Lab10.

Task1. Трое студентов, пятикурсник, третьекурсник и первокурсник, живут в одной комнате общежития и любят участвовать в соревнованиях по программированию по правилам ACM.

Code:

import java.io.\*;  
import java.util.\*;  
public class ACM {  
 static int calculatePenalty(int[] times) {  
 int totalPenalty = 0;  
 int penaltyTime = 0;  
 for (int time : times) {  
 totalPenalty += penaltyTime;  
 penaltyTime += time;  
 }  
 return totalPenalty;  
 }  
 static int winner(int N, int[] taskTimes) {  
 int[] fifthYearTimes = Arrays.*copyOf*(taskTimes, N);  
 int[] thirdYearTimes = new int[N];  
 int[] freshmanTimes = new int[N];  
  
 // Reverse the task times for the third-year student  
 for (int i = 0; i < N; i++) {  
 thirdYearTimes[i] = taskTimes[N - 1 - i];  
 }  
  
 // Sort the task times for the freshman student  
 System.*arraycopy*(taskTimes, 0, freshmanTimes, 0, N);  
 Arrays.*sort*(freshmanTimes);  
  
 int fifthYearPenalty = *calculatePenalty*(fifthYearTimes);  
 int thirdYearPenalty = *calculatePenalty*(thirdYearTimes);  
 int freshmanPenalty = *calculatePenalty*(freshmanTimes);  
 if (fifthYearPenalty < thirdYearPenalty && fifthYearPenalty < freshmanPenalty) {  
 return 5;  
 } else if (thirdYearPenalty < freshmanPenalty) {  
 return 3;  
 } else {  
 return 1;  
 }  
 }  
 public static void main(String[] args) {  
 try {  
 File inputFile = new File("input.txt");  
 Scanner scanner = new Scanner(inputFile);  
 int N = scanner.nextInt();  
 int[] taskTimes = new int[N];  
 for (int i = 0; i < N; i++) {  
 taskTimes[i] = scanner.nextInt();  
 }  
 scanner.close();  
 int winnerIndex = *winner*(N, taskTimes);  
 File outputFile = new File("output.txt");  
 FileWriter writer = new FileWriter(outputFile);  
 writer.write(Integer.*toString*(winnerIndex));  
 writer.close();  
 } catch (FileNotFoundException e) {  
 System.*out*.println("Input file not found.");  
 e.printStackTrace();  
 } catch (IOException e) {  
 System.*out*.println("Error writing to output file.");  
 e.printStackTrace();  
 }  
 }  
}

Task2. Паша придумал новые технологии нарезки продуктов для фирмы «Нарежем всё!!!», за что фирма выплатила ему премию. На неё он хочет покрасить стены в своей комнате. Для этого ему необходимо узнать, сколько банок краски нужно купить. Его комната имеет длину a, ширину b и высоту h. В его комнате есть дверной проем – шириной w и высотой q, а также окно шириной s и высотой p. Известно, что одной банки хватает на k квадратных метров площади окрашиваемой поверхности. Напишите программу, которая сможет решить эту задачу

Code:

import java.util.Scanner;  
public class Area {  
 public static void main(String[] args){  
 Scanner scanner = new Scanner(System.*in*);  
 System.*out*.println("Enter three elements:");  
 int[] abh = new int[3];  
 for (int i = 0; i < 3; i++) {  
 abh[i] = scanner.nextInt();  
 }  
 System.*out*.println("Enter the elements:");  
 int[] wq=new int[2];  
 for (int i = 0; i < 2; i++) {  
 wq[i] = scanner.nextInt();  
 }  
 System.*out*.println("Enter the elements:");  
 int[] sp=new int[2];  
 for (int i = 0; i < 2; i++) {  
 sp[i] = scanner.nextInt();  
 }  
 System.*out*.println("Enter the k:");  
 int k=scanner.nextInt();  
  
 int area= ((abh[0]\*abh[2])+(abh[1]\*abh[2]))\*2;  
 int area\_door=wq[0]\*wq[1];  
 int area\_window=sp[0]\*sp[1];  
 int s=area-(area\_door+area\_window);  
 int banks=s/k;  
 System.*out*.print("Банки нужные для покраски дома: "+banks);  
  
 }  
}

Task3. Премии оказалось достаточно не только для покраски стен, но и обновления домашней техники. Анна, жена Паши, считает, что деньги, сэкономленные на скидке – всё равно, что заработанные. Анне нужно купить холодильник, микроволновку и телевизор. Составьте программу, которая выведет такой набор из этих трёх товаров, что скидка в тенге будет наибольшей. В случае одинакового размера скидки, покупается товар с наименьшей ценой.

Code:

Product.java

class Product {  
 int id;  
 int price;  
 int discount;  
 Product(int id, int price, int discount) {  
 this.id = id;  
 this.price = price;  
 this.discount = discount;  
 }  
 int getDiscountedPrice() {  
 return price - (price \* discount / 100);  
 }  
}

Main.java

import java.util.Scanner;  
public class Main {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
 int L = scanner.nextInt(); // Number of refrigerators  
 int M = scanner.nextInt(); // Number of microwaves  
 int N = scanner.nextInt(); // Number of televisions  
 Product[] refrigerators = *readProducts*(scanner, L, 1);  
 Product[] microwaves = *readProducts*(scanner, M, L + 1);  
 Product[] televisions = *readProducts*(scanner, N, L + M + 1);  
 // Find the most profitable combination  
 int maxDiscount = 0;  
 int fridgeId = 0;  
 int microwaveId = 0;  
 int tvId = 0;  
 for (int i = 0; i < L; i++) {  
 for (int j = 0; j < M; j++) {  
 for (int k = 0; k < N; k++) {  
 int discount = refrigerators[i].getDiscountedPrice() + microwaves[j].getDiscountedPrice() + televisions[k].getDiscountedPrice();  
 if (discount > maxDiscount) {  
 maxDiscount = discount;  
 fridgeId = refrigerators[i].id;  
 microwaveId = microwaves[j].id;  
 tvId = televisions[k].id;  
 } else if (discount == maxDiscount) {  
 if (refrigerators[i].price + microwaves[j].price + televisions[k].price <  
 refrigerators[fridgeId - 1].price + microwaves[microwaveId - L - 1].price + televisions[tvId - L - M - 1].price) {  
 fridgeId = refrigerators[i].id;  
 microwaveId = microwaves[j].id;  
 tvId = televisions[k].id;  
 }  
 }  
 }  
 }  
 }  
  
 // Output result  
 System.*out*.println(fridgeId + " " + microwaveId + " " + tvId);  
  
 scanner.close();  
 }  
 private static Product[] readProducts(Scanner scanner, int count, int startId) {  
 Product[] products = new Product[count];  
 for (int i = 0; i < count; i++) {  
 int price = scanner.nextInt();  
 int discount = scanner.nextInt();  
 products[i] = new Product(startId + i, price, discount);  
 }  
 return products;  
 }  
}

Tasks from habr.com

1. Removing from array

1) public class removing {  
 public static void main(String[] args) {  
 int arr[] = {1, 2, 8, 6, 3, 9, 7, 5, 9, 7};  
 int value = 6;  
  
 for (int i = 0; i < arr.length; i++) {  
 if (arr[i] == value) {  
 continue;  
 }  
 System.*out*.print(arr[i] + " ");  
 }  
 }  
}

2) import java.util.Arrays;  
public class remove {  
 public static void main(String[] args) {  
 int[] arr = {1, 2, 8, 6, 3, 9, 7, 5, 9, 7};  
 int valueToRemove = 6;  
 int count = 0;  
 for (int i = 0; i < arr.length; i++) {  
 if (arr[i] == valueToRemove) {  
 count++;  
 }  
 }  
 int[] newArr = new int[arr.length - count];  
 int index = 0;  
 for (int i = 0; i < arr.length; i++) {  
 if (arr[i] != valueToRemove) {  
 newArr[index] = arr[i];  
 index++;  
 }  
 }  
 System.*out*.println("Array after removing " + valueToRemove + ": " + Arrays.*toString*(newArr));  
 }  
}

1. Генерация случайного элемента с весом

public class getRandom {  
 private int[] extendedValues; // значения  
 public getRandom(int[] values, int[] weights) {  
 // Сумма длин всех отрезков  
 int sum = 0;  
 for (int weight : weights) {  
 sum += weight;  
 }  
 extendedValues = new int[sum];  
 int cursor = 0;  
 for (int i = 0; i < weights.length; i++) {  
 for (int j = 0; j < weights[i]; j++) {  
 extendedValues[cursor++] = values[i];  
 }  
 }  
 }  
 public int getRandom() {  
 int random = (int) (Math.*random*() \* extendedValues.length);  
 return extendedValues[random];  
 }  
 public static void main(String[] args) {  
 int[] values = {20, 55, 36, 14, 87};  
 int[] weights = {20, 56, 84, 35, 96};  
 getRandom getRandomInstance = new getRandom(values, weights);  
 // Generate and print some random values  
 for (int i = 0; i < 5; i++) {  
 int randomValue = getRandomInstance.getRandom();  
 System.*out*.println("Random Value: " + randomValue);  
 }  
 }  
}

3. Найти корень уравнения

public class root {  
 // вспомогательный метод  
 public static double func(double x){  
 return Math.*cos*(Math.*pow*(x, 5)) + Math.*pow*(x, 4) - 345.3 \* x - 23;  
 }  
 // решить уравнение  
 public static double solve(double start, double end){  
 if(end - start <= 0.001){  
 return start;  
 }  
 double x = start + (end - start) / 2;  
 if(*func*(start) \* *func*(x) > 0){  
 return *solve*(x, end);  
 } else {  
 return *solve*(start, x);  
 }  
 }  
 public static void main(String[] args) {  
 System.*out*.println(*solve*(0, 10));  
 }  
}