Introduction To Programming
Tutorial 6
(See Canvas→Assignments for due dates and marks)

Note: Do not include your name, student ID or any personally identifiable info in your submission as the submission may be used for peer reviews; your submission will not be lost as Canvas keeps track of these internally.

Please follow all of the steps below in the given sequence:

1. Read all unread announcements and unread replies to announcements under Canvas→Announcements.

2.1 Do any missed tutorials before going further.

2.2 Watch any unwatched recordings of the compulsory **Weekly Live Lecture** and any important videos in the Extra Videos Playlist.

2.3 If you need help in addition to what has been shown in the compulsory weekly live lecture, you are also expected to speak to your **group tutor via discussion forums** and attend/watch their live sessions. Please note that group tutors cannot debug your assessment code on your behalf as debugging is a part of every programming assessment.

2.4 **If you still have have any unresolved questions** or **if you need further feedback**, post the relevant parts of your submitted work in a new post under Canvas→Discussions→Tutorial discussions and ask from your group tutor. E.g. you can ask "*In the live lesson Gayan did ___ with ___. I didn't do ___ so should I be doing this as well?*", etc. Please note that the university requires teaching to be conducted in an equitable manner so your tutors will require you to post questions in the discussion forums.

3. Check any available feedback of your previous submissions and if you have any unresolved questions or if you need further feedback, post the relevant parts of your submitted work in a new post under Canvas→Discussions→Tutorial discussions and ask from your tutor. E.g. you can ask "*Gayan showed _____ but I did mine like _____, so which is the better approach and why?*", etc. Please note that the university requires teaching to be conducted in an equitable manner so please only use email for matters such as special consideration.

4. Follow the materials under Canvas→Modules→Week 6…

5.1 (Hypothetical scenario) You turn up to your programming job one fine afternoon and find out that a programmer in your coding team has quit without giving any notice and all that the company has of the code that they were working on is the following "screenshot" (from which you would not normally be able to copy and paste):

```
import javax.swing.JOptionPane;

public class RenovationProjectManager {
   public static void main(String[] args) {
      double wallArea = 0, cost, height, length, costPerSqm;
      int selection;
      String tempInput, wallNames;
      String menu = "Menu:\n";
      menu += "1. Calculate paint required for a wall\n";
      menu += "2. Calculate paint required for project";
      tempInput = JOptionPane.showInputDialog(menu);
      while (tempInput != null) {
         selection = Integer.parseInt(tempInput);
         if (selection == 1) {
            costPerSqm = Double.parseDouble(JOptionPane.showInputDialog("Enter cost per sq.m ($)"));
            tempInput = JOptionPane.showInputDialog("Enter a wall name");
            height = Double.parseDouble(JOptionPane.showInputDialog("Enter " + tempInput + " wall height (m)"));
            length = Double.parseDouble(JOptionPane.showInputDialog("Enter " + tempInput + " wall length (m)"));
            wallArea = height * length;
            cost = wallArea * costPerSqm;
            JOptionPane.showMessageDialog(null,
                  "Cost to paint " + tempInput + " wall of " + wallArea + " sq.m. is $" + cost);
         } else if (selection == 2) {
            costPerSqm = Double.parseDouble(JOptionPane.showInputDialog("Enter cost per sq.m ($)"));
            wallNames = "";
            wallArea = 0;
            cost = 0;
            tempInput = JOptionPane.showInputDialog("Enter a wall name (cancel to finish)");
            while (tempInput != null) {
               height = Double.parseDouble(JOptionPane.showInputDialog("Enter " + tempInput + " wall height (m)"));
               length = Double.parseDouble(JOptionPane.showInputDialog("Enter " + tempInput + " wall length (m)"));
               wallArea += height * length;
               wallNames += tempInput + ", ";
               tempInput = JOptionPane.showInputDialog("Enter a wall name (cancel to finish)");
            }
            cost = wallArea * costPerSqm;
            JOptionPane.showMessageDialog(null,
                  "Cost to paint " + wallNames + "wall(s) of " + wallArea + " sq.m. is $" + cost);
         } else {
            JOptionPane.showMessageDialog(null, "Invalid choice!");
         }
         tempInput = JOptionPane.showInputDialog(menu);
      }
   }
}
```

Your first task is to run the code in Eclipse and add some high-level, plain-English comments, explaining what the different parts of the code are doing. **Speak to your group tutors via forums/tutor chats for advice.**

5.2. **With the help of your group tutors via forums/tutor chats:** Create a method for each of the (two) menu options and then move the code relevant to each menu option from the main method to the relevant methods that you created. In place of the original code that was moved, you must have a call to the method that would now do the work instead.

For this week, you must ensure that:
1. Only method definitions are at the class level (e.g. do not create class/object member variables).
2. Only the 'main' method definition has any mention of 'static'. All method methods should be explicitly 'public' and not static.

5.3. **With the help of your group tutors via forums/tutor chats:** Create a constructor method and then move all of the code in the 'main' method to the constructor method instead. Add comments to the code and explain what further changes you needed to make to the code in 5.2.

5.4. **With the help of your group tutors via forums/tutor chats:** Add a comment to your main method explaining what we should have in the main method and what we should not.

5.5. Add comments in the style required by Assignment 2. See rubric in section 9 of the Assignment 2 PDF.

Submission Checklist (Compulsory):
1. Format your code (e.g. Eclipse→Source→Format).
2. It is OK if your independent investigative efforts before the lesson are not fully functional but a program with even one red dot under Eclipse (programs with syntax errors) are not runnable/testable Java and cannot be marked. Remove any parts of the code that result in syntax errors and explain the issues in code comments instead. Do not keep code that is commented out.
3. Go to Canvas→Assignments→**Independent Investigative Effort 6** and select 'submit assignment'.
4. Select to attach files from your computer, navigate to your Eclipse workspace folder→Project folder→src folder and select the (one) final version of your `RenovationProjectManager.java` file. Please do not submit more than 1 file at a time as it delays the marking process.
5. Verify your submitted file as shown during the week 1 chat session.

6. If you have not submitted your final version of A2, add comments explaining your plan. Note that this will not be marked but it is to help you progress.

Submission Checklist:
1. Ensure that your code does not have any red dots (Java errors) as code with such errors cannot be tested/marked and will receive 0 for that submission. If your code has red-dots, refer back to similar code and fix the error or remove the code that is causing the problem. You must not leave any commenteded out code in your submissions. Yellow dots are warnings and these are different.
2. Ensure that you have added comments to your .java file explaining what you have done and any potential alternative approaches.
3. Format your code (e.g. Eclipse→Source→Format).
4. Go to Canvas→Assignments→**Independent Investigative Effort 6** and select 'submit assignment'.
5. Select to attach files from your computer, navigate to your Eclipse workspace folder→Project folder→src folder and select the (one) final version of your `.java` file. Please **do not submit more than 1 file** as it delays the marking process. You can also context select on the .java file name from package/project explorer inside Eclipse and find its exact location. Only the last submission is the official submission.
5. Verify your submitted files as shown during the week 1 session.

Having trouble with usernames, passwords, access, etc.? Please call the RMIT IT Service and Support Centre for quick help on 03-9925 8888 and remember to ask for a reference number and pass it on to your instructor.

Need extensions or special consideration? Please follow details and process below:
https://www.rmit.edu.au/students/student-essentials/assessment-and-exams/assessment/special-consideration