# Week 04: Coursework

## Java Advanced

1. Fork and clone the following Github repository.

<https://github.com/busyQA-java-developer/busyqa-04-java-3-coursework-initial.git>

1. Import the Gradle project into Eclipse.
2. There are several exercises in the repository. Please read the following instructions to complete the Java programs:
3. *Exercise AbstractClass:* This program prompts the user for a book’s title, reads it from the keyboard, and stores the value in a string-typed variable called title. Write the MyBook class that extends the Book class and implement the setTitle(String s) abstract method. Uncomment the block of code indicated in the source file and run the program.
4. *Exercise ExceptionGeneration*: Write a method that computes the power of a number. The method’s signature is:

*long power(int n, int p)*

The method takes two integers, n and p, to calculate the value of n to the power of p. If either n or p is negative, the method must throw an exception that says "n or p should not be negative". If both n and p are zero, the method must throw an exception that says "n and p should not be zero". For example, -4 and -5 would result in java.lang.Exception: n or p should not be negative. Write the power(int, int) method in the class MyCalculator and return the appropriate result. Uncomment the line of code indicated in the source file and run the program.

1. *Exercise Inheritance*: Write a class called Arithmetic that has a method with the following signature:

int add(int a, int b)

The method takes two integers and returns the sum. Then write a class called Adder that extends the Arithmetic class. Finally, uncomment the lines of code indicated in the source file and run the program.

1. *Exercise InterfaceImplementation:* Write a class called Calculator that implements the AdvancedArithmetic interface and its abstract method:

int divisorSum(int n)

The method takes an integer number and returns the sum of all its divisors. For example, the divisors of 6 are 1, 2, 3 and 6, so the divisorSum() method should return 12. The maximum input value is 1000. After completing the Calculator class, uncomment the code indicated in the main() method and run the program.

1. Exercise Overriding: Override the getNumberOfTeamMembers() method in the Soccer class to print out the “Each team has 11 players in the Soccer Class” message instead of the “Each team has n players in the Generic Sports” message of the parent’s class. Run the program.
2. After you finish the five exercises, please commit and push your changes to your GitHub repository.
3. You can find the solutions for the exercises above in the Github repository:

<https://github.com/busyQA-java-developer/busyqa-04-java-4-coursework-complete.git>

# Additional Exercises - Optional

## Collections, Sorting, and Search Algorithms.

1. Please create a new Gradle project in Eclipse.
2. Inside the project, create a java package called “*com.busyqa.exercises”.*
3. *Collections* - Open the following link and create a class in the “*com.busyqa.exercises*” package for each exercise from the ArrayList exercises 1 to 15, from the HashSet exercises 1 to 12, and from the HashMap exercises 1 to 12. You can find the solutions to the exercises in the link.

<https://www.w3resource.com/java-exercises/collection/index.php>

1. *Sorting* - Open the following link and create a class in the “*com.busyqa.exercises*” package for exercises 1 and 2. You can find the solutions to the exercises in the link.

<https://www.w3resource.com/java-exercises/sorting/index.php>

1. *Search* - Open the following link and create a class in the “*com.busyqa.exercises*” package for exercises 1 and 2. You can find the solutions to the exercises in the link.

<https://www.w3resource.com/java-exercises/search/index.php>

1. After you finish the exercises, please upload your project to your GitHub repository.

// End