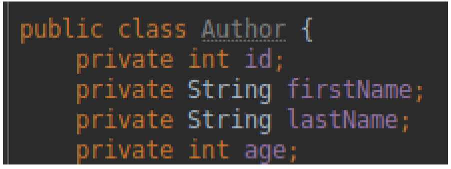
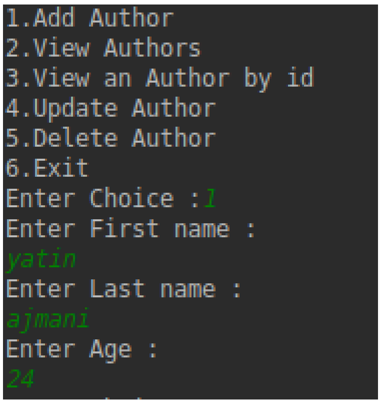
Exercise

1. Create a class Author with instance variables firstName, lastName and age.

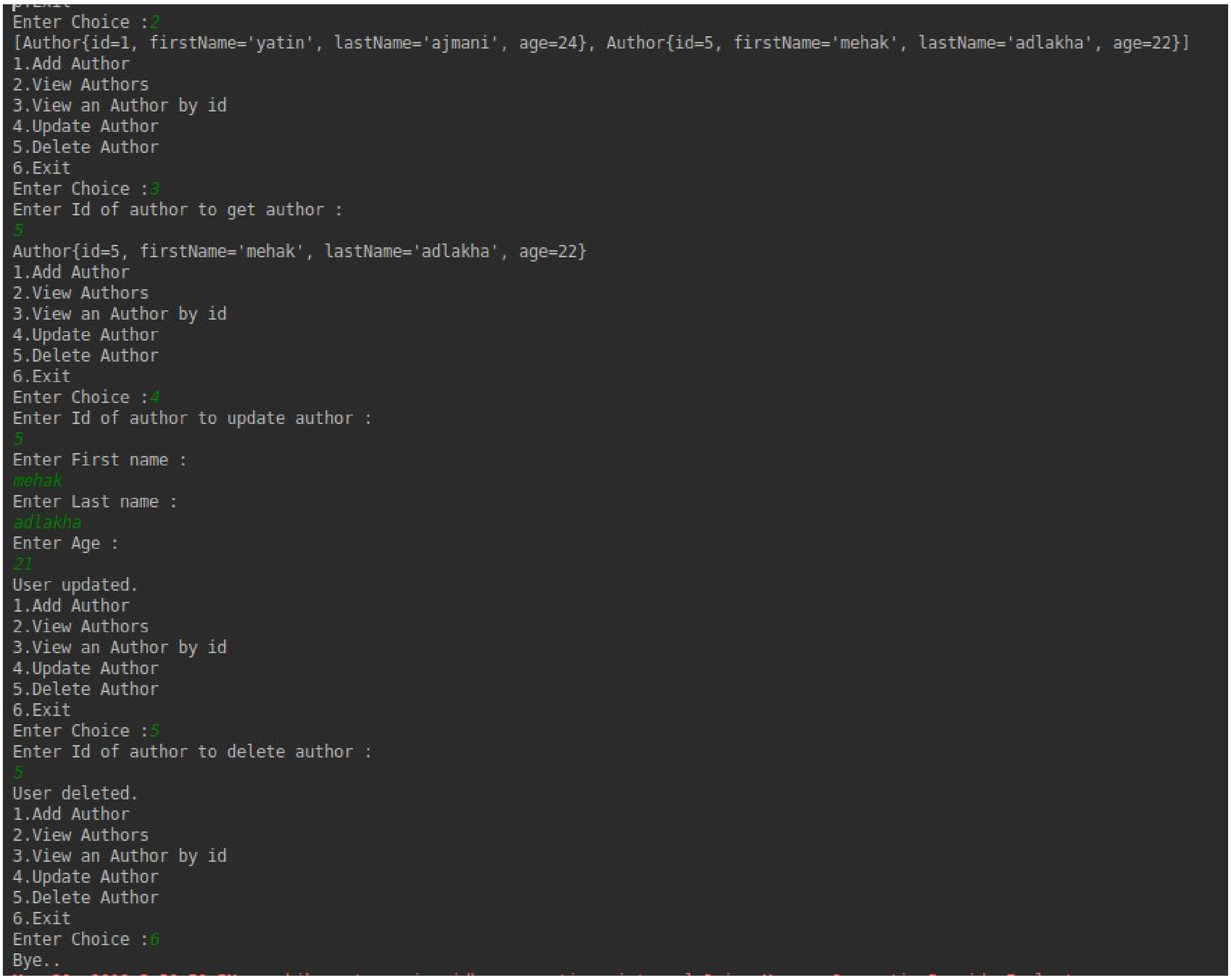


```
public class Author {
    private int id;
    private String firstName;
    private String lastName;
    private int age;
```

2. Perform CRUD operation for Author class.

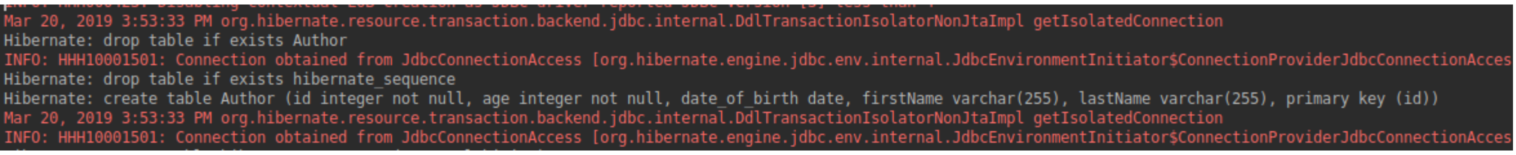


```
1.Add Author
2.View Authors
3.View an Author by id
4.Update Author
5.Delete Author
6.Exit
Enter Choice :1
Enter First name :
yatin
Enter Last name :
ajmani
Enter Age :
24
```



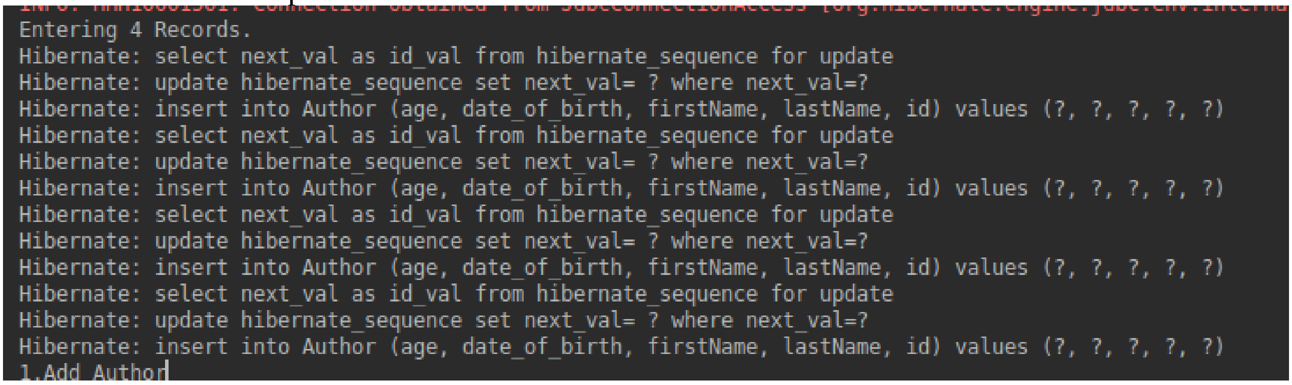
```
Enter Choice :2
[Author{id=1, firstName='yatin', lastName='ajmani', age=24}, Author{id=5, firstName='mehak', lastName='adlakha', age=22}]
1.Add Author
2.View Authors
3.View an Author by id
4.Update Author
5.Delete Author
6.Exit
Enter Choice :3
Enter Id of author to get author :
5
Author{id=5, firstName='mehak', lastName='adlakha', age=22}
1.Add Author
2.View Authors
3.View an Author by id
4.Update Author
5.Delete Author
6.Exit
Enter Choice :4
Enter Id of author to update author :
5
Enter First name :
mehak
Enter Last name :
adlakha
Enter Age :
21
User updated.
1.Add Author
2.View Authors
3.View an Author by id
4.Update Author
5.Delete Author
6.Exit
Enter Choice :5
Enter Id of author to delete author :
5
User deleted.
1.Add Author
2.View Authors
3.View an Author by id
4.Update Author
5.Delete Author
6.Exit
Enter Choice :6
Bye..
```

3. Use hbm2ddl create to introduce Date of Birth for Author.



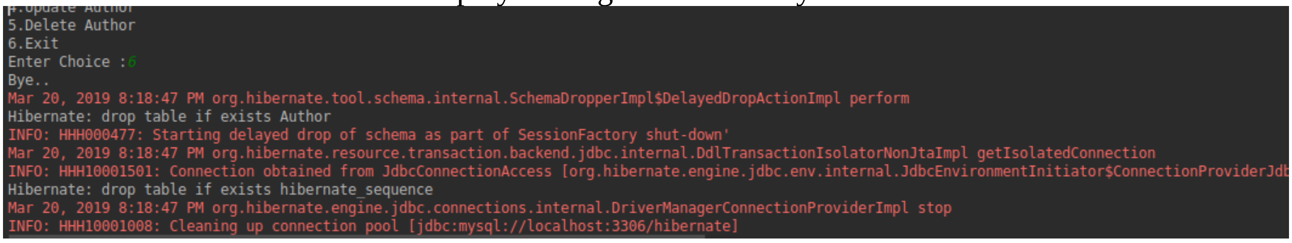
```
Mar 20, 2019 3:53:33 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
Hibernate: drop table if exists Author
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAcces
Hibernate: drop table if exists hibernate_sequence
Hibernate: create table Author (id integer not null, age integer not null, date_of_birth date, firstName varchar(255), lastName varchar(255), primary key (id))
Mar 20, 2019 3:53:33 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAcces
```

4. Use hbm2dll update to insert at least 4 records for Author.



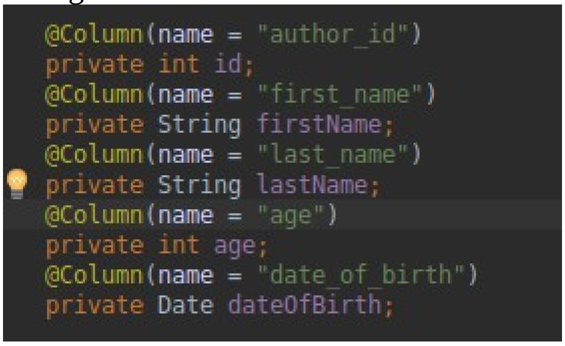
```
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.interna
Entering 4 Records.
Hibernate: select next_val as id_val from hibernate_sequence for update
Hibernate: update hibernate_sequence set next_val= ? where next_val=?
Hibernate: insert into Author (age, date_of_birth, firstName, lastName, id) values (?, ?, ?, ?, ?)
Hibernate: select next_val as id_val from hibernate_sequence for update
Hibernate: update hibernate_sequence set next_val= ? where next_val=?
Hibernate: insert into Author (age, date_of_birth, firstName, lastName, id) values (?, ?, ?, ?, ?)
Hibernate: select next_val as id_val from hibernate_sequence for update
Hibernate: update hibernate_sequence set next_val= ? where next_val=?
Hibernate: insert into Author (age, date_of_birth, firstName, lastName, id) values (?, ?, ?, ?, ?)
Hibernate: select next_val as id_val from hibernate_sequence for update
Hibernate: update hibernate_sequence set next_val= ? where next_val=?
Hibernate: insert into Author (age, date_of_birth, firstName, lastName, id) values (?, ?, ?, ?, ?)
1.Add Author
```

5. Perform hbm2dll create-drop by closing session factory.



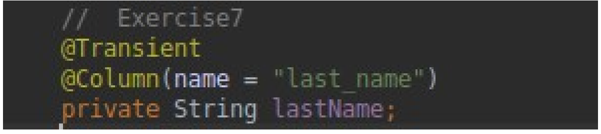
```
4.Update Author
5.Delete Author
6.Exit
Enter Choice :6
Bye..
Mar 20, 2019 8:18:47 PM org.hibernate.tool.schema.internal.SchemaDropperImpl$DelayedDropActionImpl perform
Hibernate: drop table if exists Author
INFO: HHH000477: Starting delayed drop of schema as part of SessionFactory shut-down'
Mar 20, 2019 8:18:47 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdb
Hibernate: drop table if exists hibernate_sequence
Mar 20, 2019 8:18:47 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl stop
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/hibernate]
```

6. Rename all the fields using column annotation.



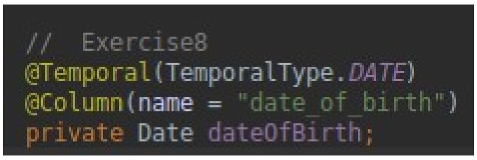
```
@Column(name = "author_id")
private int id;
@Column(name = "first_name")
private String firstName;
@Column(name = "last_name")
private String lastName;
@Column(name = "age")
private int age;
@Column(name = "date_of_birth")
private Date dateOfBirth;
```

7. Mark lastName as @Transient.



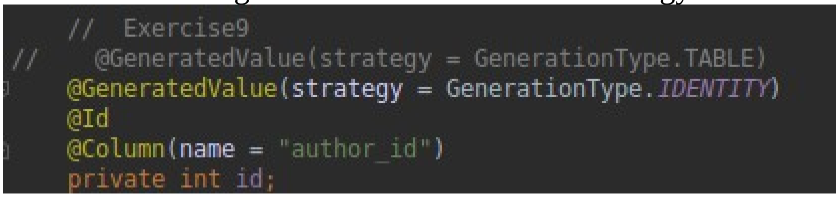
```
//  Exercise7
@Transient
@Column(name = "last_name")
private String lastName;
```

8. Use @Temporal for date of birth of Author.



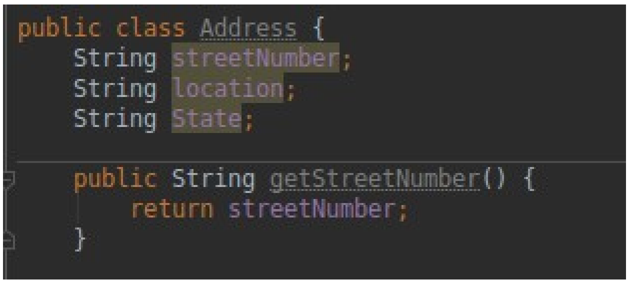
```
//  Exercise8
@Temporal(TemporalType.DATE)
@Column(name = "date_of_birth")
private Date dateOfBirth;
```

9. Generate Id for Author Using IDENTITY and TABLE starategy.



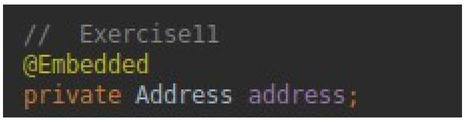
```
    //  Exercise9
//    @GeneratedValue(strategy = GenerationType.TABLE)
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Id
    @Column(name = "author_id")
    private int id;
```

10. Create a class Address for Author with instance variables streetNumber, location, State.



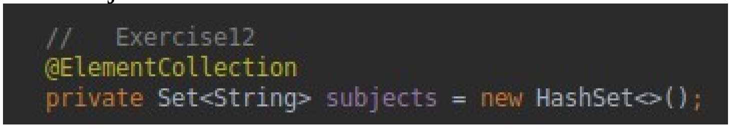
```
public class Address {
    String streetNumber;
    String location;
    String State;

    public String getStreetNumber() {
        return streetNumber;
    }
```

11. Create instance variable of Address class inside Author class and save it as embedded object.



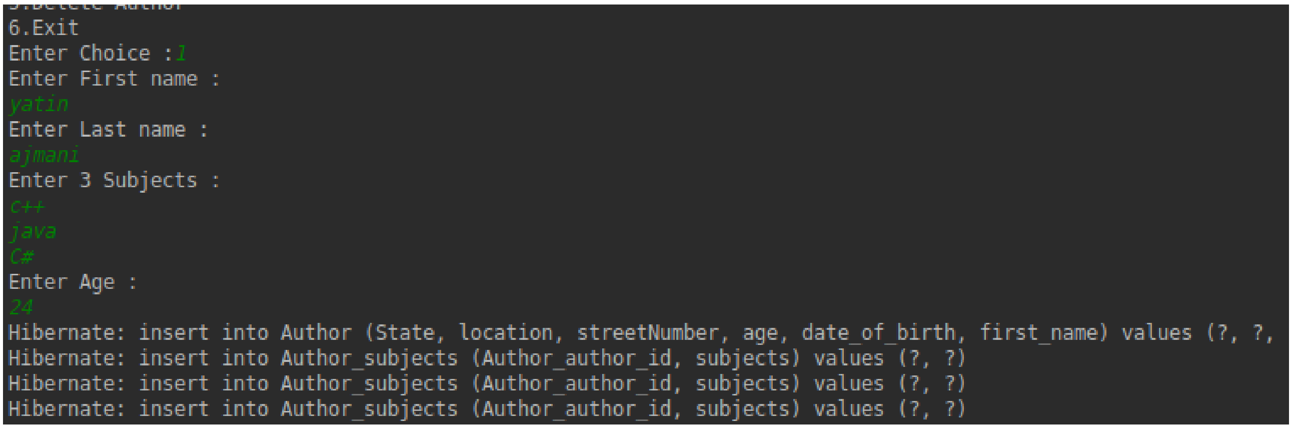
```
//  Exercise11
@Embedded
private Address address;
```

12. Introduce a List of subjects for author.



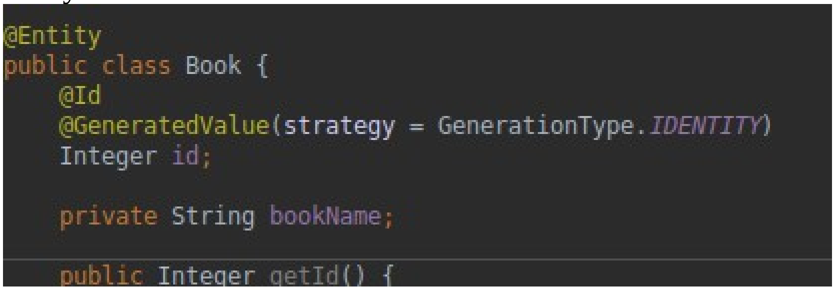
```
//   Exercise12
@ElementCollection
private Set<String> subjects = new HashSet<>();
```

13. Persist 3 subjects for each author.



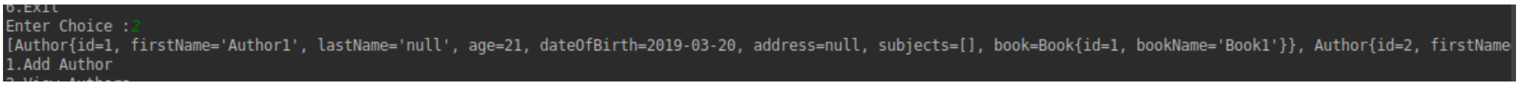
```
6.Exit
Enter Choice :1
Enter First name :
yatin
Enter Last name :
ajmani
Enter 3 Subjects :
C++
java
C#
Enter Age :
24
Hibernate: insert into Author (State, location, streetNumber, age, date_of_birth, first_name) values (?, ?,
Hibernate: insert into Author_subjects (Author_author_id, subjects) values (?, ?)
Hibernate: insert into Author_subjects (Author_author_id, subjects) values (?, ?)
Hibernate: insert into Author_subjects (Author_author_id, subjects) values (?, ?)
```

14. Create an Entity book with an instance variable bookName.

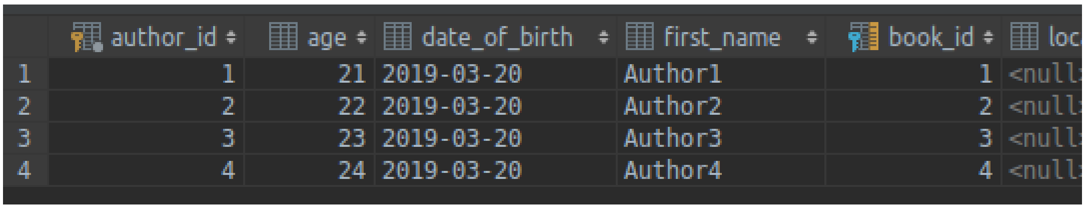


```
@Entity
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    Integer id;

    private String bookName;

    public Integer getId() {
```

15. Implement One to One mapping between Author and Book.

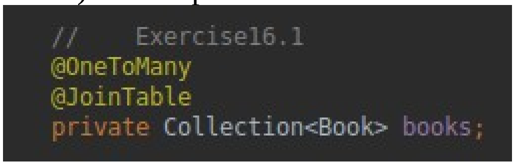


```
6.Exit
Enter Choice :2
[Author{id=1, firstName='Author1', lastName='null', age=21, dateOfBirth=2019-03-20, address=null, subjects=[], book=Book{id=1, bookName='Book1'}}, Author{id=2, firstName
1.Add Author
```

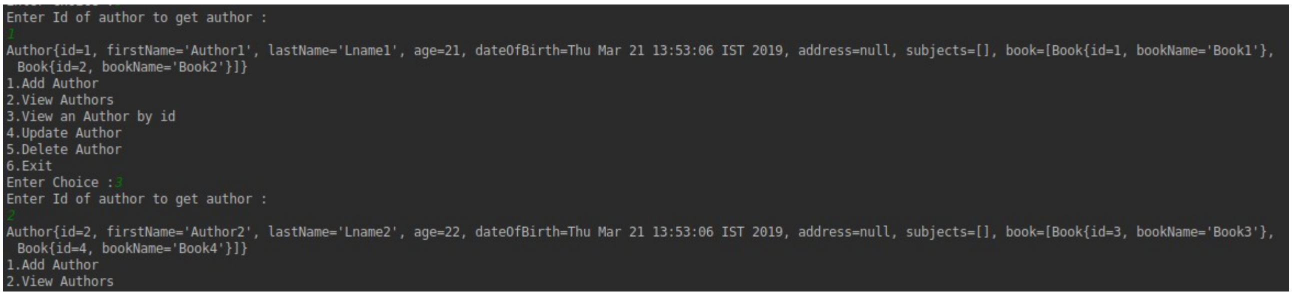


	author_id	age	date_of_birth	first_name	book_id	loc
1	1	21	2019-03-20	Author1	1	<null
2	2	22	2019-03-20	Author2	2	<null
3	3	23	2019-03-20	Author3	3	<null
4	4	24	2019-03-20	Author4	4	<null

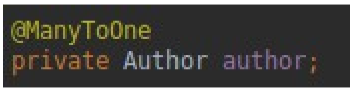
16. Implement One to Many Mapping between Author and Book(Unidirectional, BiDirectional and without additional table) and implement cascade save.



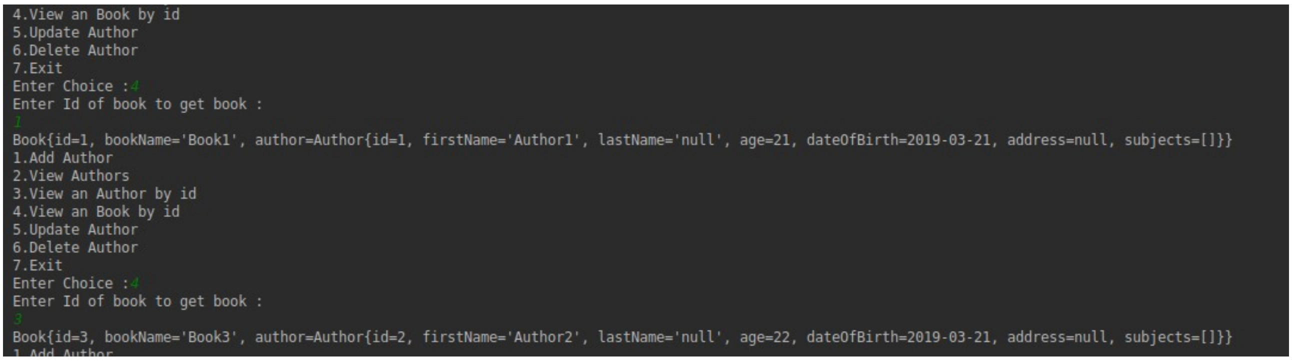
```
//    Exercise16.1
@OneToMany
@JoinTable
private Collection<Book> books;
```



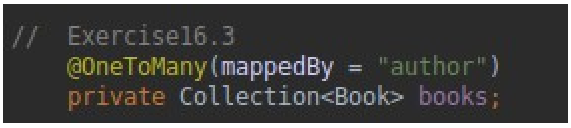
```
Enter Id of author to get author :
1
Author{id=1, firstName='Author1', lastName='Lname1', age=21, dateOfBirth=Thu Mar 21 13:53:06 IST 2019, address=null, subjects=[], book=[Book{id=1, bookName='Book1'},
 Book{id=2, bookName='Book2'}]}
1.Add Author
2.View Authors
3.View an Author by id
4.Update Author
5.Delete Author
6.Exit
Enter Choice :3
Enter Id of author to get author :
2
Author{id=2, firstName='Author2', lastName='Lname2', age=22, dateOfBirth=Thu Mar 21 13:53:06 IST 2019, address=null, subjects=[], book=[Book{id=3, bookName='Book3'},
 Book{id=4, bookName='Book4'}]}
1.Add Author
2.View Authors
```



```
@ManyToOne
private Author author;
```

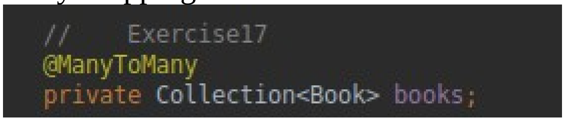


```
4.View an Book by id
5.Update Author
6.Delete Author
7.Exit
Enter Choice :4
Enter Id of book to get book :
1
Book{id=1, bookName='Book1', author=Author{id=1, firstName='Author1', lastName='null', age=21, dateOfBirth=2019-03-21, address=null, subjects=[]}}
1.Add Author
2.View Authors
3.View an Author by id
4.View an Book by id
5.Update Author
6.Delete Author
7.Exit
Enter Choice :4
Enter Id of book to get book :
3
Book{id=3, bookName='Book3', author=Author{id=2, firstName='Author2', lastName='null', age=22, dateOfBirth=2019-03-21, address=null, subjects=[]}}
1.Add Author
```

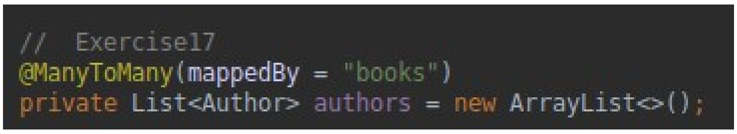


```
//  Exercise16.3
    @OneToMany(mappedBy = "author")
    private Collection<Book> books;
```

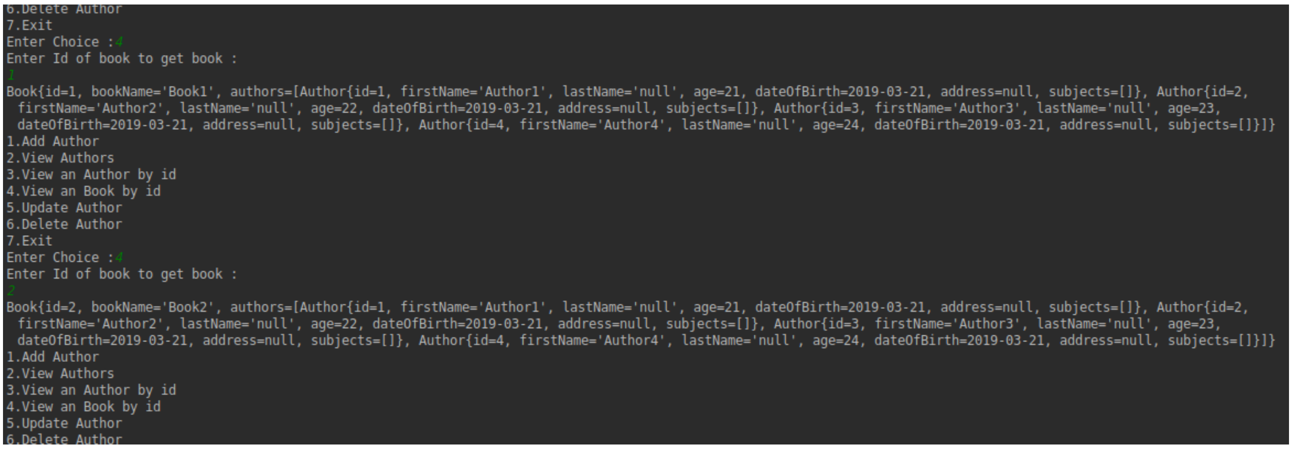
17. Implement Many to Many Mapping between Author and Book.



```
//    Exercise17
@ManyToMany
private Collection<Book> books;
```



```
//  Exercise17
@ManyToMany(mappedBy = "books")
private List<Author> authors = new ArrayList<>();
```



```
6.Delete Author
7.Exit
Enter Choice :4
Enter Id of book to get book :
1
Book{id=1, bookName='Book1', authors=[Author{id=1, firstName='Author1', lastName='null', age=21, dateOfBirth=2019-03-21, address=null, subjects=[]}, Author{id=2,
 firstName='Author2', lastName='null', age=22, dateOfBirth=2019-03-21, address=null, subjects=[]}, Author{id=3, firstName='Author3', lastName='null', age=23,
 dateOfBirth=2019-03-21, address=null, subjects=[]}, Author{id=4, firstName='Author4', lastName='null', age=24, dateOfBirth=2019-03-21, address=null, subjects=[]}]}
1.Add Author
2.View Authors
3.View an Author by id
4.View an Book by id
5.Update Author
6.Delete Author
7.Exit
Enter Choice :4
Enter Id of book to get book :
2
Book{id=2, bookName='Book2', authors=[Author{id=1, firstName='Author1', lastName='null', age=21, dateOfBirth=2019-03-21, address=null, subjects=[]}, Author{id=2,
 firstName='Author2', lastName='null', age=22, dateOfBirth=2019-03-21, address=null, subjects=[]}, Author{id=3, firstName='Author3', lastName='null', age=23,
 dateOfBirth=2019-03-21, address=null, subjects=[]}, Author{id=4, firstName='Author4', lastName='null', age=24, dateOfBirth=2019-03-21, address=null, subjects=[]}]}
1.Add Author
2.View Authors
3.View an Author by id
4.View an Book by id
5.Update Author
6.Delete Author
```