QX.

(a) In the context of `Stream`s from `java.util.stream`, explain using examples the concepts of (i) *intermediate* operations, (ii) *terminal* operations, and (iii) *short-circuiting* operations.

6 MARKS

(b) Suppose you have a collection of `AbstractSocialMediaPost` objects (i.e., `List<AbstractSocialMediaPost> posts`) where a post has the following interface:

```
public interface AbstractSocialMediaPost {
     public String getPostText();
     public User getPostAuthor();
     public int getLikes();
     public int getShares();
}
```

Illustrate using Java code how you would use the Java Streams API to

(i) Get a `List<String>` of the unique usernames of the authors of the posts in the collection. You can assume the `User` interface has a `getUsername()` method that returns a `String` representation of the username.

5 MARKS

(ii) Get a `List<AbstractSocialMediaPost>` of 3 posts that have over 50 shares.

5 MARKS

(iii) Find the first post with greater than 50 "likes", and print it to the console if one exists. You can assume that implementations of `AbstractSocialMediaPost` have an appropriate `toString()` implementation.

6 MARKS

(iv) Get the total number of "likes" across all posts.

3 MARKS

Other potential questions:

Explain using an example what the flatMap operation does in Streams processing.

Explain the difference between stateless and stateful stream operations.

For stateful operations, explain the terms bounded and unbounded.