



MACQUARIE UNIVERSITY

Faculty of Science and Engineering

COMP125 Fundamentals of Computer Science Workshop Week 2

Learning outcomes

Following are this week's learning outcomes,

- a. Perform problem-solving tasks
- b. Create a Java project from scratch
- c. Identify and eliminate bugs from an incorrect implementation

Download Workshop week 2 files from iLearn and import the project contained inside (workshop02template) in Eclipse. The process of importing Java projects from archive files is explained in week 1 tutorial worksheet.

1. Time-Distance relationship

Speed is defined as distance travelled divided by time taken. Design a solution to find out time taken to travel distance d_2 if time taken to travel distance d_1 is t_1 .

Solution: distance1 distance takes time1 time
unit distance takes time1/distance1 time
distance2 distance takes distance2 * time1 / distance1 time

2. Swap two variables

Design an algorithm that swaps the contents of two variables. If the first variable holds the value 5 and the second 8, then after the algorithm is executed, the first variable should hold the value 8 and the second 5.

Solution:

```
1 variable 1 --> temp
2 variable 2 --> variable 1
3 temp --> variable 2
```

3. String theory

For this question, you will need the following two methods that operate on a String object (assuming the object name is `str`):

- `str.length()`: returns the number of characters in the String.
- `str.charAt(int)`: returns the character at passed index, provided the index is valid (between 0 and `str.length() - 1`, including `str.length() - 1`).

Example:

```
1 String s = "hello";
2 System.out.println(s.length()); //displays 5
3 System.out.println(s.charAt(0)); //displays 'h'
4 System.out.println(s.length(4)); //displays 'o'
5 //System.out.println(s.length(-1)); will cause a StringIndexOutOfBoundsException
6 //System.out.println(s.length(5)); will cause a StringIndexOutOfBoundsException
```

In project `workshop02template`, there is a class file `StringTheory`. Please open this file. Notice that there are three methods, a main method and a helping method, `countOccurrences`.

- Complete the method `countOccurrences` which, when passed a String `aString` and a character `ch`, returns the number of occurrences of `ch` in `aString`.
- The main method has a piece of code that inputs a String and a char from the user. Write a few lines of code that displays the number of occurrences of the char input in the String input by calling the method `countOccurrences`.

Solution:

```
1 public static int countOccurrences(String aString, char ch) {
2     int count = 0;
3
4     for(int i=0; i<aString.length(); i++) //for each character
5         if(aString.charAt(i) == ch) //found a match
6             count++;
7
8     return count;
9 }
10
11 public static void main(String[] args) {
12     // ...
13     // already supplied code here
14
15     int count = countOccurrences(s, ch);
16     System.out.println(ch+" occurs "+count+" times in "+s);
17 }
```

4. Creating Java project

Follow the following instructions to create a new Java project.

- Click on File → New → Java Project.
- Give the project a name. By convention, Java project names are camel-cased, starting with a lowercase letter. For example, `myVeryOwnJavaProject`. For this example, name the project `task4project`.
- Press ENTER, or click on Finish.
- Double-click on the project. Then right-click on `src` and choose New → Package.
- Name the package `comp125`, which is the default package name for all projects in this unit. Press ENTER, or click on Finish.

f. Right-click on `comp125` and choose `New -> Class`. By convention, Java class names are camel-cased, starting with an uppercase letter. For example, `MyClass`. For this example, name the class `Task4`.

g. **Check the button that states `public static void main(String[] args)`**

h. Now you are ready to add code inside the `main` method, and add more methods.

i. Methods that are called by the `main` method, must be prefixed with keyword `static`. For example, if you have a method that returns the square of a `double` passed to it, and is called by `main`, it will be defined as,

```
1 public static double square(double num) {  
2     return num*num;  
3 }
```

Also, methods that are called by other `static` methods, must be prefixed with `static`. This is not true for all methods and will be made clearer in the next few weeks.

j. In the `main` method, write a piece of code that computes the sum of the first hundred odd integers, and displays it in the console. Console output is given using `System.out.println(stuff to output goes here)`

5. In class `Buggy`, the code in `main` attempts to compute the factorial of 5. Factorial of an integer n is defined as the product of the first n positive integers ($1 \times 2 \times \dots \times n$). However, the code contains two bugs. Identify and correct them. The value displayed when the bugs are eliminated should be 120.

Solution: The variable `factorial` should be initialized to 1 (instead of 0), and the loop expression should be $i \leq n$ (instead of $i < n$).

6. (Assessed Task) Complete the following methods in class `AssessedTask`:

- (50%): `isPerfectSquare(int n)` that returns `true` if the square root of the passed integer is an integer as well, and `false` otherwise. Hint 1: `Math.sqrt(n)` returns the square root of n where n can be an integer or a floating-point value. Hint 2: `(int)val` casts a double `val` to integer. For example, `(int)4.52` is 4.
- (35%): `timesDivisible(int n, int p)` that returns number of times n is divisible by p . For example. 250 is divisible by 5 three times ($250/5 = 50$, $50/5 = 10$, $10/5 = 2$).
- (Advanced, 15%): `arrayToString(char[] ch)` that returns a `String` containing all characters from the passed array, in the order they occur in the array. For example if the array passed is `{ 'h', 'i', '!' }`, the value returned is the `String "hi!"`.

A sample set of method calls is supplied in `main` alongwith expected outcome. Please note we may test your program with a different set of data.

You should submit the file `AssessedTask.java` on iLearn under “week 2 submission” by Sunday 9th August 11:45pm

Additional tasks (for anyone who has a little spare time)

- Write a method that when passed a `String`, returns the most frequently occurring `char` in that `String`. For example, when passed “abysmal”, the method returns ‘a’. In case of a tie, return the `char` that occurs first. For example, when passed ‘surreal’, the method returns ‘r’.
- Write a method that when passed a `String`, returns a `String` containing the most frequently occurring characters in that `String`, in the order of their occurrence. For example, when passed “abysmal”, the method returns “a”, and when passed “fantastic”, the method returns “at”.