*ПРИЛОЖЕНИЕ Ж*

|  |  |
| --- | --- |
| Класс | Листинг |
| Main | import java.util.ArrayList; import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.*in*);  System.*out*.println("Версия программы: 1\n" +  "Функции программы: работают\n" +  "Данные о разработчиках: Степанов,Красуцкий\n" +  "©\uFE0F все права защищены\n");  System.*out*.print("Введи начало работы: ");  String beginWorkingTime = sc.nextLine();  System.*out*.print("Введи конец работы: ");  String endWorkingTime = sc.nextLine();  System.*out*.print("Введи отрезки работы: ");  int consultationTime = Integer.*parseInt*(sc.nextLine());  System.*out*.print("Введи сколько занятых часов работы: ");  int n = Integer.*parseInt*(sc.nextLine());   String[] startTimes = new String[n];  int[] durations = new int[n];   for (int i = 0; i < n; i++) {  System.*out*.print("Введи начало занятой работы: ");  startTimes[i] = sc.next();  System.*out*.print("Введи продолжительность работы: ");  durations[i] = Integer.*parseInt*(sc.next());  }   ArrayList<String> freeIntervals = Calculations.*availablePeriods*(startTimes, durations, consultationTime, beginWorkingTime, endWorkingTime);   for (String interval : freeIntervals) {  System.*out*.println(interval);  }  } } |
| Calculations | import java.util.ArrayList;  public class Calculations {  static ArrayList<String> availablePeriods(String[] startTimes, int[] prodolzh, int Otrezok, String beginWorking, String endWorking) {  ArrayList<int[]> busySlots = new ArrayList<>();   int workingStart = *timeToMinutes*(beginWorking);  int workingEnd = *timeToMinutes*(endWorking);   for (int i = 0; i < startTimes.length; i++) {  int start = *timeToMinutes*(startTimes[i]);  int end = start + prodolzh[i];  busySlots.add(new int[]{start, end});  }   ArrayList<String> freeIntervals = new ArrayList<>();   for (int[] slot : busySlots) {  int start = slot[0];  int end = slot[1];   if (workingStart + Otrezok <= start) {  *addFreeInterval*(freeIntervals, workingStart, start - Otrezok);  }  workingStart = Math.*max*(workingStart, end);  }    if (workingStart + Otrezok <= workingEnd) {  *addFreeInterval*(freeIntervals, workingStart, workingEnd);  }   return freeIntervals;  }   static void addFreeInterval(ArrayList<String> freeIntervals, int start, int end) {  for (int i = start; i + 30 <= end; i += 30) {  freeIntervals.add(*minutesToTime*(i) + "\t-\t" + *minutesToTime*(i + 30));  }  }   static int timeToMinutes(String time) {  String[] parts = time.split(":");  return Integer.*parseInt*(parts[0]) \* 60 + Integer.*parseInt*(parts[1]);  }   static String minutesToTime(int minutes) {  int hours = minutes / 60;  int mins = minutes % 60;  if (mins == 0) {  return hours + ":00";  } else {  return hours + ":" + mins;  }  } } |
| Test | import org.junit.jupiter.api.Test; import java.util.ArrayList; import static org.junit.jupiter.api.Assertions.\*; class CalculationsTest {  @Test  public void AvailablePeriods() {  String[] startTimes = {"9:00", "10:30", "12:00"};  int[] prodolzh = {60, 30, 90};  int Otrezok = 30;  String beginWorking = "9:00";  String endWorking = "13:00";  ArrayList<String> expected = new ArrayList<>();  expected.add("11:00\t-\t11:30");    ArrayList<String> actual = Calculations.*availablePeriods*(startTimes, prodolzh, Otrezok, beginWorking, endWorking);  *assertEquals*(expected, actual);  }    @Test  public void TimeToMinutes() {  *assertEquals*(0, Calculations.*timeToMinutes*("0:00"));  *assertEquals*(60, Calculations.*timeToMinutes*("1:00"));  *assertEquals*(90, Calculations.*timeToMinutes*("1:30"));  *assertEquals*(120, Calculations.*timeToMinutes*("2:00"));  *assertEquals*(150, Calculations.*timeToMinutes*("2:30"));  *assertEquals*(180, Calculations.*timeToMinutes*("3:00"));  *assertEquals*(240, Calculations.*timeToMinutes*("4:00"));  }  @Test  public void MinutesToTime() {  *assertEquals*("0:00", Calculations.*minutesToTime*(0));  *assertEquals*("1:00", Calculations.*minutesToTime*(60));  *assertEquals*("1:30", Calculations.*minutesToTime*(90));  *assertEquals*("2:00", Calculations.*minutesToTime*(120));  *assertEquals*("2:30", Calculations.*minutesToTime*(150));  *assertEquals*("3:00", Calculations.*minutesToTime*(180));  *assertEquals*("4:00", Calculations.*minutesToTime*(240));  } } |