public class FacultyAdvisor {  
 private String nameAdvisor;  
 private String department;  
 private int maxAdvises;  
 public static final int MIN\_NUM = 0;  
   
 public FacultyAdvisor(String nameAdvisor, String department, int maxAdvises) {  
 if (nameAdvisor == null || nameAdvisor.equals("")) {  
 throw new IllegalArgumentException("First name must be provided");  
 }   
 if (department == null || department.equals("")) {  
 throw new IllegalArgumentException("Last name must be provided");  
 }   
 if (maxAdvises < MIN\_NUM) {  
 throw new IllegalArgumentException("Maximum number able to advise must be provided");  
 }   
 this.nameAdvisor = nameAdvisor;  
 this.department = department;  
 this.maxAdvises = maxAdvises;  
 }  
   
 public String getNameAdvisor() {   
 return this.nameAdvisor; }  
 public String getDepartment() {   
 return this.department; }  
 public int getMaxAdvises() {   
 return this.maxAdvises; }   
   
 public void setNameAdvisor(String nameAdvisor) {  
 if (nameAdvisor == null || nameAdvisor.equals("")) {  
 throw new IllegalArgumentException("Advisor's name must be provided");  
 }   
 this.nameAdvisor = nameAdvisor;  
 }  
  
 public void setDepartment(String department) {  
 if (department == null || department.equals("")) {  
 throw new IllegalArgumentException("Department name must be provided");  
 }   
 this.department = department;  
 }  
   
 public void setMaxAdvises(int maxAdvises) {  
 if (maxAdvises < MIN\_NUM) {  
 throw new IllegalArgumentException("Maximum number able to advise must be provided");  
 }   
 this.maxAdvises = maxAdvises;  
 }  
   
 public String toString() {  
 return "Name: " + this.getNameAdvisor() + "\nDepartment: " + this.getDepartment()  
 + "\nMax Advised Students: " + this.getMaxAdvises();   
 }   
}

public class Student {  
 private String nameStudent;  
 private FacultyAdvisor fa1;  
   
   
 public Student(String nameStudent, FacultyAdvisor fa1) {  
 if (nameStudent == null || nameStudent.equals("")) {  
 throw new IllegalArgumentException("Student name must be provided");  
 }  
 if (fa1 == null) {  
 throw new IllegalArgumentException("Faculty advisor must be provided");  
 }  
   
   
 this.nameStudent = nameStudent;  
 this.fa1 = fa1;  
   
 }  
  
 public String getStudentName() {   
 return this.nameStudent; }  
 public FacultyAdvisor getFacultyAdvisor() {   
 return this.fa1; }  
   
  
 public void setStudentName(String nameStudent) {  
 if (nameStudent == null || nameStudent.equals("")) {  
 throw new IllegalArgumentException("Student name must be provided");  
 }  
 this.nameStudent = nameStudent;  
 }  
  
 public void setFacultyAdvisor(FacultyAdvisor fa1) {  
 this.fa1 = fa1;  
 }   
   
 public String toString() {  
 return "Student: " + this.getStudentName() + "\n"  
 + "Faculty Advisor: "+ this.getFacultyAdvisor().toString() + "\n\n"  
 ;  
 }  
}

import javax.swing.JOptionPane;  
public class StudentImplementation {  
 public static void main(String[] args) {  
 // student name, faculty advisor name, department and max advises  
   
   
   
 Student oneStudent = new Student(StudentName(), new FacultyAdvisor(AdvisorName(), DepartmentName(), numStudentsAdvise()));  
   
 JOptionPane.showMessageDialog(null, oneStudent.toString());  
   
 }  
   
 public static String StudentName() {  
 // aStudentName  
 String aStudentName;  
 // loop to make sure that user input is valid according to data definition class validation  
 do {  
 aStudentName = JOptionPane.showInputDialog("Enter the name of the student : ");  
 // user input for data  
   
 if (aStudentName == null || aStudentName.equals("")) {  
 //error message  
 JOptionPane.showMessageDialog(null, "You must enter a student name.");   
 }  
 } while(aStudentName == null || aStudentName.equals(""));  
   
 return aStudentName;  
 }  
   
   
 public static String AdvisorName() {  
 // aFacultyName  
 String aFacultyName;  
 // loop to make sure that user input is valid according to data definition class validation  
 do {  
 aFacultyName = JOptionPane.showInputDialog("Enter the name of the faculty advisor : ");  
 // user input for data  
   
 if (aFacultyName == null || aFacultyName.equals("")) {  
 //error message  
 JOptionPane.showMessageDialog(null, "You must enter an advisor name.");   
 }  
 } while(aFacultyName == null || aFacultyName.equals(""));  
 return aFacultyName;  
 }  
   
   
 public static String DepartmentName() {  
 // aDepartmentName  
 String aDepartmentName;  
 // loop to make sure that user input is valid according to data definition class validation  
 do {  
 aDepartmentName = JOptionPane.showInputDialog("Enter the name of the department : ");  
 // user input for data  
   
 if (aDepartmentName == null || aDepartmentName.equals("")) {  
 //error message  
 JOptionPane.showMessageDialog(null, "You must enter an advisor name.");   
 }  
 } while(aDepartmentName == null || aDepartmentName.equals(""));  
 return aDepartmentName;  
 }  
   
 public static int numStudentsAdvise() {  
 // aCellphoneNumber  
 int maxAdvise;  
 // loop to make sure that user input is valid according to data definition class validation  
 do {  
 try{  
 maxAdvise = Integer.parseInt(JOptionPane.showInputDialog("Enter the max number of students to advise : ")); // user input for data  
 } catch (NumberFormatException x) {  
 maxAdvise = -1;  
 }  
 if (maxAdvise < 0) {  
 //error message  
 JOptionPane.showMessageDialog(null, "You must enter a positive number.");   
 }  
   
 } while(maxAdvise < 0);  
 return maxAdvise;  
 }  
   
   
}