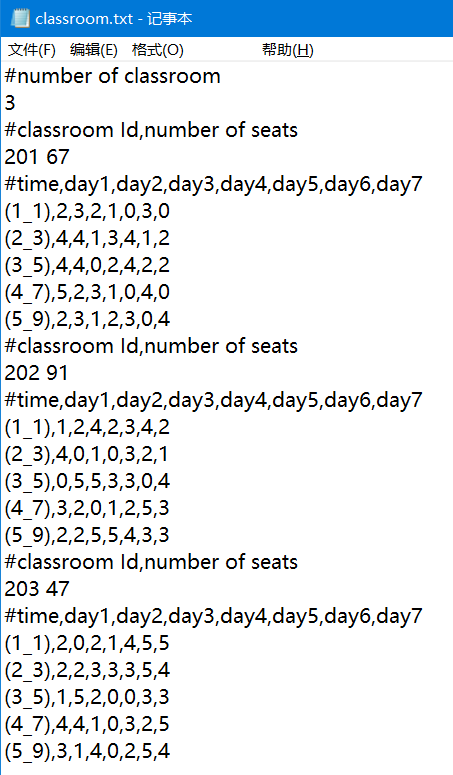
**Exercise1:**

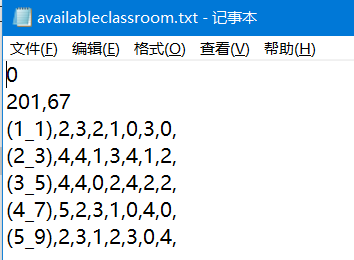
**Main:**

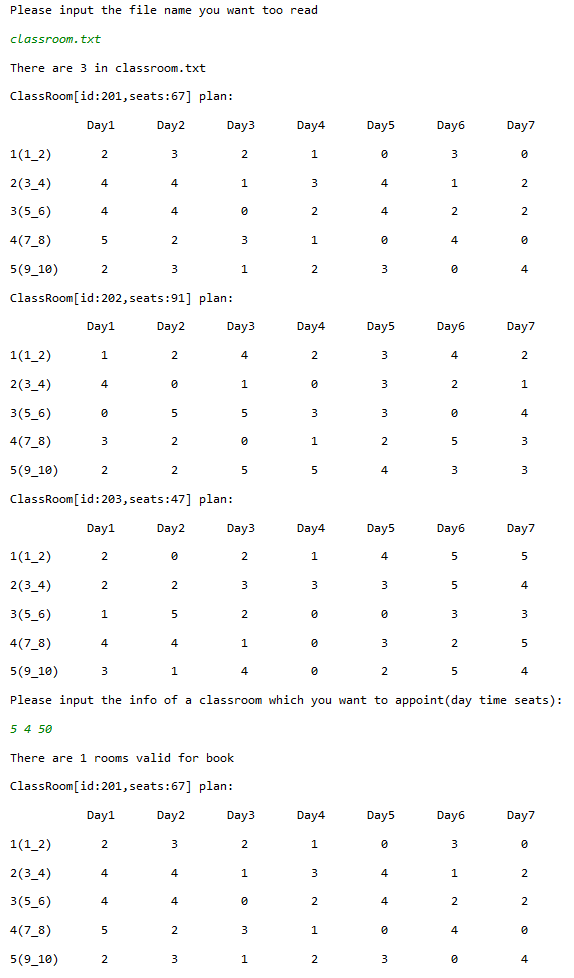
**import java.io.\*;  
import java.util.ArrayList;  
import java.util.Scanner;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Exercise1 {  
 public static ArrayList<ClassRoom> readFromFile(String fileName) throws FileNotFoundException {  
 ArrayList<ClassRoom> clsrms=new ArrayList<ClassRoom>();  
 File file1=new File("C:\\Users\\ChuYi\\Documents\\IDEA\_Workspace\\Basis of Programming Development\\src\\week12",fileName);  
 Scanner read=new Scanner(file1);  
 read.nextLine();  
 int number=read.nextInt();  
 read.nextLine();  
 for(int i=0;i<number;i++){  
 clsrms.add(new ClassRoom());  
 read.nextLine();//忽略第一行介绍的内容  
 clsrms.get(i).id=read.nextInt();  
 clsrms.get(i).seatsNumber=read.nextInt();//在从text文件中用nextInt的时候，数字之间必须用回车或者空格断开（不能用,断开），否则Scanner.mismatch  
 read.nextLine();  
 read.nextLine();  
 String[] arrangementRow=new String[5];  
 for(int j=0;j<5;j++) {  
 arrangementRow[j]=read.nextLine();  
 }//读取带(1\_1), 的arrangement  
 String[] arrangementRow2=new String[5];  
 for(int j=0;j<5;j++){  
 Pattern pTemp=Pattern.compile("\\(.\*?\\,");  
 Matcher mTemp=pTemp.matcher(arrangementRow[j]);  
 arrangementRow2[j]=mTemp.replaceAll("");  
 }//删除掉(1\_1),  
 for(int j=0;j<5;j++){  
 for(int k=0;k<7;k++){  
 clsrms.get(i).TheArragementOfTheClassRoom[j][k]=Integer.parseInt(arrangementRow2[j].split(",")[k]);  
 }  
 }//将2,3,2,1,0,3,0写入ClassRoom类的Arrangement中  
 }  
 return clsrms;  
 }  
  
 public static void printRoom(ArrayList<ClassRoom> clsrms){  
 for(int i=0;i<clsrms.size();i++){  
 clsrms.get(i).printRoom();  
 }  
 }  
  
 public static ArrayList<ClassRoom> okRooms(ArrayList<ClassRoom> clsrms,int day,int time,int seats){  
 ArrayList<ClassRoom> okRooms=new ArrayList<>();  
 for(int i=0;i<clsrms.size();i++){  
 if(clsrms.get(i).isValidforExam(day,time,seats)==true ){  
 okRooms.add(clsrms.get(i));  
 }  
 }  
 return okRooms;  
 }  
  
 public static void writeToFile(String fileName,ArrayList<ClassRoom> okRooms) throws FileNotFoundException {  
 PrintStream ps=new PrintStream(new FileOutputStream("availableclassroom.txt"));  
 System.setOut(ps);  
 for(int i=0;i<okRooms.size();i++){  
 System.out.println(i);  
 okRooms.get(i).printRoom2();  
 }  
 }  
  
 public static void main(String[] args) throws FileNotFoundException {  
 Scanner input=new Scanner(System.in);  
  
 System.out.println("Please input the file name you want too read");  
 String fileName=input.nextLine();  
 ArrayList<ClassRoom> clsrms=readFromFile(fileName);  
  
 System.out.println("There are "+clsrms.size()+" in "+fileName);  
 printRoom(clsrms);  
  
 System.out.println("Please input the info of a classroom which you want to appoint(day time seats): ");  
 int day=input.nextInt();  
 int time=input.nextInt();  
 int seats=input.nextInt();  
 ArrayList<ClassRoom> okRooms=okRooms(clsrms,day,time,seats);  
 System.out.println("There are "+okRooms.size()+" rooms valid for book");  
 printRoom(okRooms);  
  
 writeToFile("avaliableclassroom.txt",okRooms);  
  
 input.close();  
 }  
}**

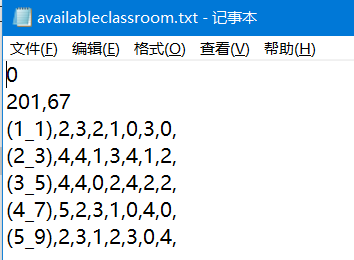
**ClassRoom:**

**import java.io.PrintWriter;  
  
public class ClassRoom {  
 int id;  
 int seatsNumber;  
 int[][] TheArragementOfTheClassRoom = new int[5][7];  
  
 ClassRoom() {  
 }//默认构造方法  
  
 ClassRoom(int id, int seats, int[][] arragement) {  
 this.id = id;  
 seatsNumber = seats;  
 TheArragementOfTheClassRoom = arragement;  
 }//构造方法  
  
 public int getID(ClassRoom cr) {  
 int id = cr.id;  
 return id;  
 }  
  
 public int getSeatsNumber(ClassRoom cr) {  
 int seatsNumber = cr.seatsNumber;  
 return seatsNumber;  
 }  
  
 public int[][] getArrangement(ClassRoom cr) {  
 int[][] arragement = cr.TheArragementOfTheClassRoom;  
 return arragement;  
 }  
  
 public void toString(ClassRoom cr) {  
 System.out.printf("ClassRoom[id:%d,seats:%d] plan: Room #%d with seats(%d)", cr.id, cr.seatsNumber, cr.id, cr.seatsNumber);  
 System.out.println();  
 }  
  
 public void printRoom() {  
 System.out.printf("ClassRoom[id:%d,seats:%d] plan: ", id, seatsNumber);  
 System.out.println();  
 System.out.print(" ");  
 for (int i = 0; i < 7; i++) {  
 System.out.print("Day" + (i + 1) + " ");  
 }  
 System.out.println();  
 for (int i = 0; i < 4; i++) {  
 System.out.print((i + 1) + "(" + (2 \* i + 1) + "\_" + (2 \* i + 2) + ")" + " ");  
 for (int j = 0; j < 7; j++) {  
 System.out.print(TheArragementOfTheClassRoom[i][j] + " ");  
 }  
 System.out.println();  
 }  
 System.out.print("5(9\_10) ");  
 for (int j = 0; j < 7; j++) {  
 System.out.print(TheArragementOfTheClassRoom[4][j] + " ");  
 }  
 System.out.println();  
 }//打印教室安排到System.out  
  
 public void printRoom2() {  
 System.out.printf("%d,%d", id, seatsNumber);  
 System.out.println();  
 for (int i = 0; i < 4; i++) {  
 System.out.print("("+(i + 1) + "\_" + (2 \* i + 1) + ")"+",");  
 for (int j = 0; j < 7; j++) {  
 System.out.print(TheArragementOfTheClassRoom[i][j] + ",");  
 }  
 System.out.println();  
 }  
 System.out.print("(5\_9),");  
 for (int j = 0; j < 7; j++) {  
 System.out.print(TheArragementOfTheClassRoom[4][j] + ",");  
 }  
 System.out.println();  
 }//打印教室安排到PrintWriter.out  
  
 public boolean isValidforExam(int day, int time, int seats) {  
 boolean result = false;  
 if (TheArragementOfTheClassRoom[time - 1][day - 1] == 0 && seats <= seatsNumber) {  
 result = true;  
 }  
 return result;  
 }//判断合格**}









**Exercise2：**

**Main:**

**public class Exercise2 {  
 public static double sumArea(GeometricObject[] a){  
 double sumArea=0;  
 for(int i=0;i<a.length;i++) {  
 sumArea += a[i].getArea();  
 }  
 return sumArea;  
 }  
  
 public static void main(String[] args) {  
 System.out.println("You have created four objects, two circles and two rectangles.");  
 Rectangle rec1=new Rectangle(Double.parseDouble(args[0]),Double.parseDouble(args[1]));  
 Rectangle rec2=new Rectangle(Double.parseDouble(args[2]),Double.parseDouble(args[3]));  
 Circle cir1=new Circle(Double.parseDouble(args[4]));  
 Circle cir2=new Circle(Double.parseDouble(args[5]));  
 System.out.printf("The Circles' radius are: %.1f, %.1f",cir1.radius,cir2.radius);  
 System.out.println();  
 System.out.printf("The rectangles' width are: %.1f, %.1f and height are: %.1f, %.1f",rec1.getWidth(),rec2.getWidth(),rec1.getHeight(),rec2.getHeight());  
 System.out.println();  
 System.out.printf("Two circles are%s equal",Circle.equals(cir1,cir2));  
 System.out.println();  
 System.out.printf("Two rectangles are%s equal",Rectangle.equals(rec1,rec2));  
 System.out.println();  
  
 ComparableCircle comCir1=new ComparableCircle(cir1.radius);  
 ComparableCircle comCir2=new ComparableCircle(cir2.radius);  
 ComparableRectangle comRec1=new ComparableRectangle(rec1.getWidth(),rec1.getHeight());  
 ComparableRectangle comRec2=new ComparableRectangle(rec2.getWidth(),rec2.getHeight());  
 System.out.printf("The circle of radius %.1f is %s than the circle of radius %.1f",cir1.radius,ComparableCircle.parse(comCir1.compareTo(comCir2)),cir2.radius);  
 System.out.println();  
 System.out.printf("The rectangle of area %.2f is %s than the rectangle of area %.2f",rec1.getArea(),ComparableRectangle.parse(comRec1.compareTo(comRec2)),rec2.getArea());  
 System.out.println();  
 GeometricObject[] total=new GeometricObject[]{cir1,cir2,rec1,rec2};  
 System.out.printf("The total areas of the four objects are: %.2f",sumArea(total));  
 }  
}**

**GeometricObject:**

**import java.util.Date;  
  
public abstract class GeometricObject {  
 private String color="white";  
 private boolean filled;  
 private Date dateCreated;  
  
 protected GeometricObject(){  
 dateCreated=new Date();  
 }  
  
 protected GeometricObject(String color,boolean filled){  
 dateCreated=new Date();  
 this.color=color;  
 this.filled=filled;  
 }  
  
 public String getColor(){  
 return color;  
 }  
  
 public void setColor(String color){  
 this.color=color;  
 }  
  
 public boolean isFilled(){  
 return filled;  
 }  
  
 public void setFilled(boolean filled){  
 this.filled=filled;  
 }  
  
 public Date getDateCreated(){  
 return dateCreated;  
 }  
  
 public String toString(){  
 return "Created on "+dateCreated+"\ncolor "+color+" and filled: "+filled;  
 }  
  
 public abstract double getArea();  
  
 public abstract double getPerimeter();  
}**

**Rectangle and ComparableRectangles:**

**public class Rectangle extends GeometricObject {  
 private double width;  
 private double height;  
  
 public Rectangle() {  
 }  
  
 public Rectangle(double width,double height){  
 this.width=width;  
 this.height=height;  
 }  
  
 public Rectangle(double width,double height,String color,boolean filled){  
 this.width=width;  
 this.height=height;  
 setColor(color);  
 setFilled(filled);  
 }  
  
 public double getWidth(){  
 return width;  
 }  
  
 public void setWidth(double width){  
 this.width=width;  
 }  
  
 public double getHeight(){  
 return height;  
 }  
  
 public void setHeight(double height){  
 this.height=height;  
 }  
  
 public double getArea(){  
 return width\*height;  
 }  
  
 public double getPerimeter(){  
 return 2\*(width+height);  
 }  
  
 public static String equals(Rectangle rec1,Rectangle rec2){  
 if(rec1.getArea()==rec2.getArea()){  
 return "";  
 }else return " not";  
 }  
}**

**class ComparableRectangle extends Rectangle implements Comparable<ComparableRectangle> {  
 public ComparableRectangle(double width, double height) {  
 super(width, height);  
 }  
  
 public int compareTo(ComparableRectangle comRectangle) {  
 if (getArea() > comRectangle.getArea()) {  
 return 1;  
 } else {  
 if (getArea() < comRectangle.getArea()) {  
 return -1;  
 } else {  
 return 0;  
 }  
 }  
 }  
  
 public static String parse(int a){  
 String result="";  
 if(a==0){  
 return result="the same";  
 }else {  
 if(a==1){  
 return result="larger";  
 }else {  
 if(a==-1){  
 return result="less";  
 }  
 }  
 }  
 return result;  
 }  
  
 public String toString() {  
 return "Width: " + getWidth() + " Height: " + getHeight() + " Area: " + getArea();  
 }  
}**

**Circle and ComparableCircles:**

**public class Circle extends GeometricObject{  
 double radius;  
  
 public Circle(){};  
  
 public Circle(double radius){  
 this.radius=radius;  
 }  
  
 public Circle(double radius,String color,boolean filled){  
 this.radius=radius;  
 setColor(color);  
 setFilled(filled);//对于父类中的private，必须要定义get和set方法  
 }  
  
 public double getRadius(){  
 return radius;  
 }  
  
 public double getDiameter(){  
 return 2\*radius;  
 }  
  
 public double getArea(){  
 return Math.PI\*Math.pow(radius,2);  
 }  
  
 public double getPerimeter(){  
 return 2\*Math.PI\*radius;  
 }  
  
 public static String equals(Circle cir1, Circle cir2){  
 if(cir1.getRadius()==cir2.getRadius()){  
 return "";  
 }else return " not";  
 }  
}  
  
class ComparableCircle extends Circle implements Comparable<ComparableCircle> {  
 public ComparableCircle(double radius) {  
 super(radius);  
 }  
  
 public int compareTo(ComparableCircle comCircle) {  
 if (getArea() > comCircle.getArea()) {  
 return 1;  
 } else {  
 if (getArea() < comCircle.getArea()) {  
 return -1;  
 } else {  
 return 0;  
 }  
 }  
 }  
  
 public static String parse(int a){  
 String result="";  
 if(a==0){  
 return result="the same";  
 }else {  
 if(a==1){  
 return result="larger";  
 }else {  
 if(a==-1){  
 return result="less";  
 }  
 }  
 }  
 return result;  
 }  
}**

