Assignments Procedure

Marking Scheme

- Marks are shown on the assignment

General Tips

- When starting out is is better **not** to write lots of code, compile it to discover lots of compile errors and then when it eventually *does* compile - it won't run.
- Better to start with something simple that works and compile and run after every small change. "*Code a little, test a little*".
- Make regular backups of your work to your own media! Don't trust the our servers!
- Keep older versions of your code so you have something to fall back on if something breaks after you make a change (usually the day before handup!).
- Email copies of your assignment to yourself every so often to archive them.
- Use Blackboard to submit assignments - NO printouts are required.

Coding Standard

Use a coding standard, here's a basic one:

1. At the top of your program state your name, email address, revision, revision history with dates, and what the program is about.
2. Be well commented including what each method/class you write does. For methods note what inputs and outputs it uses, error codes returned (where applicable).
3. Use conventions such as block capitals for names of constants or 'k' in front of constant names (e.g. RADIUS or kRadius). Use capitalization to break up words e.g. 'FirstName'. Class names should start with a capital letter, variables and methods with a lowercase letter and camel case.
4. Use meaningful names for variables/classes/methods. You should use long names but limit to <15. Avoid names that are similar or differ only in case. eg. firstName and FirstName. For classes start with a capital letter and break works with capitals e.g. SerialPort, WeightCalculator. For method and variables start with a lowercase letter e.g. customerAddress, calculateAge ( ). For GUI components e.g. Swing put the class name at the end e.g. inputjLabel, ageJTextField.
5. Try to make variables (& possible some methods) 'private' where possible.
6. For later assignments write your own methods and classes. They should be generic where possible so that they can be re-used in other programs in the future.
7. Be maintainable – easy to add new features
8. Avoid using 'hard wired' numbers in for loops, drawings etc. Use constants declared at the top of the program or pass the values into methods

9. If/else, while, for loops - try to limit nesting to max 3 levels (so it's easy to follow)
10. For later assignments consider using customised components and or generic components.
11. A more comprehensive coding standard is uploaded earlier on the slack channel so you can have a read through.

Development Tools

- You can use any development tool you like but android studio is the preferred option.

Documentation

.

Assignment Demo -

You are required to demonstrate your assignment to the lecturer. If you do not demonstrate it then it may not pass. You may be asked to make changes to your code as part of the demo.

.

For Applications/Server side/Android: you need to demonstrate the program running. Be able to show the features you've implemented and be able to answer questions on how you implemented the features in code.

Format of the Demo:
Show your program running and briefly show the important parts working. If you have spent time on an unfinished feature you should mention this at the demo and in the documentation as you may get marks for it.
You may be asked to make a small change to your program during the demo to demonstrate your understanding on what you have done.