

File - C:\Users\ctqdt\IdeaProjects\CourseScheduler\src\com\company\data\person\Student.java

```
1  /**
2   * @author Caroline Ta
3   * @since 05.19.2020
4   */
5  package com.company.data.person;
6
7  import com.company.data.course.Course;
8  import com.company.data.course.Session;
9
10 import java.io.File;
11 import java.util.ArrayList;
12 import java.util.Arrays;
13 import java.util.Scanner;
14
15 /**
16  * The type Student.
17  */
18 public class Student extends Person {
19
20     /**
21      * The Student count.
22      */
23     private int studentCount = 0;
24
25     /**
26      * The Student date of birth.
27      */
28     private String dob;
29
30     /**
31      * The Student GPA.
32      */
33     private String gpa;
34
35     /**
36      * The Student start date to attend.
37      */
38     private String dateStart;
39
40     /**
41      * The Student id.
42      */
43     private String id;
44
45     /**
46      * The Student status of scheduled or unscheduled.
47      */
48     private boolean scheduledStatus;
49
50     /**
51      * The Student's list of courses scheduled.
52      */
53     private ArrayList<Course> courseScheduledList;
54
55     /**
56      * The Student's list of sessions scheduled.
57      */
58     private ArrayList<Session> sessionScheduledList;
59
60     /**
61      * The Student's course wish list
62      */
63     private ArrayList<String> courseWishList;
64
65     // -----
66     // CONSTRUCTORS
67     // -----
68
69     /**
70      * Instantiates a new Student.
71      */
72     public Student() {
73     }
74
75     /**
76      * Instantiates a new Student.
77      *
78      * @param nameO the name o
```

```

71      * @param email      the email
72      * @param tel        the tel
73      * @param address0   the address o
74      * @param id         the id
75      * @param dob         the dob
76      * @param gpa         the gpa
77      * @param dateStart  the date start
78      * @throws Exception the exception
79      */
80      public Student(PersonName name0,
81                     String email,
82                     String tel,
83                     PersonAddress address0,
84                     String id,
85                     String dob,
86                     String gpa,
87                     String dateStart
88      ) throws Exception {
89          super(name0, email, tel, address0);
90          setDob(dob);
91          setGpa(gpa);
92          setDateStart(dateStart);
93          setId(id);
94
95          courseScheduledList = new ArrayList<>();
96          sessionScheduledList = new ArrayList<>();
97          courseWishList = new ArrayList<>();
98      }
99
100     // -----
101     // ACCESSORS - GETTER METHODS
102     // -----
103
104     /**
105      * Gets dob.
106      *
107      * @return the dob
108      */
109     public String getDob() {
110         return dob;
111     }
112
113     /**
114      * Gets gpa.
115      *
116      * @return the gpa
117      */
118     public String getGpa() {
119         return gpa;
120     }
121
122     /**
123      * Gets date start.
124      *
125      * @return the date start
126      */
127     public String getDateStart() {
128         return dateStart;
129     }
130
131     /**
132      * Gets id.
133      *
134      * @return the id
135      */
136     public String getId() {
137         return id;
138     }
139
140     /**

```

```

141     * Gets scheduled status.
142     *
143     * @return the scheduled status
144     */
145     public boolean getScheduledStatus() {
146         updateScheduledStatus();
147         return scheduledStatus;
148     }
149
150     /**
151     * Gets course scheduled list.
152     *
153     * @return the course scheduled list
154     */
155     public ArrayList<Course> getCourseScheduledList() {
156         return courseScheduledList;
157     }
158
159     // -----
160     // MUTATORS - SETTER METHODS
161     // -----
162
163     /**
164     * Sets dob.
165     *
166     * @param dob the dob
167     * @throws Exception the exception
168     */
169     public void setDob(String dob) throws Exception {
170         checkStringValue(dob, "Dob");
171         this.dob = dob;
172     }
173
174     /**
175     * Sets gpa.
176     *
177     * @param gpa the gpa
178     * @throws Exception the exception
179     */
180     public void setGpa(String gpa) throws Exception {
181         checkStringValue(gpa, "Gpa");
182         this.gpa = gpa;
183     }
184
185     /**
186     * Sets date start.
187     *
188     * @param dateStart the date start
189     * @throws Exception the exception
190     */
191     public void setDateStart(String dateStart) throws Exception {
192         checkStringValue(dateStart, "Datestart");
193         this.dateStart = dateStart;
194     }
195
196     /**
197     * Sets id.
198     *
199     * @param id the id
200     * @throws Exception the exception
201     */
202     public void setId(String id) throws Exception {
203         checkStringValue(id, "Id");
204         this.id = id;
205     }
206
207     /**
208     * Sets scheduled status.
209     *
210     * @param scheduledStatus the scheduled status

```

```

211     */
212     public void setScheduledStatus(boolean scheduledStatus) {
213         this.scheduledStatus = scheduledStatus;
214     }
215
216     // -----
217     // FUNCTIONALITY METHODS
218     // -----
219
220     /**
221      * Generate course wish list.
222      *
223      * @param pathName the path name
224      */
225     public void generateCourseWishList(String pathName)
226     // Path name: "src\com\company\inputfiles\S001_wish.txt"
227     {
228         String line;
229
230         courseWishList = new ArrayList<>();
231
232         try {
233             File wishListFile = new File(pathName);
234             Scanner input = new Scanner(wishListFile);
235
236             while (input.hasNext()) {
237                 line = input.nextLine();
238
239                 String[] lineAr = line.split("_");
240
241                 courseWishList.addAll(Arrays.asList(lineAr));
242             }
243         } catch (Exception e) {
244             e.printStackTrace();
245         }
246     }
247
248
249
250     /**
251      * Generate id.
252      */
253     public void generateId() {
254         if (studentCount < 10) {
255             this.id = "S00" + studentCount;
256         } else if (studentCount < 100) {
257             this.id = "S0" + studentCount;
258         } else {
259             this.id = "S" + studentCount;
260         }
261     }
262
263     /**
264      * Update scheduled status.
265      */
266     public void updateScheduledStatus() {
267         if (sessionScheduledList.size() > 0 || courseScheduledList.size() > 0) {
268             scheduledStatus = true;
269         } else {
270             scheduledStatus = false;
271         }
272     }
273
274
275     /**
276      * Add course.
277      *
278      * @param courseToAdd the course to add
279      */
280     public void addCourse(Course courseToAdd) {

```

```

281         if (!courseScheduledList.contains(courseToAdd)) {
282             courseScheduledList.add(courseToAdd);
283         }
284         updateScheduledStatus();
285     }
286
287     /**
288      * Remove course boolean.
289      *
290      * @param courseToRemove the course to remove
291      * @return the boolean
292      */
293     public boolean removeCourse(Course courseToRemove) {
294         courseScheduledList.remove(courseToRemove);
295         updateScheduledStatus();
296         return courseScheduledList.remove(courseToRemove);
297     }
298
299     /**
300      * Add session.
301      *
302      * @param sessionToAdd the session to add
303      */
304     public void addSession(Session sessionToAdd) {
305         if (!sessionScheduledList.contains(sessionToAdd)) {
306             sessionScheduledList.add(sessionToAdd);
307         }
308         sessionScheduledList.add(sessionToAdd);
309         updateScheduledStatus();
310     }
311
312     /**
313      * Remove session boolean.
314      *
315      * @param sessionToRemove the session to remove
316      * @return the boolean
317      */
318     public boolean removeSession(Session sessionToRemove) {
319         sessionScheduledList.remove(sessionToRemove);
320         updateScheduledStatus();
321         return sessionScheduledList.remove(sessionToRemove);
322     }
323
324     // -----
325     // OVERRIDDEN METHODS
326     // -----
327
328     /**
329      * Override toString method.
330      *
331      * @return a string description of the Student class.
332      */
333     @Override
334     public String toString() {
335
336         updateScheduledStatus();
337
338         StringBuilder builder = new StringBuilder();
339         builder.append(super.toString());
340
341         builder.append("\nDate of birth: ");
342         builder.append(dob);
343
344         builder.append("\nGPA: ");
345         builder.append(gpa);
346
347         builder.append("\nDate start: ");
348         builder.append(dateStart);
349
350         builder.append("\nID: ");

```

File - C:\Users\ctqdt\IdeaProjects\CourseScheduler\src\com\company\data\person\Student.java

```
351         builder.append(id);
352
353         builder.append("\nScheduled: ");
354         builder.append(scheduledStatus);
355
356         return builder.toString();
357     }
358 }
359
```