|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

**Отчет по лабораторной работе 3**

***по дисциплине «Языки программирования для работы с большими данными»***

Вариант

Студент ИУ6-21М **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гришина А. М.**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Степанов П. В.**

(Подпись, дата) (И.О.Фамилия)

*2020 г.*

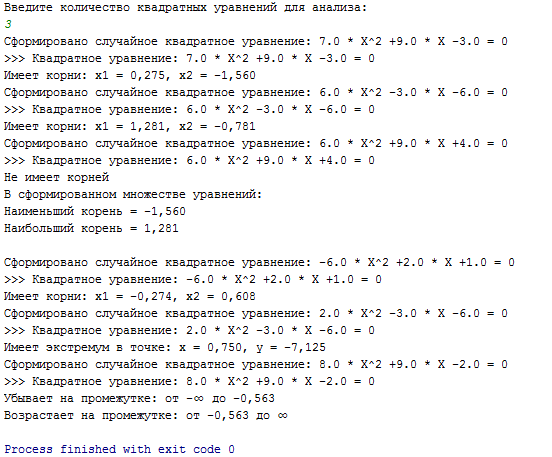
Вариант 1

1. Определить класс Квадратное уравнение. Класс должен содержать несколько конструкторов. Реализовать методы для поиска корней, экстремумов, а также интервалов убывания/возрастания. Создать массив объектов и определить наибольшие и наименьшие по значению корни.

**Код**:

**package** com.company;  
  
**import** java.util.Scanner;  
**import** java.util.Random;  
  
**public class** Main {  
  
 **public static void** main(String[] args) {  
 Scanner sc = **new** Scanner(System.***in***);  
 System.***out***.println(**"Введите количество квадратных уравнений для анализа:"**);  
 **int** count = sc.nextInt();  
 Quadratic[] Q = **new** Quadratic[count];  
 **double**[] M = **new double**[count\*2];  
 **for** (**int** i = 0; i < Q.**length**; i++ ) {  
 Q[i] = **new** Quadratic();  
 Q[i].solution();  
 M[2\*i] = Q[i].**x1**;  
 M[2\*i+1] = Q[i].**x2**;  
 }  
 **double** min = M[0], max = M[0];  
 **for**(**double** j : M){  
 min = j < min ? j : min;  
 max = j > max ? j : max;  
 }  
 System.***out***.printf(  
 **"В сформированном множестве уравнений:\n"** +  
 **"Наименьший корень = %.3f\n"** +  
 **"Наибольший корень = %.3f\n\n"**, min, max);  
 Quadratic Q1 = **new** Quadratic();  
 Q1.solution();  
 Quadratic Q2 = **new** Quadratic();  
 Q2.extremum();  
 Quadratic Q3 = **new** Quadratic();  
 Q3.intervals();  
 }  
}  
  
**class** Quadratic {  
  
 **private double a**, **b**, **c**, **ex**, **ey**;  
 **public double x1**, **x2**;  
 **private** String **s** = **""**;  
  
 Quadratic(){  
 Random r = **new** Random();  
 **while** (**a** == 0) {  
 **a** = r.nextInt(20) - 10;  
 }  
 **while** (**b** == 0) {  
 **b** = r.nextInt(20) - 10;  
 }  
 **while** (**c** == 0) {  
 **c** = r.nextInt(20) - 10;  
 }  
 Builder(**true**);  
 }  
 Quadratic(**double** qa){  
 **a** = qa;  
 Builder(**false**);  
 }  
 Quadratic(**double** qa, **double** qb){  
 **a** = qa;  
 **b** = qb;  
 Builder(**false**);  
 }  
 Quadratic(**double** qa, **double** qb, **double** qc){  
 **a** = qa;  
 **b** = qb;  
 **c** = qc;  
 Builder(**false**);  
 }  
  
 **private void** Builder(Boolean R){  
 **if** (**a** == 0)  
 System.***out***.println(**"Коэффициент \"а\" не может быть равен 0"**);  
 **else**{  
 String x2 = **a** > 0 ? (**a** + **" \* X^2 "**) : (**a** + **" \* X^2 "**);  
 String x1 = **b** == 0 ? **""** : **b** > 0 ? (**"+"** + **b** + **" \* X "**) : (**b** + **" \* X "**);  
 String x0 = **c** == 0 ? **""** : **c** > 0 ? (**"+"** + **c**) : (**""** + **c**);  
 String t = R ? **"случайное "** : **""**;  
 **s** = x2 + x1 + x0;  
 System.***out***.printf(**"Сформировано %sквадратное уравнение: %s = 0\n"**, t, **s**);  
 }  
 }  
  
 **void** solution(){  
 **double** D;  
 D = **b** \* **b** - 4 \* **a** \* **c**;  
 **if** (D >= 0){  
 **x1** = ( (-1) \* **b** + Math.*sqrt*(D) ) / (2 \* **a**);  
 **x2** = ( (-1) \* **b** - Math.*sqrt*(D) ) / (2 \* **a**);  
 System.***out***.printf(**">>> Квадратное уравнение: %s = 0\nИмеет корни: x1 = %.3f, x2 = %.3f\n"**, **s**, **x1**, **x2**);  
 }  
 **else**{  
 System.***out***.printf(**">>> Квадратное уравнение: %s = 0\nНе имеет корней\n"**, **s**);  
 }  
 }  
  
 **void** extremum() {  
 **ex** = (-1) \* **b** / (2 \* **a**);  
 **ey** = **a** \* **ex** \* **ex** + **b** \* **ex** + **c**;  
 System.***out***.printf(**">>> Квадратное уравнение: %s = 0\nИмеет экстремум в точке: x = %.3f, y = %.3f\n"**, **s**, **ex**, **ey**);  
 }  
  
 **void** intervals(){  
 **ex** = (-1) \* **b** / (2 \* **a**);  
 **ey** = **a** \* **ex** \* **ex** + **b** \* **ex** + **c**;  
 **double** ex1 = **ex** - 10, ey1 = **a** \* ex1 \* ex1 + **b** \* ex1 + **c**;  
 System.***out***.printf(**">>> Квадратное уравнение: %s = 0\n"**, **s**);  
 **if** (ey1 > **ey**){  
 System.***out***.printf(**"Убывает на промежутке: от -\u221E до %.3f\n"**, **ex**);  
 System.***out***.printf(**"Возрастает на промежутке: от %.3f до \u221E \n"**, **ex**);  
 }  
 **else**{  
 System.***out***.printf(**"Возрастает на промежутке: от -\u221E до %.3f\n"**, **ex**);  
 System.***out***.printf(**"Убывает на промежутке: от %.3f до \u221E \n"**, **ex**);  
 }  
 }  
}

**Проверка правильности работы**:



**Вывод:** Класс Квадратное уравнение содержит несколько конструкторов. В программе реализованы методы для поиска корней, экстремумов, а также интервалов убывания/возрастания. Программа работает корректно.

Вариант 2

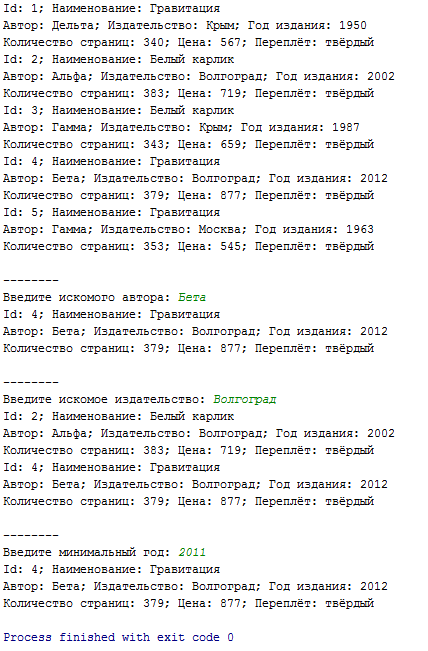
Создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль.

5. Book: id, Название, Автор(ы), Издательство, Год издания, Количество страниц, Цена, Переплет. Создать массив объектов. Вывести: a) список книг заданного автора; b) список книг, выпущенных заданным издательством; c) список книг, выпущенных после заданного года.

**Код**:

**package** com.company;  
  
**import** java.util.Random;  
**import** java.util.Scanner;  
**import** java.lang.String;  
  
**public class** Main {  
  
 **public static void** main(String[] args) {  
  
 String[] AllWriter = **new** String[]  
 {**"Альфа"**, **"Бета"**, **"Гамма"**, **"Дельта"**};  
 String[] AllPublisher = **new** String[]  
 {**"Москва"**, **"Санкт-Петербург"**, **"Волгоград"**, **"Крым"**};  
 String[] AllName = **new** String[]  
 {**"Гравитация"**, **"Сингулярность"**, **"Комета"**, **"Белый карлик"**, **"Чёрная дыра"**, **"Высадка"**};  
 String[] AllSkin = **new** String[]  
 {**"твёрдый"**, **"мягкий"**};  
 Scanner sc = **new** Scanner(System.***in***);  
  
 Book[] books = **new** Book[5];  
 **for**(**int** i = 0; i < books.**length**; i++){  
 Random r = **new** Random();  
 books[i] = **new** Book(  
 i+1, AllName[r.nextInt(6)],  
 AllWriter[r.nextInt(4)], AllPublisher[r.nextInt(4)],  
 r.nextInt(70) + 1950, r.nextInt(150) + 300, r.nextDouble() \* 1000,  
 AllSkin[r.nextInt(1)]);  
 System.***out***.println(books[i].toString());  
 }  
  
 System.***out***.print(**"\n--------\nВведите искомого автора: "**);  
 String writer = sc.nextLine();  
 **int** i = 0;  
 **for** (Book b : books) {  
 **if** (writer.equals(b.getWriter())) {  
 System.***out***.println(b.toString());  
 i += 1;  
 }  
 }  
 **if** (i == 0) System.***out***.printf(**"Книги автора \"%s\" отсутствуют\n"**, writer);  
  
 System.***out***.print(**"\n--------\nВведите искомое издательство: "**);  
 String publisher = sc.nextLine();  
 i = 0;  
 **for** (Book b : books){  
 **if** (publisher.equals(b.getPublisher())) {  
 System.***out***.println(b.toString());  
 i += 1;  
 }  
 }  
 **if** (i == 0) System.***out***.printf(**"Книги издательства \"%s\" отсутствуют\n"**, publisher);  
  
 System.***out***.print(**"\n--------\nВведите минимальный год: "**);  
 **int** year = sc.nextInt();  
 i = 0;  
 **for** (Book b : books) {  
 **if** (b.getYear() > year) {  
 System.***out***.println(b.toString());  
 i += 1;  
 }  
 }  
 **if** (i == 0) System.***out***.printf(**"Книги выпущенные после %d года отсутствуют\n"**, year);  
  
 }  
  
}  
  
**class** Book{  
  
 **private int Id** = 0, **Year** = 0, **Size** = 0;  
 **private double Price** = 0;  
 **private** String **Name** = **""**, **Writer** = **""**, **Publisher** = **""**, **Skin** = **""**;  
  
 Book(){  
 Scanner sc = **new** Scanner(System.***in***);  
 System.***out***.print(**"Введите Id: "**);  
 setId(sc.nextInt());  
 sc.nextLine();  
 System.***out***.print(**"Введите название книги: "**);  
 setName(sc.nextLine());  
 System.***out***.print(**"Введите автора: "**);  
 setWriter(sc.nextLine());  
 System.***out***.print(**"Введите издательство: "**);  
 setPublisher(sc.nextLine());  
 System.***out***.print(**"Введите год издания: "**);  
 setYear(sc.nextInt());  
 sc.nextLine();  
 System.***out***.print(**"Введите количество страниц: "**);  
 setSize(sc.nextInt());  
 sc.nextLine();  
 System.***out***.print(**"Введите стоимость: "**);  
 setPrice(sc.nextDouble());  
 sc.nextLine();  
 System.***out***.print(**"Введите тип переплёта: "**);  
 setSkin(sc.next());  
 }  
  
  
 Book(**int** id, String name, String writer, String publisher, **int** year, **int** size, **double** price, String skin){  
 setId(id);  
 setName(name);  
 setWriter(writer);  
 setPublisher(publisher);  
 setYear(year);  
 setSize(size);  
 setPrice(price);  
 setSkin(skin);  
 }  
  
 **public int** getId(){ **return Id**; }  
 **public int** getYear(){ **return Year**; }  
 **public int** getSize(){ **return Size**; }  
 **public double** getPrice(){ **return Price**; }  
 **public** String getName(){ **return Name**; }  
 **public** String getWriter(){ **return Writer**; }  
 **public** String getPublisher(){ **return Publisher**; }  
 **public** String getSkin(){ **return Skin**; }  
  
 **public void** setId(**int** id){  
 **if**(id > 0) **Id** = id;  
 **else** System.***out***.println(**"Значение идентификатора Id должно быть больше нуля"**);  
 }  
 **public void** setYear(**int** year){  
 **if** (year > 0) **Year** = year;  
 **else** System.***out***.println(**"Откуда у Вас книга старше н.э.?"**);  
 }  
 **public void** setSize(**int** size){  
 **if** (size > 0) **Size** = size;  
 **else** System.***out***.println(**"Количество страниц должно быть больше нуля"**);  
 }  
 **public void** setPrice(**double** price){  
 **if** (price >= 0) **Price** = price;  
 **else** System.***out***.println(**"Стоимость не может быть ниже нуля"**);  
 }  
 **public void** setName(String name){  
 String check = name.replaceAll(**" "**,**""**);  
 **if** (check.length() > 0) **Name** = name;  
 **else** System.***out***.println(**"Необходимо указать наименование книги"**);  
 }  
 **public void** setWriter(String writer){  
 String check = writer.replaceAll(**" "**,**""**);  
 **if** (check.length() > 0) **Writer** = writer;  
 **else Writer** = **"unknown"**;  
 }  
 **public void** setPublisher(String publisher){  
 String check = publisher.replaceAll(**" "**,**""**);  
 **if** (check.length() > 0) **Publisher** = publisher;  
 **else Publisher** = **"unknown"**;  
 }  
 **public void** setSkin(String skin){  
 **if** (skin.toLowerCase() == **"твёрдый"** || skin.toLowerCase() == **"мягкий"**) **Skin** = skin.toLowerCase();  
 **else** System.***out***.println(**"Для описания переплёта доступны значения: <твёрдый> и <мягкий>"**);  
 }  
  
 @Override  
 **public** String toString() {  
 String str =  
 **"Id: "** + **Id** + **"; Наименование: "** + **Name** +  
 **"\nАвтор: "** + **Writer** + **"; Издательство: "** + **Publisher** + **"; Год издания: "** + **Year** +  
 **"\nКоличество страниц: "** + **Size** + **"; Цена: "** + Math.*round*(**Price**) + **"; Переплёт: "** + **Skin**;  
 **return** str;  
 }  
}

**Проверка правильности работы**:



**Вывод:** Программа выводит: a) список книг заданного автора; b) список книг, выпущенных заданным издательством; c) список книг, выпущенных после заданного года. Программа работает корректно.

Вариант 3

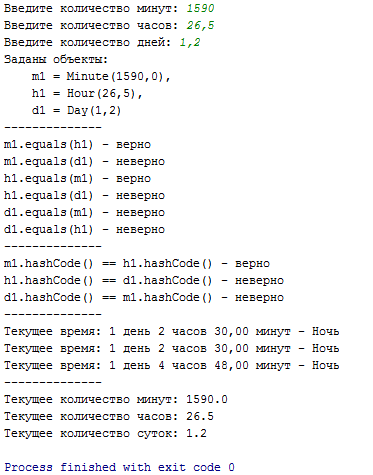
Создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString().

11. Создать объект класса Сутки, используя классы Час, Минута. Методы: вывести на консоль текущее время, рассчитать время суток (утро, день, вечер, ночь).

**Код**:

**package** com.company;  
  
**import** java.util.Random;  
**import** java.util.Scanner;  
  
**public class** Main {  
  
 **public static void** main(String[] args) {  
 Minute m1 = **new** Minute();  
 Hour h1 = **new** Hour();  
 Day d1 = **new** Day();  
  
 System.***out***.printf(  
 **"Заданы объекты:\n\tm1 = Minute(%.1f),\n\th1 = Hour(%.1f),\n\td1 = Day(%.1f)\n"**,  
 m1.getValue(), h1.getValue(), d1.getValue());  
 System.***out***.println(**"--------------"**);  
 System.***out***.println(**"m1.equals(h1) - "** + (m1.equals(h1) ? **"верно"** : **"неверно"**));  
 System.***out***.println(**"m1.equals(d1) - "** + (m1.equals(d1) ? **"верно"** : **"неверно"**));  
 System.***out***.println(**"h1.equals(m1) - "** + (h1.equals(m1) ? **"верно"** : **"неверно"**));  
 System.***out***.println(**"h1.equals(d1) - "** + (h1.equals(d1) ? **"верно"** : **"неверно"**));  
 System.***out***.println(**"d1.equals(m1) - "** + (d1.equals(m1) ? **"верно"** : **"неверно"**));  
 System.***out***.println(**"d1.equals(h1) - "** + (d1.equals(h1) ? **"верно"** : **"неверно"**));  
 System.***out***.println(**"--------------"**);  
 System.***out***.println(**"m1.hashCode() == h1.hashCode() - "** + (m1.hashCode() == h1.hashCode() ? **"верно"** : **"неверно"**));  
 System.***out***.println(**"h1.hashCode() == d1.hashCode() - "** + (h1.hashCode() == d1.hashCode() ? **"верно"** : **"неверно"**));  
 System.***out***.println(**"d1.hashCode() == m1.hashCode() - "** + (d1.hashCode() == m1.hashCode() ? **"верно"** : **"неверно"**));  
 System.***out***.println(**"--------------"**);  
 m1.CurrentTime();  
 h1.CurrentTime();  
 d1.CurrentTime();  
 System.***out***.println(**"--------------"**);  
 System.***out***.println(m1.toString());  
 System.***out***.println(h1.toString());  
 System.***out***.println(d1.toString());  
 }  
}  
  
**class** Minute{  
 **private double Value** = 0;  
 **public double** getValue(){  
 **return Value**;  
 }  
 Minute(**double** value){  
 **Value** = value;  
 }  
 Minute(){  
 Scanner sc = **new** Scanner(System.***in***);  
 System.***out***.print(**"Введите количество минут: "**);  
 **Value** = sc.nextDouble();  
 }  
 **void** CurrentTime(){  
 **int** d = (**int**)(**Value**) / 1440;  
 **int** h = (**int**)(**Value** - d \* 1440) / 60;  
 **double** m = **Value** - d \* 1440 - h \* 60;  
 String dt = **""**;  
 **if** (h < 6) dt = **"Ночь"**;  
 **else if** (h < 12) dt = **"Утро"**;  
 **else if** (h < 18) dt = **"День"**;  
 **else if** (h < 24) dt = **"Вечер"**;  
 System.***out***.printf(**"Текущее время: %d день %d часов %.2f минут - %s\n"**, d, h, m, dt);  
 }  
 @Override  
 **public boolean** equals(Object o){  
 **if** (o.getClass().getName() == **"com.company.Hour"** ){  
 Hour h = (Hour) o;  
 **return Value** / 60 == h.getValue();  
 }  
 **if** (o.getClass().getName() == **"com.company.Day"** ){  
 Day d = (Day) o;  
 **return Value** / 24 / 60 == d.getValue();  
 }  
 **return false**;  
 }  
 @Override  
 **public int** hashCode(){  
 **return** (**int**)(**Value**);  
 }  
 @Override  
 **public** String toString(){  
 **return "Текущее количество минут: "** + **Value**;  
 }  
}  
  
**class** Hour{  
 **private double Value** = 0;  
 **public double** getValue(){  
 **return Value**;  
 }  
 Hour(**double** value){  
 **Value** = value;  
 }  
 Hour(){  
 Scanner sc = **new** Scanner(System.***in***);  
 System.***out***.print(**"Введите количество часов: "**);  
 **Value** = sc.nextDouble();  
 }  
 **void** CurrentTime(){  
 Minute m = **new** Minute(**Value** \* 60);  
 m.CurrentTime();  
 }  
 @Override  
 **public boolean** equals(Object o){  
 **if** (o.getClass().getName() == **"com.company.Minute"** ){  
 Minute m = (Minute) o;  
 **return Value** \* 60 == m.getValue();  
 }  
 **if** (o.getClass().getName() == **"com.company.Day"** ){  
 Day d = (Day) o;  
 **return Value** / 24 == d.getValue();  
 }  
 **return false**;  
 }  
 @Override  
 **public int** hashCode(){  
 **return** (**int**)(**Value** \* 60);  
 }  
 @Override  
 **public** String toString(){  
 **return "Текущее количество часов: "** + **Value**;  
 }  
}  
  
**class** Day{  
 **private double Value** = 0;  
 **public double** getValue(){  
 **return Value**;  
 }  
 Day(**double** value){  
 **Value** = value;  
 }  
 Day(){  
 Scanner sc = **new** Scanner(System.***in***);  
 System.***out***.print(**"Введите количество дней: "**);  
 **Value** = sc.nextDouble();  
 }  
 **void** CurrentTime(){  
 Minute m = **new** Minute(**Value** \* 24 \* 60);  
 m.CurrentTime();  
 }  
 @Override  
 **public boolean** equals(Object o){  
 **if** (o.getClass().getName() == **"com.company.Minute"** ){  
 Minute m = (Minute) o;  
 **return Value** \* 24 \* 60 == m.getValue();  
 }  
 **if** (o.getClass().getName() == **"com.company.Hour"** ){  
 Hour h = (Hour) o;  
 **return Value** \* 24 == h.getValue();  
 }  
 **return false**;  
 }  
 @Override  
 **public int** hashCode(){  
 **return** (**int**)(**Value** \* 60 \* 24);  
 }  
 @Override  
 **public** String toString(){  
 **return "Текущее количество суток: "** + **Value**;  
 }  
}

**Проверка правильности работы**:



**Вывод**: Программа выводит текущее время и рассчитывает время суток. Программа работает корректно.

Вариант 4

Построить модель программной системы.  
  
9. Система Интернет-магазин. Администратор добавляет информацию о Товаре. Клиент делает и оплачивает Заказ на Товары. Администратор регистрирует Продажу и может занести неплательщиков в «черный список».   
  
**Код**:

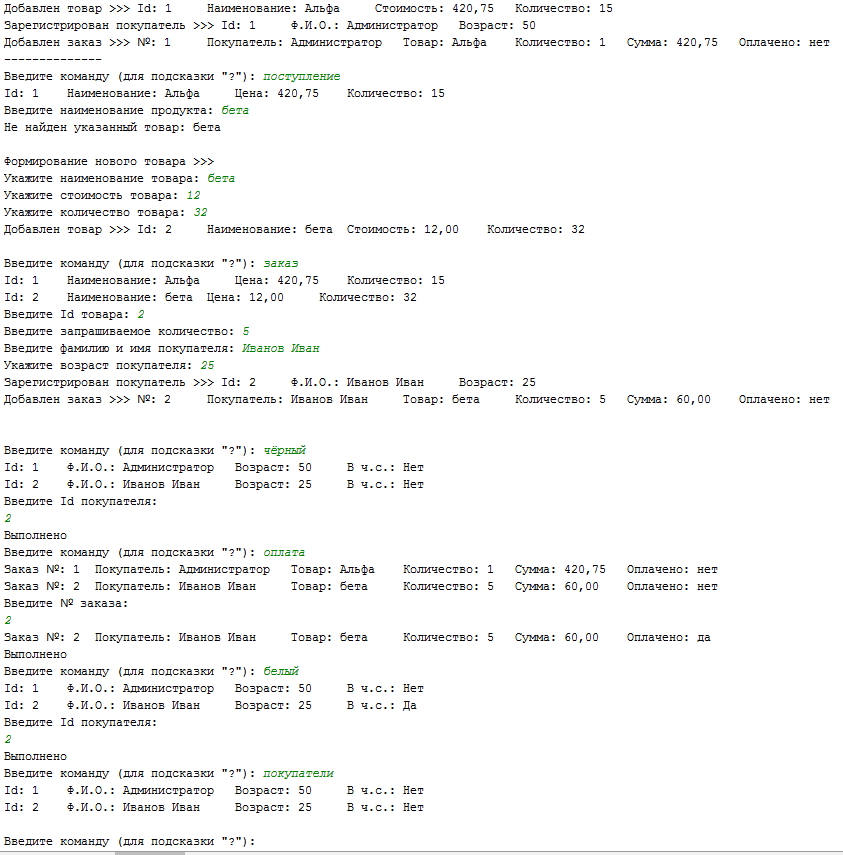
**package** com.company;  
  
**import** java.util.Scanner;  
  
**public class** Main {  
  
 **static** Product[] *products* = **new** Product[100];  
 **static** Order[] *orders* = **new** Order[100];  
 **static** Buyer[] *buyers* = **new** Buyer[100];  
  
 **static void** all(Product[] o){  
 **for** (**int** i = 0; i < o.**length**; i++) {  
 o[i].info();  
 **if** (o[i].last()) **break**;  
 }  
 }  
 **static void** all(Order[] o){  
 **for** (**int** i = 0; i < o.**length**; i++) {  
 o[i].info();  
 **if** (o[i].last()) **break**;  
 }  
 }  
 **static void** all(Buyer[] o){  
 **for** (**int** i = 0; i < o.**length**; i++) {  
 o[i].info();  
 **if** (o[i].last()) **break**;  
 }  
 }  
  
 **public static void** main(String[] args) {  
 Scanner cmd = **new** Scanner(System.***in***);  
 **boolean** done = **false**;  
 String temp;  
 **int** itemp;  
  
 *products*[0] = **new** Product(**"Альфа"**, 420.75, 15);  
 *buyers*[0] = **new** Buyer(**"Администратор"**, 50);  
 *orders*[0] = **new** Order(*buyers*[0], *products*[0], 1);  
  
 *//all(products); all(buyers); all(orders);* System.***out***.println(**"--------------"**);  
  
 **while** (**true**) {  
 System.***out***.print(**"Введите команду (для подсказки \"?\"): "**);  
 String com = cmd.nextLine();  
 **switch** (com) {  
 **case "?"**:  
 System.***out***.println(**"Доступны следующие команды: "**);  
 System.***out***.println(**"\"поступление\" - изменение кол-ва товара"**);  
 System.***out***.println(**"\"заказ\" - сформировать заказ"**);  
 System.***out***.println(**"\"оплата\" - фиксация оплаты пользователя"**);  
 System.***out***.println(**"\"чёрный\" - добавление пользователя в чёрный список"**);  
 System.***out***.println(**"\"белый\" - удаление пользователя из чёрный список"**);  
 System.***out***.println(**"\"заказы\" - все заказы"**);  
 System.***out***.println(**"\"товары\" - все товары"**);  
 System.***out***.println(**"\"покупатели\" - все покупатели"**);  
 cmd.nextLine();  
 **break**;  
  
 **case "поступление"**:  
 done = **false**;  
 *all*(*products*);  
 System.***out***.print(**"Введите наименование продукта: "**);  
 temp = cmd.nextLine();  
 **for** (**int** p = 1; !*products*[p-1].last(); p++) {  
 **if** (*products*[p-1].getName().equals(temp)) {  
 System.***out***.print(**"Введите изменение количества: "**);  
 *products*[p].changeCount(cmd.nextInt());  
 *products*[p].info();  
 done = **true**;  
 **break**;  
 }  
 }  
 System.***out***.println(done ? **"Выполнено"** : **"Не найден указанный товар: "** + temp);  
 **for** (**int** p = 0; p < *products*.**length**; p++) {  
 **if** (*products*[p].last())  
 *products*[p + 1] = **new** Product();  
 **break**;  
 }  
 **break**;  
  
 **case "заказ"**:  
 done = **false**;  
 *all*(*products*);  
 System.***out***.print(**"Введите Id товара: "**);  
 **int** oP = cmd.nextInt();  
  
 **for** (**int** p = 0; p < *products*.**length**; p++) {  
 **if** (*products*[p].getId() == oP) {  
 done = **true**;  
 System.***out***.print(**"Введите запрашиваемое количество: "**);  
 **int** oCount = cmd.nextInt();  
 **if** (*products*[p].getCount() >= oCount) {  
 System.***out***.print(**"Введите фамилию и имя покупателя: "**);  
 cmd.nextLine();  
 String oBuyer = cmd.nextLine();  
 **boolean** f = **false**;  
 **for** (**int** b = 0; b < *buyers*.**length**; b++) {  
 **if** (oBuyer.equals(*buyers*[b].**FIO**)) {  
 f = **true**;  
 **if** (*buyers*[b].getBlackList() == **true**) {  
 System.***out***.println(**"Покупатель находится в чёрном списке"**);  
 **break**;  
 }  
 **else**{  
 **for** (**int** o = 0; o < *orders*.**length**; o++) {  
 **if** (*orders*[o].last())  
 *orders*[o + 1] = **new** Order(*buyers*[b], *products*[p], oCount);  
 **break**;  
 }  
 *products*[p].changeCount((-1)\*oCount);  
 }  
 }  
 **if** (*buyers*[b].last()) **break**;  
 }  
 **if** (!f) {  
 **for** (**int** b = 0; b < *buyers*.**length**; b++) {  
 **if** (*buyers*[b].last())  
 *buyers*[b + 1] = **new** Buyer(oBuyer);  
 **for** (**int** o = 0; o < *orders*.**length**; o++) {  
 **if** (*orders*[o].last())  
 *orders*[o + 1] = **new** Order(*buyers*[b + 1], *products*[p], oCount);  
 **break**;  
 }  
 **break**;  
 }  
 }  
 **break**;  
 }  
 **else** {  
 System.***out***.printf(**"Указанного количества %d нет в наличии\n"**, oCount);  
 **break**;  
 }  
 }  
 **if** (*products*[p].last()) **break**;  
 }  
 System.***out***.println(done ? **""** : **"Не найден запрашиваемый товар: Id = "** + oP);  
 **break**;  
  
 **case "оплата"**:  
 done = **false**;  
 *all*(*orders*);  
 System.***out***.println(**"Введите № заказа: "**);  
 itemp = cmd.nextInt();  
 **for** (**int** i = 0; i < *orders*.**length**; i++) {  
 **if** (*orders*[i].getId() == itemp) {  
 *orders*[i].purchase();  
 *orders*[i].info();  
 done = **true**;  
 **break**;  
 }  
 **if** (*orders*[i].last()) **break**;  
 }  
 System.***out***.println(done ? **"Выполнено"** : **"Не найден указанный заказ: "** + itemp);  
 **break**;  
  
 **case "чёрный"**:  
 done = **false**;  
 *all*(*buyers*);  
 System.***out***.println(**"Введите Id покупателя: "**);  
 itemp = cmd.nextInt();  
 **for** (**int** i = 0; i < *buyers*.**length**; i++) {  
 **if** (*buyers*[i].getId() == itemp) {  
 *buyers*[i].inBlackList();  
 done = **true**;  
 **break**;  
 }  
 **if** (*buyers*[i].last()) **break**;  
 }  
 System.***out***.println(done ? **"Выполнено"** : **"Не найден указанный покупатель: "** + itemp);  
 cmd.nextLine();  
 **break**;  
  
 **case "белый"**:  
 done = **false**;  
 *all*(*buyers*);  
 System.***out***.println(**"Введите Id покупателя: "**);  
 itemp = cmd.nextInt();  
 **for** (**int** i = 0; i < *buyers*.**length**; i++) {  
 **if** (*buyers*[i].getId() == itemp) {  
 *buyers*[i].outBlackList();  
 done = **true**;  
 **break**;  
 }  
 **if** (*buyers*[i].last()) **break**;  
 }  
 System.***out***.println(done ? **"Выполнено"** : **"Не найден указанный покупатель: "** + itemp);  
 cmd.nextLine();  
 **break**;  
  
 **case "заказы"**:  
 *all*(*orders*);  
 cmd.nextLine();  
 **break**;  
  
 **case "товары"**:  
 *all*(*products*);  
 cmd.nextLine();  
 **break**;  
  
 **case "покупатели"**:  
 *all*(*buyers*);  
 cmd.nextLine();  
 **break**;  
  
 **default**:  
 System.***out***.println(**"Неизвестная команда"**);  
 }  
  
 }  
 }  
}

**package** com.company;  
  
**import** java.util.Scanner;  
  
**public class** Buyer {  
 **private static int** *Counter* = 0;  
 **private int Id**;  
 String **FIO**;  
 **private int Age**;  
 **private boolean BlackList** = **false**;  
  
 **public int** getId() {  
 **return Id**;  
 }  
 **public boolean** getBlackList() {  
 **return BlackList**;  
 }  
  
 Buyer(String fio) {  
 Scanner sc = **new** Scanner(System.***in***);  
 **FIO** = fio;  
 System.***out***.print(**"Укажите возраст покупателя: "**);  
 **Age** = sc.nextInt();  
 **Id** = ++*Counter*;  
 System.***out***.printf(**"Зарегистрирован покупатель >>> Id: %d\t Ф.И.О.: %s\t Возраст: %d\n"**, **Id**, **FIO**, **Age**);  
 }  
  
 Buyer(String fio, **int** age){  
 **FIO** = fio;  
 **Age** = age;  
 **Id** = ++*Counter*;  
 System.***out***.printf(**"Зарегистрирован покупатель >>> Id: %d\t Ф.И.О.: %s\t Возраст: %d\n"**, **Id**, **FIO**, **Age**);  
 }  
  
 **void** info(){  
 System.***out***.printf(**"Id: %d\t Ф.И.О.: %s\t Возраст: %d\t В ч.с.: %s\n"**, **Id**, **FIO**, **Age**, **BlackList** ? **"Да"** : **"Нет"**);  
 }  
  
 **boolean** last(){  
 **return Id** == *Counter* ? **true** : **false**;  
 }  
  
 **void** inBlackList(){  
 **BlackList** = **true**;  
 }  
  
 **void** outBlackList(){  
 **BlackList** = **false**;  
 }  
}

**package** com.company;  
  
**public class** Order {  
 **private static int** *Counter* = 0;  
 **private int Id**;  
 **private** Buyer **Man**;  
 **private** Product **Name**;  
 **private int Count**;  
 **private double Total**;  
 **private boolean Purchase** = **false**;  
  
 **public int** getId() {  
 **return Id**;  
 }  
  
 Order(Buyer man, Product name, **int** count) {  
 **Man** = man;  
 **Name** = name;  
 **Count** = count;  
 **Total** = count \* name.getPrice();  
 **Id** = ++*Counter*;  
 System.***out***.printf(  
 **"Добавлен заказ >>> №: %d\t Покупатель: %s\t Товар: %s\t Количество: %d\t Сумма: %.2f\t Оплачено: %s\n"**,  
 **Id**, **Man**.**FIO**, **Name**.getName(), **Count**, **Total**, **Purchase** ? **"да"** : **"нет"**);  
 }  
  
 **void** info(){  
 System.***out***.printf(  
 **"Заказ №: %d\t Покупатель: %s\t Товар: %s\t Количество: %d\t Сумма: %.2f\t Оплачено: %s\n"**,  
 **Id**, **Man**.**FIO**, **Name**.getName(), **Count**, **Total**, **Purchase** ? **"да"** : **"нет"**);  
 }  
  
 **boolean** last(){  
 **return Id** == *Counter* ? **true** : **false**;  
 }  
  
 **void** purchase(){  
 **Purchase** = **true**;  
 }  
}

**package** com.company;  
  
**import** java.util.Scanner;  
  
**public class** Product {  
 **private static int** *Counter* = 0; *// общее кол-во товаров* **private int Id**; *// номер текущего товара* **private** String **Name**; *// название* **private double Price**; *// цена* **private int Count**; *// количество* **public int** getId() {  
 **return Id**;  
 }  
 **public int** getCount() {  
 **return Count**;  
 }  
 **public** String getName() {  
 **return Name**;  
 }  
 **public double** getPrice() {  
 **return Price**;  
 }  
  
 **void** setCount(**int** count) {  
 **if** (count >= 0)  
 **Count** = count;  
 **else** System.***out***.println(**"Количество товара не может быть ниже 0"**);  
 }  
 **void** setPrice(**double** price) {  
 **if** (price >= 0)  
 **Price** = price;  
 **else** System.***out***.println(**"Стоимость товара не может быть ниже 0"**);  
 }  
  
 **void** changeCount(**int** count) {  
 **if** ((**Count** + count) >= 0)  
 **Count** += count;  
 **else** System.***out***.printf(**"Товар %s отсутствует в требуемом количестве. В наличии: %d\n"**, **Name**, **Count**);  
 }  
  
 Product() {  
 Scanner sc = **new** Scanner(System.***in***);  
 System.***out***.println(**"\nФормирование нового товара >>>"**);  
 System.***out***.print(**"Укажите наименование товара: "**);  
 **Name** = sc.nextLine();  
 System.***out***.print(**"Укажите стоимость товара: "**);  
 **Price** = sc.nextDouble();  
 System.***out***.print(**"Укажите количество товара: "**);  
 **Count** = sc.nextInt();  
 **Id** = ++*Counter*;  
 System.***out***.printf(**"Добавлен товар >>> Id: %d\t Наименование: %s\t Стоимость: %.2f\t Количество: %d\n"**, **Id**, **Name**, **Price**, **Count**);  
 }  
  
 Product(String name, **double** price, **int** count) {  
 **Name** = name.replaceAll(**" "**, **""**).length() > 0 ? name : **"unknown"**;  
 setPrice(price);  
 setCount(count);  
 **Id** = ++*Counter*;  
 System.***out***.printf(**"Добавлен товар >>> Id: %d\t Наименование: %s\t Стоимость: %.2f\t Количество: %d \n"**, **Id**, **Name**, **Price**, **Count**);  
 }  
  
 **boolean** last(){  
 **return Id** == *Counter* ? **true** : **false**;  
 }  
  
 **void** info(){  
 System.***out***.printf(**"Id: %d\t Наименование: %s\t Цена: %.2f\t Количество: %d \n"**, **Id**, **Name**, **Price**, **Count**);  
 }  
}

**Проверка правильности работы**:



**Вывод**: Программа добавляет информацию о Товаре. Клиент делает и оплачивает Заказ на Товары. Администратор регистрирует Продажу и может занести неплательщиков в «черный список». Программа работает корректно.