FIT9131 Programming Foundations

Week 5 Exercises

A. Homework checklist

To be up to date you should have completed the following:

- Lab exercises from weeks 1– 4.
- Read Chapters 1–4 of the textbook.

Assignment 1 checklist:

You should have started work on the coding for your assignment. By the end of this week, you should have at least two classes (**`Player`** & **`LuckyDipGenerator`**) fully coded.

If you have not started on the coding, you must consult your tutor immediately (or attend one of the weekly Helpdesk Consultation sessions). ***Remember that no one will be able to help you with the coding if you leave it until the very last moment***.

Loops (iteration or repetition) were covered in this week's lecture. This should help you with coding the **`Game`** class.

You should be showing you tutor your code for the assignment that you have completed so far to get some feedback on what you have done.

Reminder: Assignment 1 is due in Week 7. There is no extension to the due date.

There will be 2 additional resources for you this week (both found in the Week 5 section):

1) **A 10-minute video presentation on Academic Integrity**
2) **A 5-minute Quiz to test your understanding of Academic Integrity**

These resources provide important advice for you on the university policy on Academic Integrity. Please make sure you spend some time doing them, as they are crucial for you when submitting any assessable items (such as your 2 Assignments).

B. Exercises for Week 5

As we have done in other weeks, create a new project in BlueJ for your work this week. Name the project **Week5** and create all your classes for the week in this project.

Have the current version of your Assignment 1 ready to show your tutor during the class Your tutor will make a note of your progress and give you some feedback on what you have done (or not done), either during the lab or the helpdesk session.

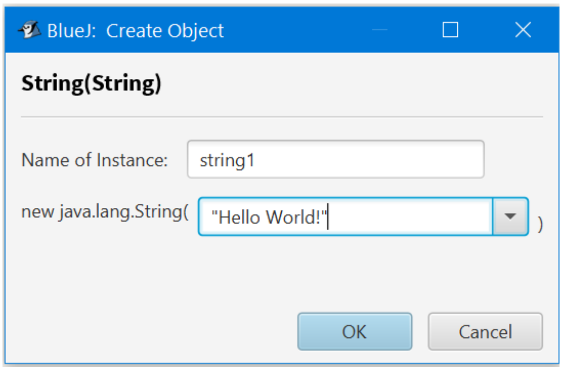
1. Using the Java class library and introduction to the String class

This exercise will give you practice in accessing the Java class library. They will also help you to become familiar with the `String` class and some of the methods that can be used on `String` objects.

To look up the Java documentation for the `String` class, visit the website, https://docs.oracle.com/en/java/javase/14/docs/api/. This website contains the complete Java documentation and will be very useful for you, so it is important that you become familiar with it. Enter `java.lang.String` into the search box at the top of the screen. This will show a description of the `String` class. Look under the ***Method Summary*** section to see a list of each `String` class method.

Now we will create a `String` object. While in your **Week5** project in BlueJ, select `Use Library Class` from the `Tools` menu. Into the combo box labeled **Class** enter `String`. This will show you a list of the available **constructors** for a `String` class object. Select the constructor `String(String)` and click on the "OK" button.

This action will result in the dialog below being shown, asking for the value of the `String` parameter of the constructor to be entered. Enter a string and click "OK".



Name of Instance: string1

new java.lang.String("Hello World!")

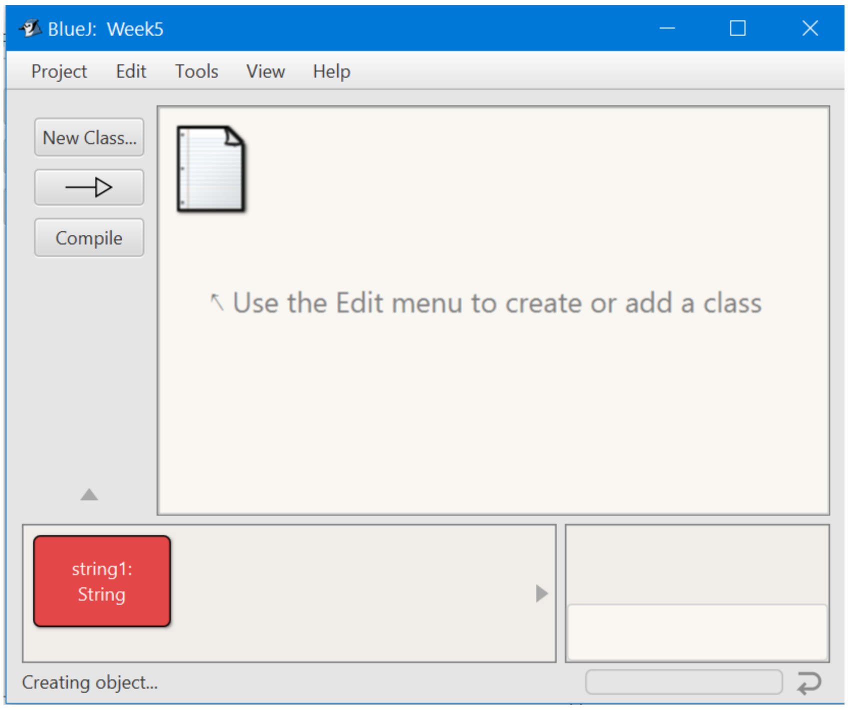
This will result in a `String` object being created on the BlueJ object bench with the value "Hello World!".

Another way to create a `String` object on the object bench is to type a command into the *Code Pad* for example:

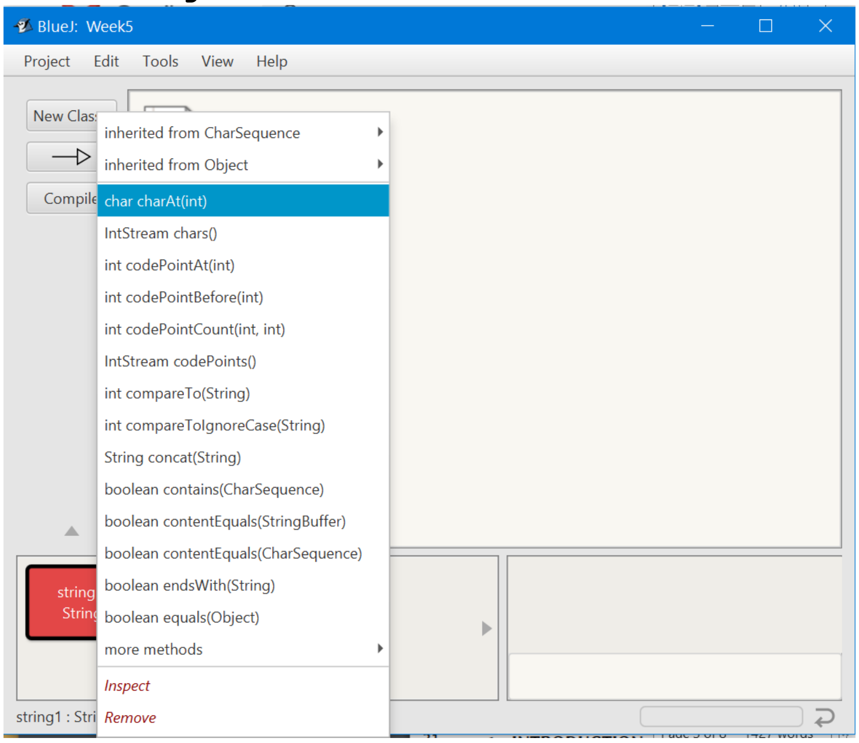
```
new String("Hello World, again!!")
```

Then, click on the red object icon that appears to add the object to the workbench.

Using either of the above methods you should see a String object on the workbench as follows.



If you right-click on the **`String`** object on the you will see all the available public methods of the **`String`** class as shown below.



Any of these methods can be invoked by simply pointing to the method name and clicking the left mouse button.

2. Exploring the String class

Explore the String class further by creating objects on the object bench, of various lengths and different combinations of characters, and invoking the methods available.

Describe in your own words in the space provided what the following methods do.

`length()`	
`substring(int, int)`	
`substring(int)`	
`indexOf(int)`	
`lastIndexOf(int)`	
`charAt(int)`	
`equals(Object)`	
`equalsIgnoreCase(String)`	
`toUpperCase( )`	
`toLowerCase( )`	
`trim( )`	

For more practice, try calling the String object's methods from the BlueJ Code Pad as well.

3. Using the String class

The following example will give you practice in using some of the methods in the **`String`** class. Create a class, name it **`Week5Q3`**. Then add the following three methods (one at a time):

a) a method called **`checkStringLength`** that has a single parameter of type **`String`**. The method will return a **`boolean`** value of ***true*** if the parameter is exactly four characters in length, otherwise it will return ***false***.

b) a method called **`isCharacterNumeric`** that has a single parameter of type **`char`**. The method will return a **`boolean`** value of ***true*** if the parameter is a numeric character, otherwise it will return ***false***.

   Hint: look up the ***Java*** **`Character`** class, and check out its **`isDigit`** method.

c) ***Optional*** (do this if you have time at the end of the class – otherwise do it as homework): a method called **`isStringNumeric`** that has a single parameter of type **`String`**. The method will return a **`boolean`** value of ***true*** if the parameter is a numeric string, otherwise it will return ***false***.

   Hint 1: you will need a loop in this method.
   Hint 2: can you think of how you can use the **`isCharacterNumeric`** method you wrote in Q.3(b) above?

Test the methods to see if they work as expected.

Preparation for homework: How can you use the three methods above to find out if a given string is exactly 4 characters long ***AND*** is entirely numeric?

4. Random number generation

Examine the following code. Read the comments above the method headers to understand what the methods do. Copy the code into your **Week5** project. Compile the class, create an object from it, and call the methods. Observe the results.

Hint: Part of this code may be of help to you in Assignment 1. Which class could you use this in?

```
public class Week5Q4    // constructors omitted for brevity
{
    // print a random number
    public void printRandomNumber(int maximumNumber)
    {
        System.out.println(generateRandomNumber(maximumNumber));
    }

    // generate and return a random number between 1 and the
    // maximum number, inclusive, entered as a parameter
    public int generateRandomNumber(int maximumNumber)
    {
        return 1 + (int)(Math.random() * maximumNumber);
    }
  }
```

Note: this is only of several ways to generate random numbers in Java. Do some research if you want to find out about other alternatives.

5. Collections

Open the **music-organizer-v1** project in BlueJ. This project may be found in the chapter04 projects folder.

a) Create a **`MusicOrganizer`** object.

b) Add the name of an audio file to the collection. This requires just a simple string for the name of an audio file but there are some sample names in the audio folder in the **`chapter04`** projects that you could use. (Hint: use the **`addFile`**(**`String`**) method).

c) Add the name of another audio file to the collection.

d) Predict what happens when each of the following methods are invoked in the order shown. Write your answer in the space provided.

`listFile(1)`	
`removeFile(0)`	
`listFile(1)`	
`getNumberOfFiles()`	
`listFile(0)`	

e) Check your predictions by calling the methods in BlueJ.

f) Add a method called **`checkIndex`** to the **`MusicOrganiser`** class that will take an integer parameter and checks whether it is a valid index for the collection. To be valid an index must in the range 0 to (size of the collection – 1), inclusive.

C. Homework

1. Finish the lab exercises for week 5.

2. Read sections 6.3-6.5, 6.11, 6.12 from Chapter 6 and sections 7.1-7.4 from Chapter 7.

3. Write the following declarations:

   - a field named **library** that can hold an **ArrayList** of objects of type **Book**.
   - a local variable named **fit9131** that can hold an **ArrayList** of objects of type **Student**.

4. Continue working on the assignment. In your Week 5 lab you will be asked to show your assignment progress to your tutor again.

D. Pre-lab tasks to be assessed in Week 6

1. Write a method called **`displayStateName`** that is passed a postcode of type **`String`**. A valid postcode is defined as one which contains exactly 4 numeric characters. If the postcode is valid the **`displayStateName`** method displays the name of the state, based by the first character of the postcode, as follows:

```
2   :     NSW
3   :     Victoria
4   :     Queensland
5   :     South Australia
6   :     Western Australia
7   :     Tasmania
0, 1, 8 or 9 : Postcode not known
```

If the postcode is invalid, then the string “Invalid postcode” is displayed.

Hint: You can use the methods that you wrote in in Q.3 of this week’s exercises.

2. Add a method to the **`MusicOrganiser`** class called **`displayCollection()`** that will display on the screen the names of the audio files in the collection. Display each name on a separate line. At the end of the list display the total number of files in the collection. Write two versions of this method:
   i) Using a **for-each** loop
   ii) Using a **while** loop.