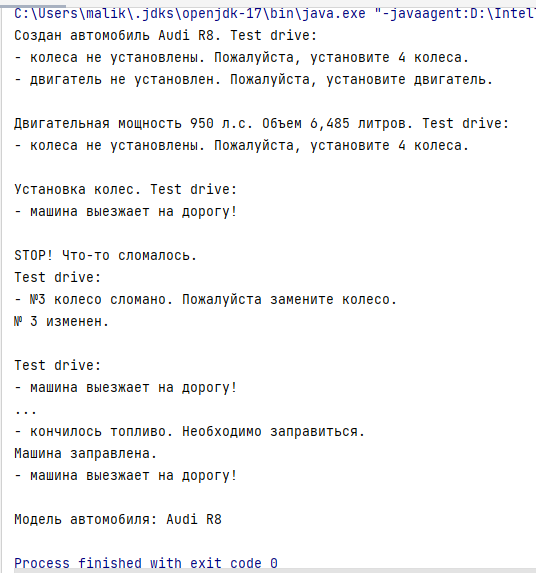


Рисунок 3 – Результат задачи 3

* 1. Задача 4

Создать класс Payment с внутренним классом, с помощью объектов которого можно сформировать покупку из нескольких товаров.

import java.io.IOException;  
public class Main {  
 public static void main(String[] args) throws IOException {  
 Payment payment = new Payment();  
 Payment.Order purchase = payment.new Order();  
 purchase.createShoppingList();  
 payment.amountMoneyPay(purchase.getOrderList());  
 payment.printShoppingList(purchase.getOrderList());  
 }  
}

import java.io.BufferedReader;  
import java.io.IOException;  
import java.io.InputStreamReader;  
import java.util.HashMap;  
import java.util.Map;  
public class Payment {  
 //Вычисление суммы денег к оплате за список товаров  
 public int amountMoneyPay(Map<Product, Integer> shoppingList) {  
 int sum = 0;  
 for (Map.Entry<Product, Integer> pair : shoppingList.entrySet()) {  
 sum += pair.getKey().getPrice() + pair.getValue();  
 }  
 return sum;  
 }  
 //вывод в консоль списка покупок и суммы к оплате  
 public void printShoppingList(Map<Product, Integer> shoppingList) {  
 System.*out*.println("Список покупок:");  
 for (Map.Entry<Product, Integer> pair : shoppingList.entrySet()) {  
 System.*out*.println(pair.getKey().toString() + " - " + pair.getValue() + " шт.");  
 }  
 System.*out*.println("Сумма к оплате: " + amountMoneyPay(shoppingList) + " руб.");  
 }  
 public class Order {  
 private Map<Product, Integer> orderList;  
 public Map<Product, Integer> getOrderList() {  
 return orderList;  
 }  
 //Создание списка покупок  
 public void createShoppingList() throws IOException {  
 BufferedReader reader = new BufferedReader(new InputStreamReader(System.*in*));  
 Map shoppingList = new HashMap<Product, Integer>();  
 boolean search = true;  
 while (search) {  
 Product product = DataUtilProduct.*searchProduct*(reader);  
 Integer quantity = DataUtilProduct.*readQuantity*(reader);  
 shoppingList.put(product, quantity);  
 while (true) {  
 System.*out*.print("Продолжить выбор товара (Да/Нет)? - ");  
 String continued = reader.readLine();  
 if (continued.equalsIgnoreCase("Да")) {  
 search = true;  
 break;  
 } else if (continued.equalsIgnoreCase("Нет")) {  
 search = false;  
 break;  
 } else {  
 System.*out*.println("Попробуйте еще раз.");  
 }  
 }  
 }  
 this.orderList = shoppingList;  
 }  
 }  
}

import java.io.BufferedReader;  
import java.io.IOException;  
import java.util.\*;  
public class DataUtilProduct {  
 public static List<Product> *productList* = new ArrayList<Product>() {  
 {  
 this.add(new Product("смартфон","samsung galaxy A53",26975));  
 this.add(new Product("смартфон","realme C21Y",10364));  
 this.add(new Product("смартфон","honor 70",36309));  
 this.add(new Product("ноутбук","asus",93500));  
 this.add(new Product("ноутбук","acer travel mate",63085));  
 this.add(new Product("ноутбук","lenovo",107190));  
 this.add(new Product("планшет","huawei",13050));  
 this.add(new Product("планшет","tcl",35000));  
 this.add(new Product("планшет","prestigio",10000));  
 this.add(new Product("наушники беспроводные","xiaomi redmi buds",1692));  
 this.add(new Product("наушники беспроводные","philips TAH4205BK",2995));  
 this.add(new Product("наушники беспроводные","zte",3300));  
 this.add(new Product("телевизор","blaupunkt",20990));  
 this.add(new Product("телевизор","thomson",30135));  
 this.add(new Product("телевизор","lg",58025));  
 }  
 };  
 //создание уникального списка товаров по наименованию  
 private static Set<String> createNameSet() {  
 Set<String> nameSet = new TreeSet<>();  
 for (Product list : *productList*) {  
 nameSet.add(list.getName());  
 }  
 return nameSet;  
 }  
 //ввод наименования товара и проверка его наличия в списке  
 private static String readNameProduct(BufferedReader reader) throws IOException {  
 System.*out*.println("Выберите наименование товара из списка: ");  
 System.*out*.print(*createNameSet*() + " - ");  
 String name;  
 while (true) {  
 name = reader.readLine();  
 if (*createNameSet*().contains(name.toLowerCase())) {  
 break;  
 } else {  
 System.*out*.print("Ничего не найдено, попробуйте еще раз: ");  
 }  
 }  
 return name;  
 }  
 //Поиск тавара по наименованию  
 private static List<Product> searchByNameProduct(BufferedReader reader) throws IOException {  
 String name = *readNameProduct*(reader);  
 List<Product> products = new ArrayList<>();  
 for (Product product : *productList*) {  
 if (name.equalsIgnoreCase(product.getName())) {  
 products.add(product);  
 }  
 }  
 return products;  
 }  
 //ввод производителя товара и проверка его наличия в списке  
 private static String readManufacturerProduct(Set<String> manufacturerSet, BufferedReader reader) throws IOException {  
 System.*out*.println("Выберите производителя товара из списка: ");  
 System.*out*.print(manufacturerSet + " - ");  
 String manufacturer;  
 while (true) {  
 manufacturer = reader.readLine();  
 if (manufacturerSet.contains(manufacturer.toLowerCase())) {  
 break;  
 } else {  
 System.*out*.print("Ничего не найдено, попробуйте еще раз: ");  
 }  
 }  
 return manufacturer;  
 }  
 //Поиск товара  
 public static Product searchProduct(BufferedReader reader) throws IOException {  
 Product product = null;  
 List<Product> list = *searchByNameProduct*(reader);  
 Set<String> manufacturerSet = new TreeSet<>();  
 for (Product element : list) {  
 manufacturerSet.add(element.getManufacturer());  
 }  
 String manufacturer = *readManufacturerProduct*(manufacturerSet, reader);  
 for (Product element : list) {  
 if (manufacturer.equalsIgnoreCase(element.getManufacturer())) {  
 product = element;  
 break;  
 }  
 }  
 return product;  
 }  
 //ввод количества товара  
 public static int readQuantity(BufferedReader reader) throws IOException {  
 System.*out*.print("Введите количество товара: ");  
 int quantity = Integer.*parseInt*(reader.readLine());  
 return quantity;  
 }  
}

public class Product {  
 private String name;  
 private String manufacturer;  
 private int price;  
 public Product(String name, String manufacturer, int price) {  
 this.name = name;  
 this.manufacturer = manufacturer;  
 this.price = price;  
 }  
 public String getName() {  
 return name;  
 }  
 public void setName(String name) {  
 this.name = name;  
 }  
 public String getManufacturer() {  
 return manufacturer;  
 }  
 public void setManufacturer(String manufacturer) {  
 this.manufacturer = manufacturer;  
 }  
 public int getPrice() {  
 return price;  
 }  
 public void setPrice(int price) {  
 this.price = price;  
 }  
 @Override  
 public String toString() {  
 return "наименование '" + name + '\'' +  
 ", производитель '" + manufacturer + '\'' +  
 ", цена " + price + " руб.";  
 }  
}

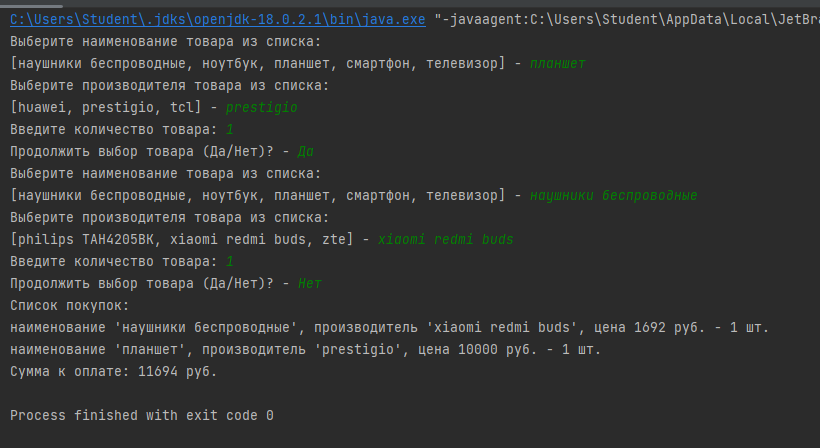


Рисунок 4 – Результат задачи 4

* 1. Задача 5

В тексте каждую букву заменить ее порядковым номером в алфавите. При выводе в одной строке печатать текст с двумя пробелами между буквами, в следующей строке внизу под каждой буквой печатать ее номер.

public class Main {  
 public static void main(String[] args) {  
 String text = "Я хочу пятёрку по учебной практике хаш хаш хаш";  
 int temp;  
 StringBuilder sb = new StringBuilder();  
 sb.append(text);  
 System.*out*.println("Оригинал -> " + sb.toString());  
 String formatchar = "%-3s";  
 String formatint = "%-3d";  
 for (int i = 0; i < sb.length(); i++) {  
 if (sb.charAt(i) != ' ') {  
 System.*out*.format(formatchar, sb.charAt(i));  
 }  
 }  
 System.*out*.println();  
 for (int i = 0; i < sb.length(); i++) {  
 if (sb.charAt(i) != ' ') {  
 temp = *getIndex*(sb.charAt(i));  
 System.*out*.format(formatint, temp);  
 }  
 }  
 }  
 public static int getIndex(char letter) {  
 int index = 0;  
 char lowerCase = Character.*toLowerCase*(letter);  
 switch (lowerCase) {  
 case 'а':  
 index = 1;  
 break;  
 case 'б':  
 index = 2;  
 break;  
 case 'в':  
 index = 3;  
 break;  
 case 'г':  
 index = 4;  
 break;  
 case 'д':  
 index = 5;  
 break;  
 case 'е':  
 index = 6;  
 break;  
 case 'ё':  
 index = 7;  
 break;  
 case 'ж':  
 index = 8;  
 break;  
 case 'з':  
 index = 9;  
 break;  
 case 'и':  
 index = 10;  
 break;  
 case 'й':  
 index = 11;  
 break;  
 case 'к':  
 index = 12;  
 break;  
 case 'л':  
 index = 13;  
 break;  
 case 'м':  
 index = 14;  
 break;  
 case 'н':  
 index = 15;  
 break;  
 case 'о':  
 index = 16;  
 break;  
 case 'п':  
 index = 17;  
 break;  
 case 'р':  
 index = 18;  
 break;  
 case 'с':  
 index = 19;  
 break;  
 case 'т':  
 index = 20;  
 break;  
 case 'у':  
 index = 21;  
 break;  
 case 'ф':  
 index = 22;  
 break;  
 case 'х':  
 index = 23;  
 break;  
 case 'ц':  
 index = 24;  
 break;  
 case 'ч':  
 index = 25;  
 break;  
 case 'ш':  
 index = 26;  
 break;  
 case 'щ':  
 index = 27;  
 break;  
 case 'ъ':  
 index = 28;  
 break;  
 case 'ы':  
 index = 29;  
 break;  
 case 'ь':  
 index = 30;  
 break;  
 case 'э':  
 index = 31;  
 break;  
 case 'ю':  
 index = 32;  
 break;  
 case 'я':  
 index = 33;  
 break;  
 }  
 return index;  
 }  
}

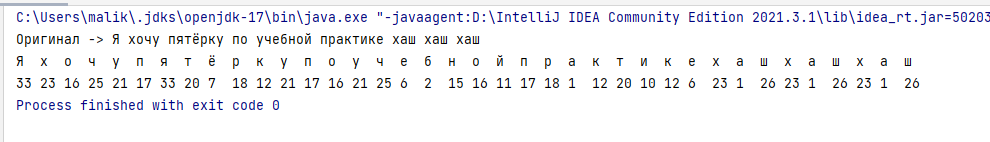


Рисунок 5 – Результат задачи 5

* 1. Задача 7

Составить программу получения на экране рисунка бокал с мороженным.

import javax.imageio.ImageIO;  
import javax.swing.JFrame;  
import javax.swing.JPanel;  
import javax.swing.WindowConstants;  
import java.awt.Graphics;  
import java.awt.image.BufferedImage;  
import java.io.File;  
import java.io.IOException;  
public class Main {  
 public static void main(String[] args) throws IOException{  
 JFrame frame = *buildFrame*();  
 final BufferedImage image = ImageIO.*read*(new File("C:\\Users\\malik\\Desktop\\УП1\\IceCream.png"));  
 JPanel pane = new JPanel() {  
 @Override  
 protected void paintComponent(Graphics g) {  
 super.paintComponent(g);  
 g.drawImage(image, 20, 20, null);  
 }  
 };  
 frame.add(pane);  
 }  
 private static JFrame buildFrame() {  
 JFrame frame = new JFrame("Бокал с мороженным");  
 frame.setDefaultCloseOperation(WindowConstants.*EXIT\_ON\_CLOSE*);  
 frame.setSize(300, 300);  
 frame.setVisible(true);  
 return frame;  
 }  
}

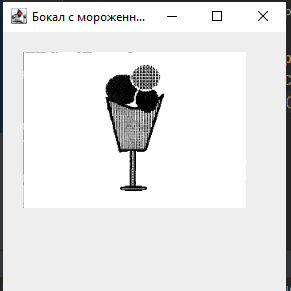
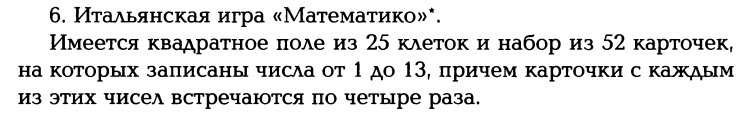
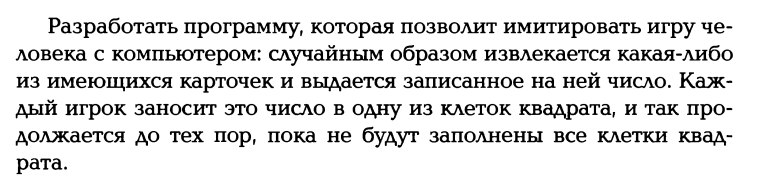


Рисунок 6 – Результат задачи 6

* 1. Задача 9







import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.util.HashMap;  
import java.util.LinkedList;  
import java.util.List;  
import java.util.Map;  
import javax.swing.\*;  
public class Main {  
 final static int *btns\_count* = 14;  
 final static int *btn\_width* = 60;  
 final static int *btn\_height* = 50;  
 static List<JButton>*player\_buttons* = new LinkedList<>();  
 static List<JButton> *bot\_buttons* = new LinkedList<>();  
 static List<Integer> *bot\_buttons\_active* = new LinkedList<>();  
 static Map<Integer, Integer> *integers* = new HashMap<>();  
 static Map<Integer, Integer> *bot\_integers* = new HashMap<>();  
 static int *steps* = 0;  
 static JLabel *player\_score\_label* = new JLabel("Очки: 0");  
 static JLabel *bot\_score\_label* = new JLabel("Очки: 0");  
 static int *score\_sum* = 0;  
 static int *score\_sum\_bot* = 0;  
 public static void main(String[] args) {  
 JFrame f = new JFrame();  
 JPanel panel = new JPanel();  
 JLabel player = new JLabel("Игрок");  
 player.setBounds(0, 10, 200, 50);  
 JLabel bot = new JLabel("Компьютер");  
 bot.setBounds(400, 10, 200, 50);  
 ActionListener actionListener = new TestActionListener();  
 panel.add(player);  
 panel.add(bot);  
 *player\_score\_label*.setBounds(0, 300, 200, 10);  
 *bot\_score\_label*.setBounds(400, 300, 200, 10);  
 panel.add(*player\_score\_label*);  
 panel.add(*bot\_score\_label*);  
 *player\_score\_label*.setVisible(false);  
 *bot\_score\_label*.setVisible(false);  
 for (int i = 0; i < 14; i++){  
 *integers*.put(i, 0);  
 *bot\_integers*.put(i, 0);  
 }  
 int value;  
 while (*bot\_buttons\_active*.size() < 25){  
 value = (int) (Math.*random*() \* 25);  
 if (!*bot\_buttons\_active*.contains(value)){  
 *bot\_buttons\_active*.add(value);  
 }  
 }  
 int x = 0;  
 int y = 0;  
 for (int i = 0; i < 25; i++){  
 if (i % 5 == 0){  
 x = 0;  
 y++;  
 }  
 JButton button1 = new JButton("0");  
 button1.setActionCommand(i + "");  
 button1.setBounds(*btn\_width* \* x,*btn\_height* \* y,*btn\_width*,*btn\_height*);  
 panel.add(button1, i);  
 button1.addActionListener(actionListener);  
 *player\_buttons*.add(button1);  
 x++;  
 }  
 x = 0;  
 y = 0;  
 for (int i = 0; i < 25; i++){  
 if (i % 5 == 0){  
 x = 0;  
 y++;  
 }  
 JButton button1 = new JButton("0");  
 button1.setActionCommand(i + "");  
 button1.setBounds(*btn\_width* \* x + 400,*btn\_height* \* y,*btn\_width*,*btn\_height*);  
 panel.add(button1, i);  
 button1.setEnabled(false);  
 *bot\_buttons*.add(button1);  
 x++;  
 }  
 panel.setLayout(null);  
 panel.setLocation(0,0);  
 f.setSize(720, 360);  
 f.add(panel);  
 f.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 f.setVisible(true);  
 }  
 public static class TestActionListener implements ActionListener {  
 public void actionPerformed(ActionEvent e) {  
 Object source = e.getSource();  
 ((JButton) source).setText("" + *rnd*(*btns\_count*, false));  
 ((JButton) source).setEnabled(false);  
 *bot\_buttons*.get(*bot\_buttons\_active*.get(0)).setText("" + *rnd*(*btns\_count*, true));  
 *bot\_buttons\_active*.remove(0);  
 if(*steps* == 24){  
 int[] result = *result*(*player\_buttons*,*bot\_buttons*);  
 *player\_score\_label*.setVisible(true);  
 *player\_score\_label*.setText("Очки: " + result[0]);  
 *bot\_score\_label*.setVisible(true);  
 *bot\_score\_label*.setText("Очки: " + result[1]);  
 }  
 else{  
 *steps*++;  
 }  
 }  
 }  
 public static int[] result( List<JButton>player\_buttons, List<JButton>bot\_buttons){  
 int[] result = new int[2];  
 result[0] = 0;  
 result[1] = 0;  
 int x1 = 0;  
 int y1 = 0;  
 int[][] values = new int[5][5];  
 for (int i = 0; i < 24; i++){  
 if (i % 5 == 0){  
 x1 = 0;  
 y1++;  
 }  
 values[y1-1][x1] = Integer.*parseInt*(player\_buttons.get(i).getText());  
 x1++;  
 }  
 int score\_row\_sum = 0;  
 int score\_row = 0;  
 for (int k = 0; k < 14; k++){  
 for (int i = 0; i < 5; i++){  
 for (int j = 0; j < 5; j++){  
 if (values[i][j] == k && values[i][j]!=-1){  
 score\_row++;  
 }  
 }  
 if (score\_row > 1){  
 score\_row\_sum = score\_row\_sum + score\_row;  
 }  
 score\_row = 0;  
 }  
 if (score\_row\_sum == 2){  
 *score\_sum* = *score\_sum* + 10;  
 }  
 if (score\_row\_sum >= 3){  
 *score\_sum* = *score\_sum* + 40;  
 }  
 score\_row\_sum = 0;  
 }  
 result[0] = *score\_sum*;  
 x1 = 0;  
 y1 = 0;  
 values = new int[5][5];  
 for (int i = 0; i < 24; i++){  
 if (i % 5 == 0){  
 x1 = 0;  
 y1++;  
 }  
 values[y1-1][x1] = Integer.*parseInt*(bot\_buttons.get(i).getText());  
 x1++;  
 }  
 score\_row\_sum = 0;  
 score\_row = 0;  
 for (int k = 0; k < 14; k++){  
 for (int i = 0; i < 5; i++){  
 for (int j = 0; j < 5; j++){  
 if (values[i][j] == k && values[i][j]!=-1){  
 score\_row++;  
 }  
 }  
 if (score\_row > 1){  
 score\_row\_sum = score\_row\_sum + score\_row;  
 }  
 score\_row = 0;  
 }  
 if (score\_row\_sum == 2){  
 *score\_sum\_bot* = *score\_sum\_bot* + 10;  
 }  
 if (score\_row\_sum >= 3){  
 *score\_sum\_bot* = *score\_sum\_bot* + 40;  
 }  
 score\_row\_sum = 0;  
 }  
 result[1] = *score\_sum\_bot*;  
 return result;  
 }  
 public static int rnd(final double max, boolean isBot)  
 {  
 int value = (int) (Math.*random*() \* max);  
 if (!isBot){  
 while (*integers*.get(value) > 3){  
 value = (int) (Math.*random*() \* max);  
 }  
 *integers*.put(value, *integers*.get(value) + 1);  
 }  
 else {  
 while (*bot\_integers*.get(value) > 3){  
 value = (int) (Math.*random*() \* max);  
 }  
 *bot\_integers*.put(value, *bot\_integers*.get(value) + 1);  
 }  
 return value;}}



Рисунок 7 – Результат задачи 7

* 1. Меню с задачами

Выбор задач реализовать через меню! Одно приложение должно содержать решение 9-ти задач.

import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.net.URL;  
import javax.swing.JButton;  
import javax.swing.JFrame;  
import javax.swing.JPanel;  
import javax.swing.JTextField;  
public class Main extends JFrame {  
 private JTextField textField;  
 public Main() {  
 super("Соединение заданий");  
 createGUI();  
 }  
 public void createGUI() {  
 setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 JPanel panel = new JPanel();  
 panel.setLayout(new FlowLayout());  
 ActionListener actionListener = new TestActionListener();  
 JButton button1 = new JButton("Задача 1");  
 button1.setActionCommand("https://github.com/Euros2308/-1-.-19-1-1/tree/main/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B01");  
 panel.add(button1);  
 button1.addActionListener(actionListener);  
 JButton button2 = new JButton("Задача 2");  
 button2.setActionCommand("https://github.com/Euros2308/-1-.-19-1-1/tree/main/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B02");  
 panel.add(button2);  
 button2.addActionListener(actionListener);  
 JButton button3 = new JButton("Задача 3");  
 button3.setActionCommand("https://github.com/Euros2308/-1-.-19-1-1/tree/main/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B03");  
 panel.add(button3);  
 button3.addActionListener(actionListener);  
 JButton button4 = new JButton("Задача 4");  
 button4.setActionCommand("https://github.com/Euros2308/-1-.-19-1-1/tree/main/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B04");  
 panel.add(button4);  
 button4.addActionListener(actionListener);  
 JButton button5 = new JButton("Задача 5");  
 button5.setActionCommand("https://github.com/Euros2308/-1-.-19-1-1/tree/main/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B05");  
 panel.add(button5);  
 button5.addActionListener(actionListener);  
 JButton button6 = new JButton("Задача 6");  
 button6.setActionCommand("https://github.com/Euros2308/-1-.-19-1-1/tree/main/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B07");  
 panel.add(button6);  
 button6.addActionListener(actionListener);  
 JButton button7 = new JButton("Задача 7");  
 button7.setActionCommand("https://github.com/Euros2308/-1-.-19-1-1/tree/main/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B09");  
 panel.add(button7);  
 button7.addActionListener(actionListener);  
 getContentPane().add(panel);  
 setPreferredSize(new Dimension(320, 200));  
 }  
 public class TestActionListener implements ActionListener {  
 public void actionPerformed(ActionEvent e) {  
 try {  
 Desktop.*getDesktop*().browse(new URL(e.getActionCommand()).toURI());  
 } catch (Exception ex) {  
 System.*out*.println(ex.toString());  
 }  
 }  
 }  
 public static void main(String[] args) {  
 javax.swing.SwingUtilities.*invokeLater*(new Runnable() {  
 public void run() {  
 JFrame.*setDefaultLookAndFeelDecorated*(true);  
 Main frame = new Main();  
 frame.pack();  
 frame.setLocationRelativeTo(null);  
 frame.setVisible(true);  
 }  
 });  
 }  
}

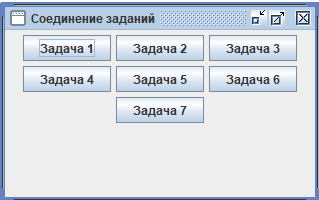


Рисунок 8 – Результат задания

* 1. Задача 10

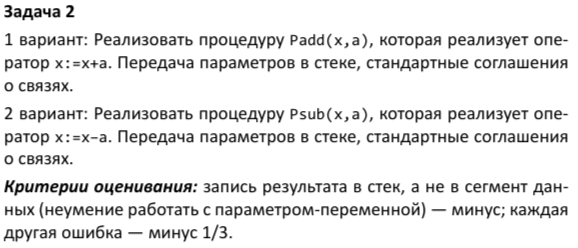


Таблица 1 - Структура программы 1

| Код команды | Мнемокод | Комментарий |
| --- | --- | --- |
| AF | XRA A | Очистить аккумулятор |
| 3E | MVI A | Записать в аккумулятор |
| 08 |  | Число x |
| 06 | MVI B | Записать в регистр B |
| 10 |  | Число a |
| 80 | ADD B | Сложить x и a |
| E7 | RST 7 | Прервать выполнение программы |

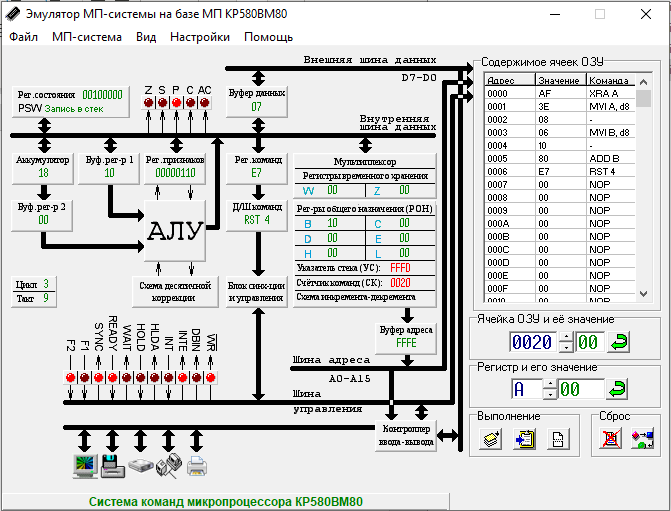


Рисунок 9 – Результат программы 1

Таблица 2 - Структура программы 2

| Код команды | Мнемокод | Комментарий |
| --- | --- | --- |
| AF | XRA A | Очистить аккумулятор |
| 3E | MVI A | Записать в аккумулятор |
| 08 |  | Число x |
| 06 | MVI B | Записать в регистр B |
| 10 |  | Число a |
| 90 | SUB B | Вычесть из x и a |
| E7 | RST 7 | Прервать выполнение программы |

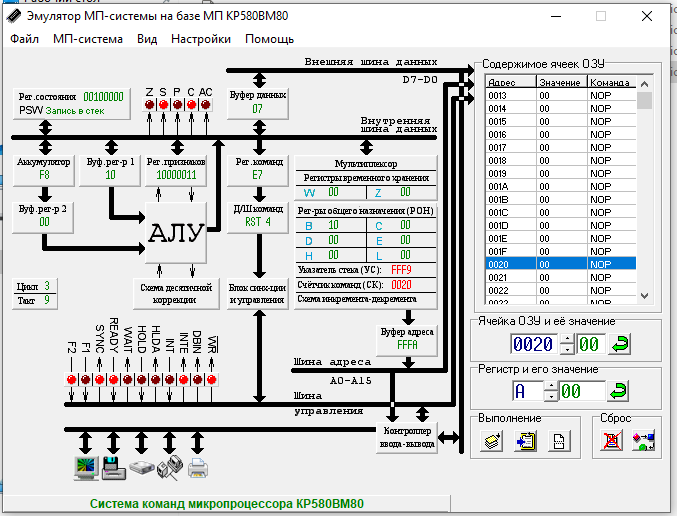
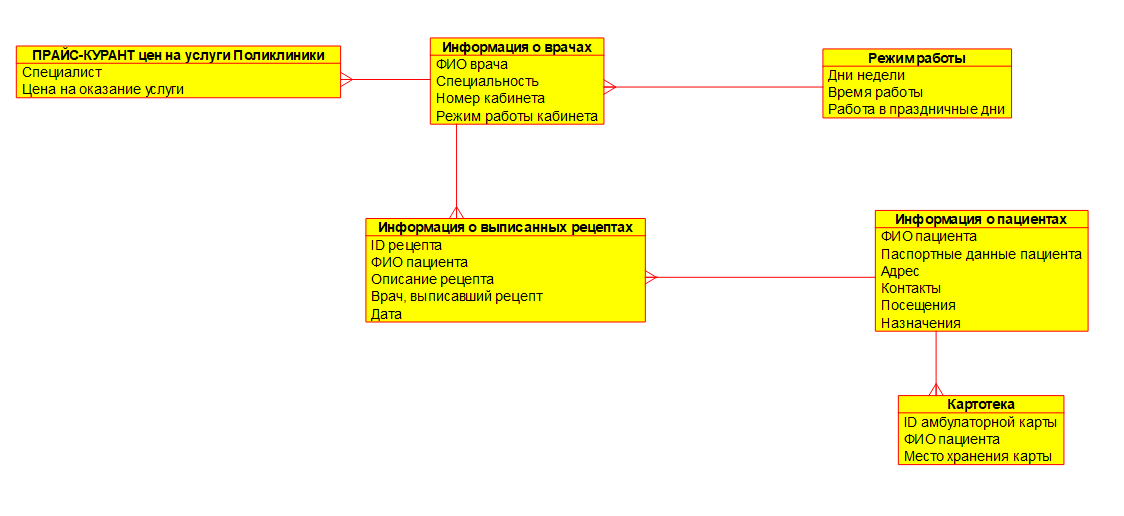


Рисунок 10 – Результат программы 2

1. Разработка АИС «Поликлиника»
   1. ER-диаграмма



* 1. Техническое задание