# Exercises Module 16 - Collections

## Task 1

Continue working on the Amazon project. Replace all arrays in the project with generic lists.

Create one generic List of Books, add some books to this list, and iterate over the list and call the *printBookDetails* method on every book.

Also create a generic List of Movies, add some movies to this list, and iterate over the list and call the *printMovieDetails* method on every movie.

What did you have to change in the loops when now using generic Lists instead of arrays?

Any difference between the for loop and the for-each loop?

---

## Task 2

Continue working on the Amazon project. We need to add a productId to the Movie class to be able to identify every unique movie in the store.

Add a new private variable to the Movie class, with the name *productId* and with the type of long.

Add a public setter and getter to the productId in the Movie class, and also add the productId to all of the constructors in the class. This will make it necessary to add a productId when creating the movies. Add a unique id of your choice to every movie that you create.

Change the *printMovieDetails* method to also print out the productId variable.

---

## Task 3

Continue working on the Amazon project. We also need to be able to search the movies in the store and find the movie with a specific productId.

Create a new method with the name *findMovieById* that returns a Movie, and that takes two input arguments:

- a productId of type long (the id to search for)

- a List of Movies (the list of movies to search in)

In the method, look through the list of movies and return the movie with the corresponding id.

If no movie was found with the specified productId, return null.

In the main method, use the new *findMovieById* method to search the moves with a specified productId.

If a movie was found, print "Found this movie: " and then the output from the *printMovieDetails* method from the movie.

If no movie was found, print "Didn't find a movie with productId: " and then the productId that was used in the search.

Try to run the program with productIds that exists and productIds that don't exist!

---

## Task 4

Continue working on the Amazon project. Are there any downside to searching a List like in task 3?

What about if you have millions of movies in the store. The search would sometimes take a very long time to execute. Sometimes millions of movies would have to be examined in the search before a movie was found. The time of the search would be very different for different searches.

A list isn't the best type of collection to use when searching by id.

Try instead to use a Map, and put every movie into the map with the productId as the key.

Create a Map called movieMap with a key of Long and a value of Movie.

Add all movies to the map with the productId as key and the movie as value.

Then try to find movies by productId in the map instead of the list. Are there any advantages of using a map instead? What about the time it takes to execute every search?

---

## Task 5

Create a program that returns the number of characters and the number of unique characters in a string entered by the user. Use a Set<Character> to find out about every unique character.

An example of executing the program can look like this (text colored in green is entered by the user):

```
Please write something: Hello, world!
The number of characters: 13
The number of unique characters: 10
```