

```

1 /**
2  * @author Caroline Ta
3  * @since 05.19.2020
4  */
5
6 // -----
7 // Ctrl + Shift + Alt + G = Generate all Javadoc documents
8 // Ctrl + Shift + Alt + Z = Remove all Javadoc documents
9
10 // Ctrl + Shift + A: "Press Alt+Enter to assign a shortcut"
11
12 // Collapse javadoc comments: Alt + "-"
13 // Expand javadoc comments: Alt + "="
14
15 // Tools > Generate Javadoc
16
17 // Tools > Generate UML Diagram
18 // -----
19
20 package com.company;
21
22 import com.company.data.course.*;
23 import com.company.data.person.*;
24
25 import java.io.File;
26 import java.io.FileNotFoundException;
27 import java.io.FileOutputStream;
28 import java.io.IOException;
29 import java.util.Scanner;
30
31 import static java.lang.Boolean.parseBoolean;
32 import static java.lang.Integer.parseInt;
33
34 /**
35  * The type Course scheduler main.
36  */
37 public class CourseSchedulerMain {
38
39     /**
40      * The entry point of application.
41      *
42      * @param args the input arguments
43      * @throws IOException the io exception
44      */
45     public static void main(String[] args) throws IOException {
46
47         //-----
48         // CONSTANTS & VARIABLES
49         //-----
50
51         final String WELCOME =
52             "\n-----" +
53             "\nWELCOME TO THE COURSE SCHEDULER" +
54             "\n-----" +
55             "\n[0] Exit the program" +
56             "\n[1] Log in as an Admin" +
57             "\n[2] Log in as a Faculty" +
58             "\n[3] Log in as a Student";
59
60         final String ADMIN =
61             "\n-----" +
62             "\nYOU ARE LOGGED IN AS ADMIN" +
63             "\n-----" +
64             "\n[0] Exit - to COURSE SCHEDULER" +
65             "\n[1] View Course Schedule" +
66             "\n[2] View Faculty List" +
67             "\n[3] View Student List" +
68             "\n[4] Output file ScheduledCourseSessions.txt" +
69             "\n[5] Output file UnscheduledCourseSessions.txt" +
70             "\n[6] Output file Faculty.txt" +
71             "\n[7] Output file ScheduledStudents.txt" +
72             "\n[8] Output file UnscheduledStudents.txt";
73
74         final String FACULTY =
75             "\n-----" +
76             "\nYOU ARE LOGGED IN AS FACULTY" +
77             "\n-----" +
78             "\n[0] Exit - to COURSE SCHEDULER" +
79             "\n[1] View Course Schedule" +
80             "\n[2] View My Course List" +
81             "\n[3] View My Information" +
82             "\n[4] Assign myself to a Course" +
83             "\n[5] Remove myself from a Course";
84
85         final String STUDENT =
86             "\n-----" +
87             "\nYOU ARE LOGGED IN AS STUDENT" +
88             "\n-----" +
89             "\n[0] Exit - to COURSE SCHEDULER" +
90             "\n[1] View Course Schedule" +
91             "\n[2] View My Course List" +

```

File - C:\Users\ctqdt\IdeaProjects\CourseScheduler\src\com\company\CourseSchedulerMain.java

```
92         "\n[3] View My Information" +
93         "\n[4] Enroll myself to a Course" +
94         "\n[5] Unenroll myself from a Course";
95
96     // INPUT FILE NAMES
97
98     final String FACULTY_INPUT_FILE = "src\\com\\company\\inputFiles\\info_faculty_input.txt";
99     final String STUDENT_INPUT_FILE = "src\\com\\company\\inputFiles\\info_student_input.txt";
100    final String COURSE_INPUT_FILE = "src\\com\\company\\inputFiles\\info_course_input.txt";
101
102    final String SESSION_1A_INPUT_FILE = "src\\com\\company\\inputFiles\\1A_session_id_input.txt";
103    final String SESSION_1B_INPUT_FILE = "src\\com\\company\\inputFiles\\1B_session_id_input.txt";
104    final String SESSION_1C_INPUT_FILE = "src\\com\\company\\inputFiles\\1C_session_id_input.txt";
105    final String SESSION_1D_INPUT_FILE = "src\\com\\company\\inputFiles\\1D_session_id_input.txt";
106    final String SESSION_4A_INPUT_FILE = "src\\com\\company\\inputFiles\\4A_session_id_input.txt";
107
108    // OUTPUT FILE PATHS
109
110    final String SCHEDULED_COURSE_SESSIONS_OUTPUT = "out\\outputFiles\\ScheduledCourseSessions.txt";
111    final String UNSCHEDULED_COURSE_SESSIONS_OUTPUT = "out\\outputFiles\\UnscheduledCourseSessions.txt";
112    final String FACULTY_OUTPUT = "out\\outputFiles\\Faculty.txt";
113    final String SCHEDULED_STUDENTS_OUTPUT = "out\\outputFiles\\ScheduledStudents.txt";
114    final String UNSCHEDULED_STUDENTS_OUTPUT = "out\\outputFiles\\UnscheduledStudents.txt";
115
116    // FOR PERSON - 10 categories
117
118    String firstName;
119    String middleName;
120    String lastName;
121    String email;
122    String tel;
123    String street;
124    String city;
125    String state;
126    String zip;
127    String id;
128
129    // FOR FACULTIES - 2 categories
130
131    String hireDate;
132    String tenured;
133
134    // FOR STUDENTS - 3 categories
135
136    String dob;
137    String gpa;
138    String dateStart;
139
140    // FOR COURSE
141    String department;
142    String code;
143    String description;
144    String min;
145    String max;
146
147    String line;
148
149    int loginOption = 0;
150    String loginNameTel;
151
152    String fileData;
153    FileOutputStream fos;
154
155    //-----
156    // START GENERATE INFORMATION FROM INPUT FILES
157    //-----
158
159    PersonDirectory personDirectory = new PersonDirectory();
160    CourseDirectory courseDirectory = new CourseDirectory();
161
162    Scanner input;
163
164    // GENERATE LIST OF FACULTY OBJECTS WITH INFO FROM INPUT FILE
165    try {
166        File facultyFile = new File(FACULTY_INPUT_FILE);
167        input = new Scanner(facultyFile);
168
169        while (input.hasNext()) {
170
171            line = input.nextLine();
172
173            String[] lineAr = line.split("_");
174
175            firstName = lineAr[0];
176            middleName = lineAr[1];
177            lastName = lineAr[2];
178            email = lineAr[3];
179            tel = lineAr[4];
180            street = lineAr[5];
181            city = lineAr[6];
182            state = lineAr[7];
```

File - C:\Users\ctqdt\IdeaProjects\CourseScheduler\src\com\company\CourseSchedulerMain.java

```
183         zip = lineAr[8];
184         id = lineAr[9];
185         hireDate = lineAr[10];
186         tenured = lineAr[11];
187
188         Person person = new Faculty(new PersonName(firstName, middleName, lastName), email, tel,
189                                     new PersonAddress(street, city, state, zip), id, hireDate, parseBoolean(tenured));
190
191         personDirectory.addPerson(person);
192     }
193 } catch (Exception e) {
194     e.printStackTrace();
195 }
196
197 // GENERATE LIST OF STUDENT OBJECTS WITH INFO FROM INPUT FILE
198 try {
199     File studentFile = new File(STUDENT_INPUT_FILE);
200     input = new Scanner(studentFile);
201
202     while (input.hasNext()) {
203
204         line = input.nextLine();
205
206         String[] lineAr = line.split("_");
207
208         firstName = lineAr[0];
209         middleName = lineAr[1];
210         lastName = lineAr[2];
211         email = lineAr[3];
212         tel = lineAr[4];
213         street = lineAr[5];
214         city = lineAr[6];
215         state = lineAr[7];
216         zip = lineAr[8];
217         id = lineAr[9];
218         dob = lineAr[10];
219         gpa = lineAr[11];
220         dateStart = lineAr[12];
221
222         Person person = new Student(new PersonName(firstName, middleName, lastName), email, tel,
223                                     new PersonAddress(street, city, state, zip), id, dob, gpa, dateStart);
224
225         personDirectory.addPerson(person);
226     }
227 } catch (Exception e) {
228     e.printStackTrace();
229 }
230
231 // GENERATE LIST OF COURSE OBJECTS WITH INFO FROM INPUT FILE
232 try {
233     File courseFile = new File(COURSE_INPUT_FILE);
234     input = new Scanner(courseFile);
235     Course course;
236
237     while (input.hasNext()) {
238
239         line = input.nextLine();
240
241         String[] lineAr = line.split("_");
242
243         department = lineAr[0];
244         code = lineAr[1];
245         description = lineAr[2];
246         id = lineAr[3];
247         min = lineAr[4];
248         max = lineAr[5];
249
250         course = new Course(department, code, description, id, parseInt(min), parseInt(max));
251
252         courseDirectory.addCourse(course);
253     }
254
255     // GENERATE LIST OF SESSIONS FOR EACH COURSE
256     courseDirectory.generateSession(courseDirectory.getCourseList().get(0).getCourseId(), SESSION_1A_INPUT_FILE);
257     courseDirectory.generateSession(courseDirectory.getCourseList().get(1).getCourseId(), SESSION_1B_INPUT_FILE);
258     courseDirectory.generateSession(courseDirectory.getCourseList().get(2).getCourseId(), SESSION_1C_INPUT_FILE);
259     courseDirectory.generateSession(courseDirectory.getCourseList().get(3).getCourseId(), SESSION_1D_INPUT_FILE);
260     courseDirectory.generateSession(courseDirectory.getCourseList().get(4).getCourseId(), SESSION_4A_INPUT_FILE);
261
262 } catch (Exception e) {
263     e.printStackTrace();
264 }
265
266 //-----
267 // START COURSE SCHEDULER PROGRAM
268 //-----
269
270 input = new Scanner(System.in);
271 System.out.println(WELCOME);
272 System.out.print("Option: ");
273 loginOption = input.nextInt();
```

```

274
275     while (loginOption != 0) {
276         switch (loginOption) {
277             //-----
278             // ADMIN LOGIN
279             //-----
280             case 1 -> {
281                 System.out.println(ADMIN);
282                 System.out.print("Option: ");
283                 loginOption = input.nextInt();
284                 while (loginOption != 0) {
285                     switch (loginOption) {
286                         case 1: // View Course Session
287                             System.out.println(courseDirectory.printCourseWSession());
288                             break;
289
290                         case 2: // View Faculty List
291                             System.out.println(personDirectory.getFacultyAsString());
292                             break;
293
294                         case 3: // View Student List
295                             System.out.println(personDirectory.getStudentAsString());
296                             break;
297
298                         case 4: // ScheduledCourseSessions.txt
299                             try {
300                                 fileData = courseDirectory.getScheduledCourseSessionsOutputData();
301
302                                 fos = new FileOutputStream(SCHEDULED_COURSE_SESSIONS_OUTPUT);
303                                 fos.write(fileData.getBytes());
304                                 fos.flush();
305                                 fos.close();
306
307                                 System.out.println("\nOutputting file ScheduledCourseSessions.txt...");
308                             } catch (Exception e) {
309                                 e.printStackTrace();
310                             }
311                             break;
312
313                         case 5: // UnscheduledCourseSessions.txt
314                             try {
315                                 fileData = courseDirectory.getUnscheduledCourseSessionsOutputData();
316
317                                 fos = new FileOutputStream(UNSCHEDULED_COURSE_SESSIONS_OUTPUT);
318                                 fos.write(fileData.getBytes());
319                                 fos.flush();
320                                 fos.close();
321
322                                 System.out.println("\nOutputting file UnscheduledCourseSessions.txt...");
323                             } catch (Exception e) {
324                                 e.printStackTrace();
325                             }
326                             break;
327
328                         case 6: // Faculty.txt
329                             try {
330                                 fileData = personDirectory.getFacultyOutputData();
331
332                                 fos = new FileOutputStream(FACULTY_OUTPUT);
333                                 fos.write(fileData.getBytes());
334                                 fos.flush();
335                                 fos.close();
336
337                                 System.out.println("\nOutputting file Faculty.txt...");
338                             } catch (Exception e) {
339                                 e.printStackTrace();
340                             }
341                             break;
342
343                         case 7: // ScheduledStudents.txt
344                             try {
345                                 fileData = personDirectory.getScheduledStudentsOutputData();
346
347                                 fos = new FileOutputStream(SCHEDULED_STUDENTS_OUTPUT);
348                                 fos.write(fileData.getBytes());
349                                 fos.flush();
350                                 fos.close();
351
352                                 System.out.println("\nOutputting file ScheduledStudents.txt...");
353                             } catch (Exception e) {
354                                 e.printStackTrace();
355                             }
356                             break;
357
358                         case 8: // UnscheduledStudents.txt
359                             try {
360                                 fileData = personDirectory.getUnscheduledStudentsOutputData();
361
362                                 fos = new FileOutputStream(UNSCHEDULED_STUDENTS_OUTPUT);
363                                 fos.write(fileData.getBytes());
364                                 fos.flush();

```

```

365         fos.close();
366
367         System.out.println("\nOutputting file UnscheduledStudents.txt...");
368     } catch (Exception e) {
369         e.printStackTrace();
370     }
371     break;
372 }
373 System.out.println(ADMIN);
374 System.out.print("Option: ");
375 loginOption = input.nextInt();
376 }
377 }
378
379 //-----
380 // FACULTY LOGIN
381 //-----
382 // TEST INPUT: Zeph Prissy Spalding 480586
383 // Zeph_Prissy_Spalding_zps@gmail.com_480586_KLM_Mission Viejo_CA_92692_F001_01/01/2010_true
384 case 2 -> {
385     System.out.println();
386     System.out.println("Enter your full name and phone number as the following format");
387     System.out.print("(FirstName MiddleName LastName 123456): ");
388     firstName = input.next();
389     middleName = input.next();
390     lastName = input.next();
391     tel = input.next();
392     System.out.println(FACULTY);
393     System.out.print("Option: ");
394     loginOption = input.nextInt();
395     while (loginOption != 0) {
396         switch (loginOption) {
397             case 1: // View Course Schedule
398                 System.out.println(courseDirectory.printCourseWSession());
399                 break;
400
401             case 2: // View My Course List
402                 System.out.println(((Faculty) personDirectory.findPerson(tel)).getFacultyCourseList());
403                 break;
404
405             case 3: // View My Information
406                 System.out.println(personDirectory.findPerson(tel).toString());
407                 break;
408
409             case 4: // Assign myself to a Course
410                 System.out.print("\nEnter the Id of the Course you would like to teach: ");
411                 String courseIdAdd = input.next().toUpperCase();
412                 Course courseToAdd = courseDirectory.findCourse(courseIdAdd);
413
414                 ((Faculty) personDirectory.findPerson(tel)).addCourse(courseToAdd);
415
416                 System.out.println();
417                 System.out.println("-----");
418                 System.out.println("SESSIONS AVAILABLE FOR THIS COURSE");
419                 System.out.println("-----");
420                 for (int i = 0; i < courseToAdd.sessionList.size(); i++) {
421                     System.out.print(courseToAdd.sessionList.get(i).toString());
422                 }
423
424                 System.out.print("\nEnter the Id of the Session you would like to teach: ");
425                 String sessionIdAdd = input.next().toUpperCase();
426                 Session sessionToAdd = courseDirectory.findCourse(courseIdAdd).findSession(sessionIdAdd);
427
428                 sessionToAdd.setTeacher(((Faculty) personDirectory.findPerson(tel)));
429
430                 System.out.println();
431                 System.out.println("-----");
432                 System.out.println("SESSIONS FOR THIS COURSE IS UPDATED");
433                 System.out.println("-----");
434                 for (int i = 0; i < courseToAdd.sessionList.size(); i++) {
435                     System.out.print(courseToAdd.sessionList.get(i).toString());
436                 }
437                 break;
438
439             case 5: // Remove myself from a Course
440                 System.out.print("\nEnter the Id of the Course you would like to quit: ");
441                 String courseIdRemove = input.next().toUpperCase();
442                 Course courseToRemove = courseDirectory.findCourse(courseIdRemove);
443
444                 ((Faculty) personDirectory.findPerson(tel)).removeCourse(courseToRemove);
445
446                 System.out.println();
447                 System.out.println("-----");
448                 System.out.println("SESSIONS AVAILABLE FOR THIS COURSE");
449                 System.out.println("-----");
450                 for (int i = 0; i < courseToRemove.sessionList.size(); i++) {
451                     System.out.print(courseToRemove.sessionList.get(i).toString());
452                 }
453
454                 System.out.print("\nEnter the Id of the Session you would like to quit: ");
455                 String sessionIdRemove = input.next().toUpperCase();

```

```

456 sessionToRemove = courseDirectory.findCourse(courseIdRemove).findSession(sessionIdRemove);
457
458 sessionToRemove.unsetTeacher();
459
460 System.out.println();
461 System.out.println("-----");
462 System.out.println("SESSIONS FOR THIS COURSE IS UPDATED");
463 System.out.println("-----");
464 for (int i = 0; i < courseToRemove.sessionList.size(); i++) {
465     System.out.print(courseToRemove.sessionList.get(i).toString());
466 }
467 break;
468 default:
469 }
470 System.out.println(FACULTY);
471 System.out.print("Option: ");
472 loginOption = input.nextInt();
473 }
474 }
475 //-----
476 // STUDENT LOGIN
477 //-----
478 // TEST INPUT: Ruthie Bernadine Hanley 182575
479 // TEST INPUT: Camille Alfred Emmett 681690
480 // Ruthie_Bernadine_Hanley_rbh@gmail.com_182575_JKL_Mission Viejo_CA_92692_S010_10/01/1998_4.0_10/20/2016
481 // Camille_Alfred_Emmett-cae@gmail.com_681690_IJK_Mission Viejo_CA_92692_S009_09/01/1998_3.9_09/20/2016
482 case 3 -> {
483     System.out.println();
484     System.out.println("Enter your full name and phone number as the following format");
485     System.out.print("(FirstName MiddleName LastName 123456): ");
486     firstName = input.next();
487     middleName = input.next();
488     lastName = input.next();
489     tel = input.next();
490     System.out.println(STUDENT);
491     System.out.print("Option: ");
492     loginOption = input.nextInt();
493     while (loginOption != 0) {
494         switch (loginOption) {
495             case 1: // View Course Schedule
496                 System.out.println(courseDirectory.printCourseWSession());
497                 break;
498
499             case 2: // View My Course List
500                 System.out.println(((Student) personDirectory.findPerson(tel)).getCourseScheduledList());
501                 break;
502
503             case 3: // View My Information
504                 System.out.println(personDirectory.findPerson(tel).toString());
505                 break;
506
507             case 4: // Enroll myself to a Course
508                 System.out.print("\nEnter the Id of the Course you would like to enroll: ");
509                 String courseIdAdd = input.next().toUpperCase();
510                 Course courseToAdd = courseDirectory.findCourse(courseIdAdd);
511
512                 // ADDING COURSE HERE
513                 ((Student) personDirectory.findPerson(tel)).addCourse(courseToAdd);
514
515                 System.out.println();
516                 System.out.println("-----");
517                 System.out.println("SESSIONS AVAILABLE FOR THIS COURSE");
518                 System.out.println("-----");
519                 for (int i = 0; i < courseToAdd.sessionList.size(); i++) {
520                     System.out.print(courseToAdd.sessionList.get(i).toString());
521                 }
522
523                 System.out.print("\nEnter the Id of the Session you would like to enroll: ");
524                 String sessionIdAdd = input.next().toUpperCase();
525                 Session sessionToAdd = courseDirectory.findCourse(courseIdAdd).findSession(sessionIdAdd);
526
527                 // ADDING SESSION HERE
528                 sessionToAdd.addStudent(((Student) personDirectory.findPerson(tel)));
529
530                 // UPDATE COURSE WITH SESSION + STUDENT
531                 courseDirectory.findCourse(courseIdAdd).enrollStudent(((Student) personDirectory.findPerson(tel)
532     )), sessionToAdd);
533
534                 System.out.println();
535                 System.out.println("-----");
536                 System.out.println("SESSIONS FOR THIS COURSE IS UPDATED");
537                 System.out.println("-----");
538                 for (int i = 0; i < courseToAdd.sessionList.size(); i++) {
539                     System.out.print(courseToAdd.sessionList.get(i).toString());
540                 }
541                 break;
542
543             case 5: // Unenroll myself from a Course
544                 System.out.print("\nEnter the Id of the Course you would like to enroll: ");
545                 String courseIdRemove = input.next().toUpperCase();
546                 Course courseToRemove = courseDirectory.findCourse(courseIdRemove);

```

```

546
547 // REMOVE COURSE HERE
548 ((Student) personDirectory.findPerson(tel)).removeCourse(courseToRemove);
549
550 System.out.println();
551 System.out.println("-----");
552 System.out.println("SESSIONS AVAILABLE FOR THIS COURSE");
553 System.out.println("-----");
554 for (int i = 0; i < courseToRemove.sessionList.size(); i++) {
555     System.out.print(courseToRemove.sessionList.get(i).toString());
556 }
557
558 System.out.print("\nEnter the Id of the Session you would like to enroll: ");
559 String sessionIdRemove = input.next().toUpperCase();
560 Session sessionToRemove = courseDirectory.findCourse(courseIdRemove).findSession(sessionIdRemove);
561
562 // REMOVE SESSION HERE
563 sessionToRemove.removeStudent((Student) personDirectory.findPerson(tel));
564
565 System.out.println();
566 System.out.println("-----");
567 System.out.println("SESSIONS FOR THIS COURSE IS UPDATED");
568 System.out.println("-----");
569 for (int i = 0; i < courseToRemove.sessionList.size(); i++) {
570     System.out.print(courseToRemove.sessionList.get(i).toString());
571 }
572 break;
573 }
574
575 System.out.println(STUDENT);
576 System.out.print("Option: ");
577 loginOption = input.nextInt();
578 }
579 }
580 }
581
582 System.out.println(WELCOME);
583 System.out.print("Option: ");
584 loginOption = input.nextInt();
585 } // Program running Switch statements
586 }
587 }
588

```