```
1  package socialmedia;
2
3  import java.util.ArrayList;
4
5  /**
6   * This is the Account class
7   * @author Leon Ingall, Charles Pearman-Wright
8   * @version 1.0
9   */
10
11 public class Account {
12
13    private String handle;
14    private String description;
15    private ArrayList<Post> posts = new ArrayList<Post>();
16    private int id;
17
18    /**
19     * This is the constructor method for Account.
20     * @param handle the handle of the user
21     * @param description a short personal description
22     */
23    public Account(String handle, String description) {
24       this.handle = handle;
25       this.description = description;
26    }
27    /**
28     * This is the constructor method for Account.
29     * @param description a short personal description
30     */
31    public Account(String handle) {
```

```
32          this.handle = handle;
33      }
34
35      /**
36       * This returns the id of the Account
37       * @return id
38       */
39      public int getID() {
40          return id;
41      }
42      /**
43       * This sets the the id of the Account
44       * @param id The id that will be set
45       */
46      public void setID(int id) {
47          this.id = id;
48      }
49
50      /**
51       * This returns the handle of the Account
52       * @return handle
53       */
54      public String getHandle() {
55          return handle;
56      }
57      /**
58       * This sets the handle of the Account
59       * @param handle The handle that will be set
60       */
61      public void setHandle(String handle) {
62          this.handle = handle;
```

```
63      }
64      /**
65       * This gets the description of the user
66       * @return description
67       */
68      public String getDescription() {
69         return description;
70      }
71      /**
72       * This sets the description of the Account
73       * @param desc The description that will be set
74       */
75      public void setDescription(String desc) {
76         this.description = desc;
77      }
78
79      /**
80       * This returns an ArrayList of all the posts the user has made
81       * @return posts
82       */
83      public ArrayList<Post> getPosts() {
84         return posts;
85      }
86      /**
87       * Sets the posts ArrayList
88       * @param posts
89       */
90      public void setPosts(ArrayList<Post> posts) {
91         this.posts = posts;
92      }
93   }
```

```
1   package socialmedia;
2
3   import java.util.ArrayList;
4
5   /**
6    * This is the Comment class
7    * @author Leon Ingall, Charles Pearman-Wright
8    * @version 1.0
9    */
10
11
12  public class Comment extends Post {
13
14      private int postID;
15      private int id;
16      private ArrayList<Post> children = new ArrayList<Post>();
17
18      /**
19       * This is the constructor method of Comment
20       * @param handle The handle of the user that made this comment
21       * @param postID The id of the posts that this comments
22       * @param content The message of the comment
23       */
24      public Comment(String handle, int postID, String content) {
25          super(content,handle);
26          this.postID = postID;
27      }
28      /**
29       * This gets the id of the original post
30       * @return postID
31       */
```

```
32      public int getPostID() {return postID;}
33      /**
34       * This sets the id of the original post
35       * @param postID the id to be set
36       */
37      public void setPostID(int postID) {this.postID = postID;}
38
39      /**
40       * This gets the id of the comment
41       * @return id
42       */
43      public int getID() {return id;}
44      /**
45       * This sets the id of the comment
46       * @param id The id to be set
47       */
48      public void setID(int id) {this.id = id;}
49
50      /**
51       * This gets the ArrayList of children of this comment
52       * @return childen
53       */
54      public ArrayList<Post> getChildren() {return children;}
55      /**
56       * This adds a child to the children ArrayList
57      * @param post The post to be added to the ArrayList
58       */
59      public void addChild(Post post) {children.add(post);}
60
61
62  }
```

```
1  package socialmedia;
2
3  /**
4   * This is the Endorsement class
5   * @author Leon Ingall, Charles Pearman-Wright
6   * @version 1.0
7   */
8
9  public class Endorsement extends Post {
10
11    private int postID;
12    private String content;
13    private int id;
14
15    /**
16     * This is the constructor method for Endoresement
17     * @param handle the handle of the user endoring a post
18     * @param postID the id of the post it is endorsing
19     */
20    public Endorsement(String handle, int postID) {
21      //make new post with empty content - said content gets filled in
22      //as soon as it's made in the endorsePost method
23      super("", handle);
24      this.postID = postID;
25    }
26
27    /**
28     * This gets the id of the Endorsement
29     * @return id
30     */
31    public int getID() {return id;}
```

```
32      /**
33       * This sets the id of the Endorsement
34       * @param id The id to be set
35       */
36      public void setID(int id) {this.id = id;}
37
38      /**
39       * This gets the id of the original post
40       * @return postID
41       */
42      public int getPostID() {return postID;}
43      /**
44       * This sets the id of the original post
45       * @param postID The id to be set
46       */
47      public void setPostID(int postID) {this.postID = postID;}
48
49      /**
50       * This gets the content of the Endorsement
51       * @return content
52       */
53      public String getContent() {return content;}
54      /**
55       * This sets the content of the Endorsement
56       * @param content The content to be set
57       */
58      public void setContent(String content) {this.content = content;}
59
60
61  }
```

```
1  package socialmedia;
2
3  import java.util.ArrayList;
4
5  /**
6   * This is the Post class
7   * @author Leon Ingall, Charles Pearman-Wright
8   * @version 1.0
9   */
10
11 public class Post {
12
13    private String content;
14    private String handle;
15    private ArrayList<Post> children = new ArrayList<Post>();
16    private int id;
17
18    /**
19     * This is the contstructor method of Post
20     * @param content The content of the Post
21     * @param handle The handle of the Post
22     */
23    public Post(String content, String handle) {
24       this.content = content;
25       this.handle = handle;
26    }
27
28    /**
29     * This gets the content of the Post
30     * @return content
31     */
```

```
32      public String getContent() {
33          return content;
34      }
35      /**
36       * This sets the content of the Post
37       * @param content The content to be set
38       */
39      public void setContent(String content) {
40          this.content = content;
41      }
42
43      /**
44       * This gets the handle of the Post
45       * @return handle
46       */
47      public String getHandle() {
48          return handle;
49      }
50      /**
51       * This sets the handle of the Post
52       * @param handle The handle to be set
53       */
54      public void setHandle(String handle) {
55          this.handle = handle;
56      }
57
58      /**
59       * This gets the id of the Post
60      * @return id
61       */
62      public int getID() {
```

```
63          return id;
64      }
65      /**
66       * This sets the id of the Post
67       * @param id The id to be set
68       */
69      public void setID(int id) {
70          this.id = id;
71      }
72
73      /**
74       * This gets the ArrayList of children of the post
75       * @return children
76       */
77      public ArrayList<Post> getChildren() {
78          return children;
79      }
80      /**
81       * This adds a child to the ArrayList
82       * @param post The post to be added
83       */
84      public void addChild(Post post) {
85          children.add(post);
86      }
87      /**
88       * This removes an endorsement from the list of children
89       * @param endorsement The endorsement to be removed
90       */
91      public void removeEndorsement(Endorsement endorsement) {
92          // Iterates through the children ArrayList
93          for(Post post: children) {
```

```
94              // checks if it's an Endorsement
95              if(post.getClass() == Endorsement.class && (Endorsement)post == endorsement) {
96                  // removes it
97                  children.remove(post);
98
99              }
100         }
101     }
102
103     /**
104      * This gets an ArrayList of all the endorsements of this Post
105      * @return endorsements
106      */
107     public ArrayList<Endorsement> getEndorsements() {
108         ArrayList<Endorsement> endorsements = new ArrayList<>();
109         for(Post post: children) {
110             // checks if it's an Endorsement
111             if(post.getClass() == Endorsement.class) {
112                 // adds it to the list
113                 Endorsement e = (Endorsement)post;
114                 endorsements.add(e);
115             }
116         }
117         return endorsements;
118     }
119
120     public ArrayList<Comment> getComments() {
121         ArrayList<Comment> comments = new ArrayList<>();
122         for(Post post: children) {
123             // checks if it's a comment
124             if(post.getClass() == Comment.class) {
```

```
125                 // adds it to the list
126                 Comment e = (Comment)post;
127                 comments.add(e);
128               }
129           }
130           return comments;
131       }
132
133   }
```

```
1  package socialmedia;
2
3  import java.io.FileInputStream;
4  import java.io.FileNotFoundException;
5  import java.io.FileOutputStream;
6  import java.io.IOException;
7  import java.io.ObjectInputStream;
8  import java.io.ObjectOutputStream;
9  import java.util.ArrayList;
10
11 /**
12  * SocialMedia is a compiling, functioning implementor of
13  * the SocialMediaPlatform interface.
14  *
15  * @author Leon Ingall, Charles Pearman-Wright
16  * @version 1.0
17  */
18
19 public class SocialMedia implements SocialMediaPlatform {
20
21         public ArrayList<Account> allAccounts = new ArrayList<Account>();
22         public ArrayList<Post> allPosts = new ArrayList<Post>();
23         public static int counter;
24         public static StringBuilder childrenDetails;
25         public static int postIDTally = 0;
26
27         @Override
28         public int createAccount(String handle) throws IllegalHandleException,
29 InvalidHandleException {
30
31                 Boolean valid = true;
```

```
32
33                      for (Account acc : allAccounts) {
34                              // checks if the handle is already taken
35                              if (acc.getHandle().equals(handle) ) {
36                                      valid = false;
37                              }
38                      }
39                      if (valid == false) {
40                              throw new IllegalHandleException("Error - Handle already exists.");
41                      }
42
43                      if ((handle.length()) == 0 || (handle.length() > 30)) {
44                              throw new InvalidHandleException("Handle must be between 0 and 30
45      characters");
46                      }
47
48                      if (handle.contains(" ")) {
49                              throw new InvalidHandleException("Handle can't contain whitespace.");
50                      }
51
52
53                      Account acc = new Account(handle); //create new account
54                      allAccounts.add(acc);                                   //add it to arraylist
55                      acc.setID(allAccounts.size());                  //set id to size of list of all accounts
56
57                      return acc.getID();
58              }
59
60              @Override
61              public int createAccount(String handle, String description) throws IllegalHandleException,
62      InvalidHandleException {
63
```

```
64                      Boolean valid = true;
65
66                      for (Account acc : allAccounts) {
67                              // checks if the handle is already taken
68                              if (acc.getHandle().equals(handle) ) {
69                                      valid = false;
70                              }
71                      }
72                      if (valid == false) {
73                              throw new IllegalHandleException("Error - Handle already exists.");
74                      }
75
76                      if ((handle.length()) == 0 || (handle.length() > 30)) {
77                              throw new InvalidHandleException("Handle must be between 0 and 30
78  characters");
79                      }
80
81                      if (handle.contains(" ")) {
82                              throw new InvalidHandleException("Handle can't contain whitespace.");
83                      }
84
85                      Account acc = new Account(handle,description); //create new account
86                      allAccounts.add(acc);                                   //add it to arraylist
87                      acc.setID(allAccounts.size());              //set id to size of list of all accounts
88
89                      return acc.getID();
90              }
91
92              @Override
93              public void removeAccount(int id) throws AccountIDNotRecognisedException {
94
```

```
95
96                      for (Account acc : allAccounts) {
97                              if (acc.getID() == id) {
98                                      // remove all posts of that user
99                                      for(Post post: allPosts) {
100                                             if(post.getHandle() == acc.getHandle()) {
101                                                     try {
102                     deletePost(post.getID());
103                 } catch (PostIDNotRecognisedException e) {
104                     // if post id not recognized
105                     e.printStackTrace();
106                 }
107                                             }
108                                     }
109                                     // remove the account
110                                     allAccounts.remove(acc);
111                                     return;
112                             }
113                     }
114
115                     throw new AccountIDNotRecognisedException("Given ID not found in list of
116 Accounts.");
117
118             }
119
120             @Override
121             public void removeAccount(String handle) throws HandleNotRecognisedException {
122
123
124                     for (Account acc : allAccounts) {
125                             if (acc.getHandle() == handle) {
```

```
126                                     //remove all posts of that user
127                                     for(Post post: acc.getPosts()) {
128
129                                             try {
130                                                     deletePost(post.getID());
131                                             } catch (PostIDNotRecognisedException e) {
132                                                     e.printStackTrace();
133                                             }
134
135                                     }
136                                     // remove the account
137                                     allAccounts.remove(acc);
138                                     return;
139                             }
140                     }
141
142                     throw new HandleNotRecognisedException("Given ID not found in list of
143     Accounts.");
144
145             }
146
147             @Override
148             public void changeAccountHandle(String oldHandle, String newHandle)
149                             throws HandleNotRecognisedException, IllegalHandleException,
150     InvalidHandleException {
151
152                                     // checks if the new handle is valid
153                                     if ((newHandle.length()) == 0 || (newHandle.length() > 30)) {
154                                             throw new InvalidHandleException("New handle must be
155     between 0 and 30 characters");
156                                     }
157
```

```
158                                     // checks if the new handle is valid
159                                     if (newHandle.contains(" ")) {
160                                             throw new InvalidHandleException("New handle can't
161     contain whitespace.");
162                                     }
163
164                                     for (Account acc : allAccounts) {
165                                             if (acc.getHandle() == oldHandle) {
166                                                     // sets the new handle
167                                                     acc.setHandle(newHandle);
168                                                     return;
169                                             }
170                                     }
171
172                                     throw new HandleNotRecognisedException("Old handle not found
173     in list of accounts.");
174
175             }
176
177             @Override
178             public void updateAccountDescription(String handle, String description) throws
179     HandleNotRecognisedException {
180                     for (Account acc : allAccounts) {
181                             if (acc.getHandle() == handle) {
182                                     // sets the description
183                                     acc.setDescription(description);
184                                     return;
185                             }
186                     }
187
188                     throw new HandleNotRecognisedException("Handle not found in list of accounts.");
189             }
```

```
190
191             @Override
192             public String showAccount(String handle) throws HandleNotRecognisedException {
193
194                     String outputString = "";
195                     for (Account acc : allAccounts) {
196                             if (acc.getHandle() == handle) {
197                                     // adds the id to the string
198                                     outputString += "\nID: " + Integer.toString(acc.getID());
199                                     if(acc.getHandle() != null) {
200                                             // adds the handle to the string
201                                             outputString += "\nHandle: " + acc.getHandle();
202                                     }
203                                     if (acc.getDescription() != null) {
204                                             // adds the description to the string
205                                             outputString += "\nDescription: " + acc.getDescription();
206                                     }
207                                     // add the post count to the string
208                                     outputString += "\nPost Count: " + acc.getPosts().size();
209                                     return outputString;
210                             }
211
212                     }
213                     throw new HandleNotRecognisedException("Handle not found in list of accounts.");
214
215             }
216
217             @Override
218             public int createPost(String handle, String message) throws HandleNotRecognisedException,
219     InvalidPostException {
220
```

```
221                     // checks id the message is valid
222                     if (message.length() == 0 || message.length() > 100) {
223                             throw new InvalidPostException("Message must be between 1 and 100
224     characters.");
225                     }
226
227                     for (Account acc : allAccounts) {
228                             if (acc.getHandle() == handle) {
229                                     int id = ++postIDTally;
230                                     // creates the post
231                                     Post post = new Post(message, handle);
232                                     // sets all the attributes
233                                     post.setID(id);
234                                     acc.getPosts().add(post);
235                                     allPosts.add(post);
236                                     return post.getID();
237                             }
238                     }
239
240                     //if here is reached, handle must be invalid
241                     throw new HandleNotRecognisedException("Handle not found in list of accounts");
242
243             }
244
245             @Override
246             public int endorsePost(String handle, int id) throws HandleNotRecognisedException,
247     PostIDNotRecognisedException, NotActionablePostException {
248
249                     // checking the handle
250                     Account accountEndorsing = null;
251                     Boolean isValidAccount = false;
252                     for(Account acc: allAccounts) {
```

```
253                         if(acc.getHandle() == handle) {
254                             isValidAccount = true;
255                             accountEndorsing = acc;
256                         }
257                     }
258                     if(!isValidAccount) {
259                         throw new HandleNotRecognisedException("Handle not found in list of
260 accounts");
261                     }
262
263                     // checking the post id
264                     Boolean isValidPost = false;
265                     Post endorsed_post = null;
266                     for(Post post: allPosts) {
267                         if(post.getID() == id) {
268                             if (post.getClass() == Endorsement.class) {
269                                 throw new NotActionablePostException("You cannot
270 endorse an endorsement!");
271                             }
272                             endorsed_post = post;
273                             isValidPost = true;
274                         }
275                     }
276                     if(!isValidPost) {
277                         throw new PostIDNotRecognisedException("ID not found in the list of
278 posts");
279                     }
280
281
282                     Endorsement endorsement = new Endorsement(handle, id);
283
284                     // create the endorsement content and set it.
```

```
285                     String endorsee = "";
286                     for (Post i : allPosts) {
287                             if (i.getID() == id) {
288                                     endorsee = i.getHandle();
289                             }
290                     }
291
292                     String endorsement_content = "EP@" + endorsee + ": " +
293     endorsed_post.getContent();
294                     endorsement.setContent(endorsement_content);
295                     // set the id.
296                     int idToAdd = ++postIDTally;
297                     endorsement.setID(idToAdd);
298                     // add the endorsement as a child of the post it is endorsing.
299                     Post child = endorsement;
300                     // endorsed_post.addChild(endorsement);
301                     if (endorsed_post.getClass() == Comment.class) {
302                             endorsed_post = (Comment)endorsed_post;
303                     }
304                     endorsed_post.addChild(child);
305                     // add the endorsement to the list of posts.
306                     allPosts.add(endorsement);
307
308                     accountEndorsing.getPosts().add(endorsement);
309
310
311                     return id;
312             }
313
314             @Override
315             public int commentPost(String handle, int id, String message) throws
316     HandleNotRecognisedException,
```

```
317                             PostIDNotRecognisedException, NotActionablePostException,
318     InvalidPostException {
319
320             // check if the handle is correct
321             Boolean isValidAccount = false;
322             for(Account acc: allAccounts) {
323                     if(acc.getHandle() == handle){
324                             isValidAccount = true;
325                             break;
326                     }
327             }
328             if(isValidAccount == false) {
329                     throw new HandleNotRecognisedException("Handle not found in list of
330     accounts");
331             }
332
333             //check if post id is correct
334             Boolean isValidPostID = false;
335             Post post = null;
336             for(Post p: allPosts) {
337                     // checks
338                     if(p.getID() == id) {
339                             if (p.getClass() == Endorsement.class) {
340                                     throw new NotActionablePostException("You cannt
341     comment this post!");
342                             }
343                             isValidPostID = true;
344                             post = p;
345                             break;
346                     }
347             }
348             // throws the Exception
```

```
349                     if(isValidPostID == false) {
350                             throw new PostIDNotRecognisedException("ID not found in the list of
351 posts");
352                     }
353
354                     // creates the Comment
355                     Comment comment = new Comment(handle, id, message);
356
357                     // sets the attributes
358                     comment.setID(++postIDTally);
359
360                     post.addChild(comment);
361                     allPosts.add(comment);
362
363                     return comment.getID();
364             }
365
366             @Override
367             public void deletePost(int id) throws PostIDNotRecognisedException {
368
369                     for (Post post : allPosts) {
370                             if(post.getID() == id) {
371                                     // checks if the post has children
372                                     if(post.getChildren().size() > 0) {
373                                             Post newPost = new Post(null ,"The original content was
374 removed from the system and is no longer available.");
375                                             newPost.setID(-1);
376                                             for(Post p: post.getChildren()) {
377                                                     // removes the endorsements
378                                                     if(p.getClass() == Endorsement.class) {
379                                                             Endorsement endToRemove =
380 (Endorsement)p;
```

```
381
382                                                     allPosts.remove(endToRemove);
383
384                                             }
385                                             else {
386                                                     // changes the Comments' parent
387                                                     if(p.getClass() == Comment.class) {
388                                                             p.setHandle("Unavailable");
389                                                             p.setContent("The original content
390     was removed from the system and is no longer available.");
391                                                             Comment c = (Comment)p;
392                                                             c.setPostID(newPost.getID());
393                                                             //deletePost(p.getID());
394                                                     }
395                                             }
396                                     }
397
398                                     }
399                                     allPosts.remove(post);
400                                     return;
401                             }
402                     }
403
404
405                     throw new PostIDNotRecognisedException("ID not found in the list of posts");
406
407
408
409             }
410
411             @Override
```

```
412             public String showIndividualPost(int id) throws PostIDNotRecognisedException {
413                     String outputString = "";
414                     //loop through all posts and find matching ID
415                     for (Post i : allPosts) {
416                             if (i.getID() == id) {
417                                     //if id matches, construct string to return and then return it
418                                     outputString += "\nID: " + id;
419                                     outputString += "\nAccount: " + i.getHandle();
420                                     outputString += "\nNo. endorsements: " +
421     i.getEndorsements().size();
422                                     //check if each child is a comment
423                                     outputString += " | No. comments: " + i.getComments().size();
424                                     outputString += "\n" + i.getContent();
425                                     return outputString;
426                             }
427                     }
428                     //if here is reached, then the id must be wrong
429                     throw new PostIDNotRecognisedException("Post ID not recognised.");
430             }
431
432             @Override
433             public StringBuilder showPostChildrenDetails(int id)
434                             throws PostIDNotRecognisedException, NotActionablePostException {
435
436
437                     boolean isValid = false;
438         StringBuilder childrenDetails = new StringBuilder();
439         for (Post post : allPosts) {
440           if (post.getID() == id) {
441             isValid = true;
442             childrenDetails.append(showIndividualPost(id) + "|" + "\n" + "| > ");
```

```
443             }
444         }
445         if (isValid) {
446             counter = 0;
447             while (counter < allPosts.size()) {
448                                         Post post = allPosts.get(counter);
449                 if (post.getID() == id) {
450                     for (Post child : post.getChildren()) {
451                                                             if(child.getClass() == Comment.class) {
452                                                                     // recursively adds to the StringBuilder
453
454                 childrenDetails.append(showPostChildrenDetails(child.getID()) + "\n" + "|" + "\n" + "| > ");
455                                                             }
456                     }
457                 }
458                 counter++;
459             }
460             return childrenDetails;
461         } else {
462             throw new PostIDNotRecognisedException("post id does not exist");
463                         }
464             }
465
466             @Override
467             public int getNumberOfAccounts() {
468                         // return the length of the arraylist of accounts
469                         return allAccounts.size();
470             }
471
472             @Override
473             public int getTotalOriginalPosts() {
```

```
474                 counter = 0;
475                 for (Post i : allPosts) {
476                         //if post isn't an endorsement or a comment, must be an original one
477                         //so add one to the counter
478                         if ((i.getClass() != Endorsement.class) && (i.getClass() != Comment.class)) {
479                                 counter++;
480                         }
481                 }
482                 return counter;
483         }
484
485         @Override
486         public int getTotalEndorsmentPosts() {
487                 counter = 0;
488                 for (Post i : allPosts) {
489                         //check if an endorsement, or just a normal post
490                         if (i.getClass() == Endorsement.class) {
491                                 counter++;
492                         }
493                 }
494                 return counter;
495         }
496
497         @Override
498         public int getTotalCommentPosts() {
499                 counter = 0;
500                 for (Post i : allPosts) {
501                         //check if a comment, or just a normal post
502                         if (i.getClass() == Comment.class) {
503                                 counter++;
504                         }
```

```
505                     }
506                     return counter;
507             }
508
509             @Override
510             public int getMostEndorsedPost() {
511                     //Loops through all posts and keeps a track of the one with the longest
512     endorsements ArrayList.
513                     Post mostEndorsedPost = allPosts.get(0);
514                     for (Post i : allPosts) {
515                             if (i.getEndorsements().size() > mostEndorsedPost.getEndorsements().size())
516     {
517                                     mostEndorsedPost = i;
518                             }
519                     }
520                     return mostEndorsedPost.getID();
521             }
522
523             @Override
524             public int getMostEndorsedAccount() {
525                     // Find account with the longest arraylist of endorsements.
526
527                     //pick the first account in the list and assume it's the most endorsed
528                     Account mostEndorsedAccount = allAccounts.get(0);
529                     int endorseTotal;
530
531                     for (Account i : allAccounts) {
532                             //need to count up the endorsements on each post
533                             endorseTotal = 0;
534                             int mostEndorsements = 0;
535                             //for each of this user's posts, count and sum the endorsements they have
536                             for (Post j : i.getPosts()) {
```

```
537                                     endorseTotal += j.getEndorsements().size();
538                             }
539                             //if this total is bigger than the current most, update most endorsed account
540                             if (endorseTotal > mostEndorsements) {
541                                     mostEndorsedAccount = i;
542                                     mostEndorsements = endorseTotal;
543                             }
544                     }
545
546                     return mostEndorsedAccount.getID();
547             }
548
549             @Override
550             public void erasePlatform() {
551                     // clear all the attributes
552                     allAccounts.clear();
553                     allPosts.clear();
554                     counter = 0;
555             }
556
557             @Override
558             public void savePlatform(String filename) throws IOException {
559                     try {
560                             FileOutputStream fileOut = new FileOutputStream(filename);
561                             ObjectOutputStream objectOut = new ObjectOutputStream(fileOut);
562                             ArrayList<Object> output = new ArrayList<>();
563                             output.add(allAccounts);
564                             output.add(allPosts);
565                             objectOut.writeObject(output);
566                             objectOut.close();
567                     } catch(FileNotFoundException e) {
```

```
568                             e.printStackTrace();
569                     } catch(IOException e) {
570                             e.printStackTrace();
571                     }
572             }
573
574             @Override
575             public void loadPlatform(String filename) throws IOException, ClassNotFoundException {
576                     ArrayList<Object> read = null;
577                     try {
578                             FileInputStream fileIn = new FileInputStream(filename);
579                             ObjectInputStream objectIn = new ObjectInputStream(fileIn);
580                             read = (ArrayList<Object>) objectIn.readObject();
581                             objectIn.close();
582                             allAccounts = (ArrayList<Account>) read.get(0);
583                             allPosts = (ArrayList<Post>) read.get(1);
584                     } catch(FileNotFoundException e) {
585                             e.printStackTrace();
586                     } catch(IOException e) {
587                             e.printStackTrace();
588                     } catch(ClassNotFoundException e) {
589                             e.printStackTrace();
590                     }
591
592             }
593
594
595
596     }
```