ПРОГРАММА 1

/\*  
Vorobyov Alexander Olegovich,  
1032193998, NFIbd-01-19  
-----------------------------  
11. Autobus содержащий следующие поля: название начального пункта;  
название конечного пункта; номер маршрута; время поездки  
 \*/  
  
 public class Trip {  
 private String StartPoint, FinalPoint; //переменные для назв. нач. и кон. пунктов  
 private int RouteNum; //переменные с номером маршрута  
 private double TravelTime, RouteLen, TicketPrice; //его длительностью, длиной, ценой билета.  
 public static int *n* = 6;  
  
 public Trip(String SP, String FP, int RN, double TT, double RL, double TP) { //конструктор1  
 this.StartPoint = SP;  
 this.FinalPoint = FP;  
 this.RouteNum = RN;  
 this.TravelTime = TT;  
 this.RouteLen = RL;  
 this.TicketPrice = TP;  
 }  
  
 public Trip() { //конструктор2  
 this.StartPoint = "Moscow";  
 this.FinalPoint = "Kazan";  
 this.RouteNum = 1015;  
 this.TravelTime = 12.15;  
 this.RouteLen = 820;  
 this.TicketPrice = 1703;  
 }  
  
 static void TimeDifference(Trip a1, Trip a2) { //разница в продолжит. поездок  
 double TD = Math.*abs*(a1.TravelTime - a2.TravelTime); //вычисляем модуль разницы  
 System.*out*.println("Difference in trip to " + a1.FinalPoint + " and to " + a2.FinalPoint + ": " + TD + " hour(s)");  
 }  
  
 static void ShortestTrip(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 //поиск самой короткой по времени поездки  
 double min = 1000; //переменная для минимума  
 double[] array; //массив с данными о длительности поездок  
 array = new double[*n*];  
 array[0] = a1.TravelTime; //присваиваем данные  
 array[1] = a2.TravelTime;  
 array[2] = a3.TravelTime;  
 array[3] = a4.TravelTime;  
 array[4] = a5.TravelTime;  
 array[5] = a6.TravelTime;  
 for(int i = 0; i < *n*; i++) { //через цикл находим кратчайшую поездку  
 if (array[i] < min) {  
 min = array[i];  
 }  
 }  
 System.*out*.println("The shortest trip lasts " + min + " hour(s)");  
 }  
  
 static void LongestTrip(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 //поиск самой длинной по времени поездки  
 double max = 0; //переменная для минимума  
 double[] array; //массив с данными о длительности поездок  
 array = new double[*n*];  
 array[0] = a1.TravelTime; //присваиваем данные  
 array[1] = a2.TravelTime;  
 array[2] = a3.TravelTime;  
 array[3] = a4.TravelTime;  
 array[4] = a5.TravelTime;  
 array[5] = a6.TravelTime;  
 for(int i = 0; i < *n*; i++) { //через цикл находим кратчайшую поездку  
 if (array[i] > max) {  
 max = array[i];  
 }  
 }  
 System.*out*.println("The longest trip lasts " + max + " hour(s)");  
 }  
  
 static void InfoAboutRoutes(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 //печать информации о направлении маршрута и его номера  
 String[] destination; //массив с пунктами назначения  
 destination = new String[*n*];  
 destination[0] = a1.FinalPoint; destination[1] = a2.FinalPoint; destination[2] = a3.FinalPoint;  
 destination[3] = a4.FinalPoint; destination[4] = a5.FinalPoint; destination[5] = a6.FinalPoint;  
  
 int[] numbers; //массив с номерами маршрутов  
 numbers = new int[*n*];  
 numbers[0] = a1.RouteNum; numbers[1] = a2.RouteNum; numbers[2] = a3.RouteNum;  
 numbers[3] = a4.RouteNum; numbers[4] = a5.RouteNum; numbers[5] = a6.RouteNum;  
  
 for (int j = 0; j < *n*; j++) { //печатаем по шаблону  
 System.*out*.println("Bus number " + numbers[j] + " goes to " + destination[j] + ".");  
 }  
 }  
  
 static void AverageSpeed(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 String[] destination; //массив с пунктами назначения  
 destination = new String[*n*]; // заполняем значения  
 destination[0] = a1.FinalPoint; destination[1] = a2.FinalPoint; destination[2] = a3.FinalPoint;  
 destination[3] = a4.FinalPoint; destination[4] = a5.FinalPoint; destination[5] = a6.FinalPoint;  
  
 double[] time; //массив с длительностью поездки  
 time = new double[*n*];  
 time[0] = a1.TravelTime; time[1] = a2.TravelTime; time[2] = a3.TravelTime;  
 time[3] = a4.TravelTime; time[4] = a5.TravelTime; time[5] = a6.TravelTime;  
  
 double[] len; //массив с длиной маршрута  
 len = new double[*n*];  
 len[0] = a1.RouteLen; len[1] = a2.RouteLen; len[2] = a3.RouteLen;  
 len[3] = a4.RouteLen; len[4] = a5.RouteLen; len[5] = a6.RouteLen;  
  
 for (int a = 0; a < *n*; a++) {  
 double tmp = len[a] / time[a]; //вычисление средней скорости  
 System.*out*.println("Average bus speed to " + destination[a] + ": " + Math.*ceil*(tmp) + " km/h");  
 }  
 }  
  
 static void LongestRoute(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 double[] len; //массив с длиной маршрута  
 len = new double[*n*];  
 len[0] = a1.RouteLen; len[1] = a2.RouteLen; len[2] = a3.RouteLen;  
 len[3] = a4.RouteLen; len[4] = a5.RouteLen; len[5] = a6.RouteLen;  
  
 int[] numbers; //массив с номерами маршрутов  
 numbers = new int[*n*];  
 numbers[0] = a1.RouteNum; numbers[1] = a2.RouteNum; numbers[2] = a3.RouteNum;  
 numbers[3] = a4.RouteNum; numbers[4] = a5.RouteNum; numbers[5] = a6.RouteNum;  
  
 double max = 0;  
 int num = 0;  
  
 for (int e = 0; e < *n*; e++) { //находим максимальное значение  
 if (len[e] > max) {  
 max = len[e];  
 num = numbers[e];  
 }  
 }  
  
 System.*out*.println("The longest route is by bus " + num + ": " + max + " km.");  
 }  
  
 static void ShortestRoute(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 double[] len; //массив с длиной маршрута  
 len = new double[*n*];  
 len[0] = a1.RouteLen; len[1] = a2.RouteLen; len[2] = a3.RouteLen;  
 len[3] = a4.RouteLen; len[4] = a5.RouteLen; len[5] = a6.RouteLen;  
  
 int[] numbers; //массив с номерами маршрутов  
 numbers = new int[*n*];  
 numbers[0] = a1.RouteNum; numbers[1] = a2.RouteNum; numbers[2] = a3.RouteNum;  
 numbers[3] = a4.RouteNum; numbers[4] = a5.RouteNum; numbers[5] = a6.RouteNum;  
  
 double min = 1000;  
 int num = 0;  
  
 for (int e = 0; e < *n*; e++) { //находим максимальное значение  
 if (len[e] < min) {  
 min = len[e];  
 num = numbers[e];  
 }  
 }  
  
 System.*out*.println("The shortest route is by bus " + num + ": " + min + " km.");  
 }  
  
 static void GetPrice(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 double[] price; //массив с ценами на поездки  
 price = new double[*n*]; //заполняем значения  
 price[0] = a1.TicketPrice; price[1] = a2.TicketPrice; price[2] = a3.TicketPrice;  
 price[3] = a4.TicketPrice; price[4] = a5.TicketPrice; price[5] = a6.TicketPrice;  
  
 String[] destination; //массив с пунктами назначения  
 destination = new String[*n*]; // заполняем значения  
 destination[0] = a1.FinalPoint; destination[1] = a2.FinalPoint; destination[2] = a3.FinalPoint;  
 destination[3] = a4.FinalPoint; destination[4] = a5.FinalPoint; destination[5] = a6.FinalPoint;  
  
 for (int l = 0; l < *n*; l++) {  
 System.*out*.println("A bus ticket to " + destination[l] + " costs " + price[l] + " Rub.");  
 }  
 }  
  
 static void StudentSale30(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 double[] price; //массив с ценами на поездки  
 price = new double[*n*]; //заполняем значения  
 price[0] = a1.TicketPrice; price[1] = a2.TicketPrice; price[2] = a3.TicketPrice;  
 price[3] = a4.TicketPrice; price[4] = a5.TicketPrice; price[5] = a6.TicketPrice;  
  
 String[] destination; //массив с пунктами назначения  
 destination = new String[*n*]; // заполняем значения  
 destination[0] = a1.FinalPoint; destination[1] = a2.FinalPoint; destination[2] = a3.FinalPoint;  
 destination[3] = a4.FinalPoint; destination[4] = a5.FinalPoint; destination[5] = a6.FinalPoint;  
  
 for (int l = 0; l < *n*; l++) {  
 double saleprice = price[l] \* 0.7;  
 System.*out*.println("Discount ticket price to " + destination[l] + " for students: " + Math.*ceil*(saleprice) + " Rub.");  
 }  
 }  
  
 static void ChildTariff50(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 double[] price; //массив с ценами на поездки  
 price = new double[*n*]; //заполняем значения  
 price[0] = a1.TicketPrice; price[1] = a2.TicketPrice; price[2] = a3.TicketPrice;  
 price[3] = a4.TicketPrice; price[4] = a5.TicketPrice; price[5] = a6.TicketPrice;  
  
 String[] destination; //массив с пунктами назначения  
 destination = new String[*n*]; // заполняем значения  
 destination[0] = a1.FinalPoint; destination[1] = a2.FinalPoint; destination[2] = a3.FinalPoint;  
 destination[3] = a4.FinalPoint; destination[4] = a5.FinalPoint; destination[5] = a6.FinalPoint;  
  
 for (int l = 0; l < *n*; l++) {  
 double saleprice = price[l] \* 0.5;  
 System.*out*.println("Discount ticket price to " + destination[l] + " for children: " + Math.*ceil*(saleprice) + " Rub.");  
 }  
 }  
  
 static void CheapestTrip(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 double[] price; //массив с ценами на поездки  
 price = new double[*n*]; //заполняем значения  
 price[0] = a1.TicketPrice; price[1] = a2.TicketPrice; price[2] = a3.TicketPrice;  
 price[3] = a4.TicketPrice; price[4] = a5.TicketPrice; price[5] = a6.TicketPrice;  
  
 int[] numbers; //массив с номерами маршрутов  
 numbers = new int[*n*];  
 numbers[0] = a1.RouteNum; numbers[1] = a2.RouteNum; numbers[2] = a3.RouteNum;  
 numbers[3] = a4.RouteNum; numbers[4] = a5.RouteNum; numbers[5] = a6.RouteNum;  
  
 double min = 10000;  
 int num = 0;  
  
 for (int e = 0; e < *n*; e++) { //находим минимальную стоимость  
 if (price[e] < min) {  
 min = price[e];  
 num = numbers[e];  
 }  
 }  
  
 System.*out*.println("The cheapest trip is by bus " + num + ": " + min + " Rub.");  
 }  
  
 static void MostExpensiveTrip(Trip a1, Trip a2, Trip a3, Trip a4, Trip a5, Trip a6) {  
 double[] price; //массив с ценами на поездки  
 price = new double[*n*]; //заполняем значения  
 price[0] = a1.TicketPrice; price[1] = a2.TicketPrice; price[2] = a3.TicketPrice;  
 price[3] = a4.TicketPrice; price[4] = a5.TicketPrice; price[5] = a6.TicketPrice;  
  
 int[] numbers; //массив с номерами маршрутов  
 numbers = new int[*n*];  
 numbers[0] = a1.RouteNum; numbers[1] = a2.RouteNum; numbers[2] = a3.RouteNum;  
 numbers[3] = a4.RouteNum; numbers[4] = a5.RouteNum; numbers[5] = a6.RouteNum;  
  
 double max = 0;  
 int num = 0;  
  
 for (int e = 0; e < *n*; e++) { //находим минимальную стоимость  
 if (price[e] > max) {  
 max = price[e];  
 num = numbers[e];  
 }  
 }  
  
 System.*out*.println("The most expensive trip is by bus " + num + ": " + max + " Rub.");  
 }  
  
 public static void main(String[] args) {  
 //заполнение данных об автобусах  
 Trip trip1 = new Trip("Moscow", "Kiev", 560, 10.9, 860, 3405);  
 Trip trip2 = new Trip("Moscow", "Minsk", 915, 7.9, 680, 1589);  
 Trip trip3 = new Trip("Moscow", "Sevastopol", 440, 20.5, 1900, 3105);  
 Trip trip4 = new Trip("Moscow", "Saint-Petersburg", 685, 7.5, 640, 1163);  
 Trip trip5 = new Trip("Moscow", "Arkhangelsk", 590, 16.1, 1200, 2400);  
 Trip trip6 = new Trip();  
  
 //вызов методов(12)  
 *TimeDifference*(trip1, trip2); //разница во времени между поездками  
 *ShortestTrip*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *LongestTrip*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *InfoAboutRoutes*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *AverageSpeed*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *LongestRoute*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *ShortestRoute*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *GetPrice*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *StudentSale30*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *ChildTariff50*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *CheapestTrip*(trip1, trip2, trip3, trip4, trip5, trip6);  
 *MostExpensiveTrip*(trip1, trip2, trip3, trip4, trip5, trip6);  
 }  
}

РЕЗУЛЬТАТ 1

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

ПРОГРАММА 2

/\*  
Vorobyov Alexander Olegovich,  
1032193998, NFIbd-01-19  
-----------------------------  
36. Table содержащий следующие поля: страна производитель стола;  
высота стола; материал, из которого изготовлен; цена.  
 \*/  
import java.util.Scanner;  
public class Table {  
 private String Country, Material, Colour; //переменные  
 private double Height, Price;  
 static public int *n* = 6;  
  
 public Table(String C, String M, double H, double P, String Cr) { //конструктор1  
 this.Country = C;  
 this.Material = M;  
 this.Height = H;  
 this.Price = P;  
 this.Colour = Cr;  
 }  
  
 public Table() { //конструктор2  
 this.Country = "Russia";  
 this.Material = "Chipboard";  
 this.Height = 75;  
 this.Price = 16190;  
 this.Colour = "Gray";  
 }  
  
 static void FindDomestic(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) { //поиск столов отечественного производства  
 String[] array;  
 array = new String[*n*]; //массив с данными о стране производителя  
 array[0] = t1.Country; array[1] = t2.Country; array[2] = t3.Country; //присваиваем данные  
 array[3] = t4.Country; array[4] = t5.Country; array[5] = t6.Country;  
  
 for(int i = 0; i < *n*; i++) { // находим совпадения со страной производства Russia  
 if (array[i] == "Russia") {  
 System.*out*.println("Table " + (i+1) + " is domestic");  
 }  
 }  
 }  
  
 static void FindForeign(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) { //поиск столов зарубежного производства  
 String[] array;  
 array = new String[*n*]; //массив с данными о стране производителя  
 array[0] = t1.Country; array[1] = t2.Country; array[2] = t3.Country; //присваиваем данные  
 array[3] = t4.Country; array[4] = t5.Country; array[5] = t6.Country;  
  
 for(int i = 0; i < *n*; i++) { // находим совпадения со страной производства Russia  
 if (array[i] != "Russia") {  
 System.*out*.println("Table " + (i+1) + " is foreign");  
 }  
 }  
 }  
  
 static void FindTallest(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) { //поиск самого высокого стола  
 double max = 0; //переменная для записи максимума  
 double[] array2; //массив с данными о высоте столов  
 array2 = new double[*n*];  
 array2[0] = t1.Height; array2[1] = t2.Height; array2[2] = t3.Height; //присваиваем данные  
 array2[3] = t4.Height; array2[4] = t5.Height; array2[5] = t6.Height;  
  
 for (int j = 0; j < *n*; j++) { // находим максимальное значение параметра height  
 if (array2[j] > max) {  
 max = array2[j];  
 }  
 }  
 System.*out*.println("The tallest table is " + max + " cm");  
 }  
  
 static void FindLowest(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) { //поиск самого высокого стола  
 double min = 1000; //переменная для записи максимума  
 double[] array2; //массив с данными о высоте столов  
 array2 = new double[*n*];  
 array2[0] = t1.Height; array2[1] = t2.Height; array2[2] = t3.Height; //присваиваем данные  
 array2[3] = t4.Height; array2[4] = t5.Height; array2[5] = t6.Height;  
  
 for (int j = 0; j < *n*; j++) { // находим максимальное значение параметра height  
 if (array2[j] < min) {  
 min = array2[j];  
 }  
 }  
 System.*out*.println("The lowest table is " + min + " cm");  
 }  
  
 static void FindCheapest(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) { //поиск самого дешевого стола  
 double min = 100000; //переменная для записи минимума  
 double[] price; //массив с данными о цене столов  
 price = new double[*n*];  
 price[0] = t1.Price; price[1] = t2.Price; price[2] = t3.Price; //присваиваем данные  
 price[3] = t4.Price; price[4] = t5.Price; price[5] = t6.Price;  
  
 for (int u = 0; u < *n*; u++) { //находим наименьшее значение параметра price  
 if (price[u] < min) {  
 min = price[u];  
 }  
 }  
 System.*out*.println("The cheapest table costs " + min + " Rub");  
 }  
  
 static void FindMostExpensive(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) { //поиск самого дешевого стола  
 double max = 0; //переменная для записи минимума  
 double[] price; //массив с данными о цене столов  
 price = new double[*n*];  
 price[0] = t1.Price; price[1] = t2.Price; price[2] = t3.Price; //присваиваем данные  
 price[3] = t4.Price; price[4] = t5.Price; price[5] = t6.Price;  
  
 for (int u = 0; u < *n*; u++) { //находим наименьшее значение параметра price  
 if (price[u] > max) {  
 max = price[u];  
 }  
 }  
 System.*out*.println("The most expensive table costs " + max + " Rub");  
 }  
  
 static void TemplatePrinting(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) { //печать товаров по шаблону  
 String[] material; //массив с данными о материалах стола  
 material = new String[*n*];  
 material[0] = t1.Material; material[1] = t2.Material; material[2] = t3.Material; //присваиваем данные  
 material[3] = t4.Material; material[4] = t5.Material; material[5] = t6.Material;  
  
 double[] array2; //массив с данными о высоте столов  
 array2 = new double[*n*];  
 array2[0] = t1.Height; array2[1] = t2.Height; array2[2] = t3.Height; //присваиваем данные  
 array2[3] = t4.Height; array2[4] = t5.Height; array2[5] = t6.Height;  
  
 String[] array;  
 array = new String[*n*]; //массив с данными о стране производителя  
 array[0] = t1.Country; array[1] = t2.Country; array[2] = t3.Country; //присваиваем данные  
 array[3] = t4.Country; array[4] = t5.Country; array[5] = t6.Country;  
  
 double[] price; //массив с данными о цене столов  
 price = new double[*n*];  
 price[0] = t1.Price; price[1] = t2.Price; price[2] = t3.Price; //присваиваем данные  
 price[3] = t4.Price; price[4] = t5.Price; price[5] = t6.Price;  
  
 for (int a = 0; a < *n*; a++) {  
 System.*out*.println(material[a] + " table " + array2[a] + "cm from " + array[a] + " costs " + price[a] + " Rub.");  
 }  
 }  
  
 static void SameColour(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) {  
 String[] colour;  
 colour = new String[*n*];  
 colour[0] = t1.Colour; colour[1] = t2.Colour; colour[2] = t3.Colour;  
 colour[3] = t4.Colour; colour[4] = t5.Colour; colour[5] = t6.Colour;  
  
 for (int o = 0; o < *n*-1; o++) {  
 for (int e = o+1; e < *n*; e++) {  
 if ((colour[o] == colour[e]) & (o != e)) {  
 System.*out*.println("Table " + (o+1) + " and " + (e+1) + " have the same colour: " + colour[o] + ".");  
 }  
 }  
 }  
 }  
  
 static void SameCountry(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) {  
 String[] country;  
 country = new String[*n*]; //массив с данными о стране производителя  
 country[0] = t1.Country; country[1] = t2.Country; country[2] = t3.Country; //присваиваем данные  
 country[3] = t4.Country; country[4] = t5.Country; country[5] = t6.Country;  
  
 for (int o = 0; o < *n*-1; o++) {  
 for (int e = o+1; e < *n*; e++) {  
 if ((country[o] == country[e]) & (o != e)) {  
 System.*out*.println("Table " + (o+1) + " and " + (e+1) + " are from " + country[o] + ".");  
 }  
 }  
 }  
 }  
  
 static void WhichHigher(Table t1, Table t2) {  
 if (t1.Height > t2.Height) {  
 System.*out*.println("First table is higher than the second.");  
 } else {  
 System.*out*.println("Second table is higher than the first.");  
 }  
 }  
  
 static void PriceDifference(Table t1, Table t2) {  
 if (t1.Price > t2.Price) {  
 double difp = t1.Price - t2.Price;  
 System.*out*.println("The first table is " + difp + " Rub more expensive than the second.");  
 } else {  
 double difm = t2.Price - t1.Price;  
 System.*out*.println("The first table is " + difm + " Rub cheaper than the second.");  
 }  
 }  
  
 static void SalePrice20(Table t1, Table t2, Table t3, Table t4, Table t5, Table t6) {  
 double[] price; //массив с данными о цене столов  
 price = new double[*n*];  
 price[0] = t1.Price; price[1] = t2.Price; price[2] = t3.Price; //присваиваем данные  
 price[3] = t4.Price; price[4] = t5.Price; price[5] = t6.Price;  
  
 for (int y = 0; y < *n*; y++) {  
 double sprice = price[y] \* 0.8;  
 System.*out*.println("20% discount price for table " + (y+1) + ": " + Math.*ceil*(sprice) + " Rub.");  
 }  
 }  
  
 public static void main(String[] args) {  
 //заполнение данных о столах  
 Table table1 = new Table("Russia", "Chipboard", 75.1, 7490, "White");  
 Table table2 = new Table("China", "Aluminum", 47, 6999, "Black");  
 Table table3 = new Table("Sweden", "Wood", 70, 14590, "White");  
 Table table4 = new Table("China", "Glass", 65.5, 8999, "Black");  
 Table table5 = new Table("Sweden", "Rock", 84, 19990, "Gray");  
 Table table6 = new Table();  
  
 *FindDomestic*(table1, table2, table3, table4, table5, table6); //Стол(ы) отечественного производства  
 *FindForeign*(table1, table2, table3, table4, table5, table6); //Стол(ы) зарубежного производства  
 *FindTallest*(table1, table2, table3, table4, table5, table6); //самый высокий стол  
 *FindLowest*(table1, table2, table3, table4, table5, table6); //самый низкий стол  
 *FindCheapest*(table1, table2, table3, table4, table5, table6); //самый дешевый стол  
 *FindMostExpensive*(table1, table2, table3, table4, table5, table6); //самый дорогой стол  
 *TemplatePrinting*(table1, table2, table3, table4, table5, table6); //Шаблонная печать  
 *SameColour*(table1, table2, table3, table4, table5, table6); //столы с одинаковыми цветами  
 *SameCountry*(table1, table2, table3, table4, table5, table6); //столы произведенные в одной стране  
 *WhichHigher*(table3, table4); //какой из двух столов выше  
 *PriceDifference*(table5, table6); //разница в цене между двумя столами  
 *SalePrice20*(table1, table2, table3, table4, table5, table6); //стоимость столов со скидкой 20%  
 }  
}

РЕЗУЛЬТАТ 2

Изображение выглядит как текст

Автоматически созданное описание