

```

1  //***** Library.java *****
2
3  import java.util.LinkedList;
4  import java.util.ArrayList;
5  import java.util.Scanner;
6
7  class Author {
8      public String name;
9      public BookList<Book> books = new BookList<Book>();
10     public Author() {
11     }
12     public boolean equals(Object node) {
13         return name.equals(((Author) node).name);
14     }
15     public void display() {
16         System.out.println(name);
17         books.display();
18     }
19 }
20
21 class Book {
22     public String title;
23     public Patron patron = null;
24     public Book() {
25     }
26     public boolean equals(Object node) {
27         return title.equals(((Book) node).title);
28     }
29     public String toString() {
30         return " * " + title + (patron != null ? " - checked out to " + patron.name : "") + "\n";

```

```
31     }
32 }
33
34 class CheckedOutBook {
35     public Author author = null;
36     public Book book = null;
37     public CheckedOutBook() {
38     }
39     public boolean equals(Object node) {
40         return book.title.equals(((CheckedOutBook) node).book.title) && author.name.equals(((CheckedOutBook)
41             node).author.name);
42     }
43     public String toString() {
44         return " * " + author.name + ", " + book.title + "\n";
45     }
46 }
47
48 class Patron {
49     public String name;
50     public BookList<CheckedOutBook> books = new BookList<CheckedOutBook>();
51     public Patron() {
52     }
53     public boolean equals(Object node) {
54         return name.equals(((Patron) node).name);
55     }
56     public void display() {
57         if (!books.isEmpty()) {
58             System.out.println(name + " has the following books:");
59             books.display();
60         }
```

```
61         else System.out.print(name + " has no books");
62     }
63 }
64
65 class AuthorList extends LinkedList<Author> {
66     static final long serialVersionUID = 123;
67     public AuthorList() {
68         super();
69     }
70     public void display() {
71         Object[] authors = toArray();
72         for (int i = 0; i < authors.length; i++)
73             ((Author)authors[i]).display();
74     }
75 }
76
77 class BookList<T> extends LinkedList<T> {
78     static final long serialVersionUID = 124;
79     public BookList() {
80         super();
81     }
82     public void display() {
83         for (int i = 0; i < size(); i++)
84             System.out.print(get(i));
85     }
86 }
87
88 class PatronList extends LinkedList<Patron> {
89     static final long serialVersionUID = 125;
90     public PatronList() {
```

```

91         super();
92     }
93     public void display() {
94         for (java.util.Iterator it = iterator(); it.hasNext(); )
95             ((Patron)it.next()).display();
96     }
97 }
98
99 class Library {
100     private ArrayList<AuthorList>
101         catalog = new ArrayList<AuthorList>('Z'+1);
102     private ArrayList<PatronList>
103         people = new ArrayList<PatronList>('Z'+1);
104     private String input;
105     Scanner kb = new Scanner(System.in);
106     public Library() {
107         for (int i = 0; i <= 'Z'; i++) {
108             catalog.add(i,new AuthorList());
109             people.add(i,new PatronList());
110         }
111     }
112     private String getString(String msg) {
113         System.out.print(msg + " ");
114         System.out.flush();
115         input = kb.nextLine();
116         return input.substring(0,1).toUpperCase() + input.substring(1);
117     }
118     private void status() {
119         System.out.println("Library has the following books:\n ");
120         for (int i = 'A'; i <= 'Z'; i++)

```

```
121         if (catalog.get(i).size() > 0)
122             catalog.get(i).display();
123     System.out.println("\nThe following people are using the library:\n ");
124     for (int i = 'A'; i <= 'Z'; i++)
125         if (people.get(i).size() > 0)
126             people.get(i).display();
127 }
128 private void includeBook() {
129     Author newAuthor = new Author();
130     int oldAuthor;
131     Book newBook = new Book();
132     newAuthor.name = getString("Enter author's name:");
133     newBook.title = getString("Enter the title of the book:");
134     oldAuthor = catalog.get(newAuthor.name.charAt(0)).indexOf(newAuthor);
135     if (oldAuthor == -1) {
136         newAuthor.books.add(newBook);
137         catalog.get(newAuthor.name.charAt(0)).add(newAuthor);
138     }
139     else (catalog.get(newAuthor.name.charAt(0)).get(oldAuthor)).books.add(newBook);
140 }
141 private void checkOutBook() {
142     Patron patron = new Patron(), patronRef;
143     Author author = new Author(), authorRef = new Author();
144     Book book = new Book();
145     int patronIndex, bookIndex = -1, authorIndex = -1;
146     patron.name = getString("Enter patron's name:");
147     while (authorIndex == -1) {
148         author.name = getString("Enter author's name:");
149         authorIndex = catalog.get(author.name.charAt(0)).indexOf(author);
150         if (authorIndex == -1)
```

```

151         System.out.println("Misspelled author's name");
152     }
153     while (bookIndex == -1) {
154         book.title = getString("Enter the title of the book:");
155         authorRef = catalog.get(author.name.charAt(0)).get(authorIndex);
156         bookIndex = authorRef.books.indexOf(book);
157         if (bookIndex == -1)
158             System.out.println("Misspelled title");
159     }
160     Book bookRef = authorRef.books.get(bookIndex);
161     CheckedOutBook bookToCheckOut = new CheckedOutBook();
162     bookToCheckOut.author = authorRef;
163     bookToCheckOut.book = bookRef;
164     patronIndex = people.get(patron.name.charAt(0)).indexOf(patron);
165     if (patronIndex == -1) { // a new patron in the library;
166         patron.books.add(bookToCheckOut);
167         people.get(patron.name.charAt(0)).add(patron);
168         bookRef.patron = people.get(patron.name.charAt(0)).getFirst();
169     }
170     else {
171         patronRef = people.get(patron.name.charAt(0)).get(patronIndex);
172         patronRef.books.add(bookToCheckOut);
173         bookRef.patron = patronRef;
174     }
175 }
176 private void returnBook() {
177     Patron patron = new Patron();
178     Book book = new Book();
179     Author author = new Author(), authorRef = new Author();
180     int patronIndex = -1, bookIndex = -1, authorIndex = -1;

```



```
211         "3. Return a book\n4. Status\n5. Exit\n" +
212         "Your option:").charAt(0);
213     switch (option) {
214         case '1': includeBook(); break;
215         case '2': checkOutBook(); break;
216         case '3': returnBook(); break;
217         case '4': status(); break;
218         case '5': return;
219         default: System.out.println("Wrong option, try again.");
220     }
221 }
222 }
223 public static void main(String args[]) {
224     (new Library()).run();
225 }
226 }
```