Notes from the 2020 Prelim practical exam – The Bare Bones Approach

General

1. You MUST memorise how to read in a text file using the Scanner class. Write it down from memory before you start coding your exam (first 5 minutes)
2. In the second 5 minutes of the exam name your classes strictly according to the exam paper. Do not deviate. Start again if you must.
3. Read and code the **class diagrams**. Follow them EXACTLY. Note: The constructor must have the same name as the class. Do NOT substitute your own names for classes, methods etc
4. Do not put the main method into the code needed for "question one". The main method goes into the UI class which can be question two or three or four or five. Read the exam.
5. Do not put methods inside your main method.
6. NetBeans will help you with "try catch" and the libraries you need to import.

Constructors match the class name, which matches the exam paper.

7. Constructors have the **same name** as their class.
8. You can have more than one constructor. They differ according to the **number** of parameters they accept.
9. The constructor **initialises** newly created objects i.e. gives the fields its initial values.
10. The constructor **automatically** runs code e.g. reading in the text file or loading a menu.
11. The constructor method heading does not have a return type e.g. public ~~void~~ Food ( ); public ~~String~~ Food (String n, int n);

The Interface – UI – Use the name given in the exam paper.

12. **Create the UI class first**, regardless of the order in the exam paper.
13. The UI class has the main method.
14. The UI class instantiates an object from the manager class. It creates the manager object. e.g. FoodManager fm = new FoodManager( ); // no parameters in this example
15. The UI handles input, output and defensive coding (if asked for). The manager class has all the methods for processing.
16. The UI class calls the methods in the manager class.
17. The name of the class matches its own file name EXACTLY.
18. Any other methods you need in the UI class besides the main method must be declared below and after the main method. These extra class methods must be "**static**".

Variables use the name in the class diagrams

19. Variables are either "local" to their method or "global" (declared outside of any method at the top of the class). Global variables are still generally private.
20. Global variables can be declared as final e.g. final double VAT_RATE = 0.15; // 15%

The Main Method in the UI class

21. We do not declare variables in the main method as private or public. They are local variables to main (the scope of the variable)
22. We do not declare variables inside **any** method as private or public. They are local to the method.
23. Input and output to the screen is done here (only).

The Manager Class – Use the name given in the exam paper

25. **Create the manager class second** regardless of the order of the exam paper.
26. This has most of the methods that are called by the UI class.
27. The manager class may have "**helper methods**". These are private as they help the other methods in the manager class achieve their goals.
28. There will always be a "print all" method i.e. using a for loop to create a long output String that can be returned to the UI for printing to the screen. It will always use the toString method from the relevant object.
    a. Manager class methods do not generally output to the screen. Instead they return a value for printing out by the UI class.

The Text File(s) – Do not change the name of the text file.

29. In NetBeans remember to save the text file in the project folder (not the scr folder)

The Object class – the class(es) from which you create (instantiate) objects. You will probably create an array of objects from your object class. Use the names in the exam paper.

30. Even if "question one" starts with the creation of an object class, do not do this first. Create the UI class first (may be a later question). Create your bare bones framework first i.e. UI, managerClass and then object class
31. The object class will have global private variables, the constructor method(s), the getter methods, the setter methods and the toString method.
32. The object class will have **global private variables** which you create OUTSIDE (at the top) of any methods the class may have. One of these will probably be an array (to create your array of objects)

The Golden Thread - regarding parameters

33. The while loop that reads in the text file, must match the fields in the text file. The parameters in the call, must match the parameters in the constructor (or method heading), which must match the variables in the object class. (see java-teacher for "Golden Thread")

Defensive coding

34. Only code defensive coding if asked for in the exam.

Currency – always declared as a double

35. Know how to round off to two decimal places using DecimalFormat or . . .
    a. String.format("R%.2f", myDoubleAmount)

Arrays – Size is declared versus size is not declared.

36. **Size of array is declared, and size counter is used in the while loop. E.g. Size = 11**
    a. int[ ] myIntArr = new int[20]; // This array has 20 places. Index 0 to index 19.
    b. for (int i = 0; i < **size**; i++) // This loop will only iterate only 11 times
    c. for (int i = 0; i < **myIntArr.length**; i++) // This loop will iterate 20 times because the length of the array is 20. May crash as some places may be null (not used)
37. **Size of the array is not declared**
    a. String[] myCodeArr = {"SSR", "PP", "OB", "FA", "KFC"};
    b. for (int i = 0; i < myCodeArr.length; i++) // Will not crash – no null values in array.

SQL – If you don't know points b, c and d you loose 25% of the marks in the SQL section.

a) Study practice, practice study. You do not know SQL the way you should.
b) When the fields are coming from more than one table . . . You MUST **explain the join** to SQL. You must show the primary key to foreign key relationship in your SQL code. This is done after the WHERE clause.
c) Study the **UPDATE, INSERT AND DELETE** structures. They are different to SELECT. They use "UPDATE SET WHERE", "DELETE FROM WHERE" and "INSERT INTO SELECT FROM WHERE"
d) Generating a **random number** in SQL e.g. generate a random number between 10 and 99
    a. INT (RND (custID) * 90 + 10). NOTE: The customer ID field is being used as the random number generation seed so that each random number is different for each customer. Because random numbers are doubles you need the INT function to get rid of the decimal fraction.

Advanced hints and methods

Once your bare bones program works you need to move onto the more difficult questions. Now it is a good idea to "backup" your bare bones program. Do this in NetBeans by creating a New – Other – Empty file. Cut and paste your bare bones into this **plain text file**. Save it in your Project folder (not your scr folder). You can also create a dedicated backup folder for this i.e. New – Other – Folder.

As you create the more advanced methods from the exam paper label these new methods with a comment containing the question number. This helps the examiner mark your work more accurately.