Assignment 3
Object oriented program

Read Section 1 to understand the requirements, Section 2 to understand the programming tasks that you need to carry out and Section 3 to know what you need to submit for assessment.

1. Description

In this assignment, you will develop an object oriented program in Java that obtains user data input from the standard input to create objects and to display a report to the standard output. The program also provides an option to store the report to a text file.

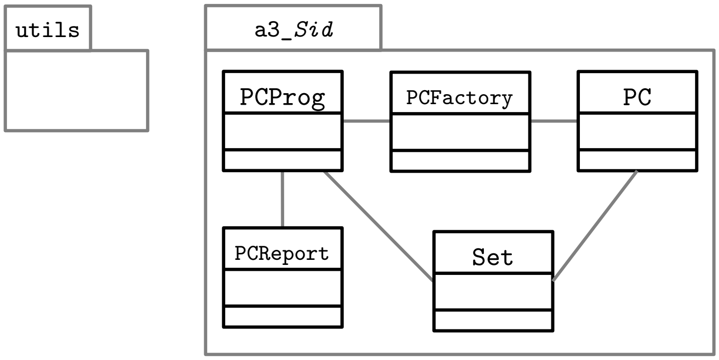

Figure 1: Program structure.

The program, named `PCProg`, processes data about personal computers. A personal computer (`PC`) is described in terms of the following attributes: model, year, manufacturer, and comps (short for components). The table below describes the domain constraints of these attributes. Note that attribute `comps` has the type `Set`. For the purpose of this program, components are simply strings which denote their names.

Attributes	type	mutable	optional	length	min	max
`model`	`String`	T	F	20	-	-
`year`	`Integer`	F	F	-	1940	-
`manufacturer`	`String`	F	F	20	-	-
`comps`	`Set<String>`	T	F	-	-	-

For example, a Dell Vostro PC (model 3650MT) with an Intel-Core-i3-6100 CPU, a 4GB-DDR3L RAM, a 500GB-Sata hard disk and an Intel-HD graphics card, is represented by the tuple:

`PC:<Vostro 3650MT,2016,Dell,Set:{Intel-Core-i3-6100 CPU,4GB-DDR3L RAM,500GB-Sata hard disk,Intel-HD graphics card}>.`

To **create a PC object**, the program first prompts the user for the required data values. Once the data values have been obtained, the program create a new PC object and add it to a set. This set is an object of the `Set` class. The program then asks the user if (s)he wishes to continue (the answer of which must be either "Y" or "N"). If the user answers "Y", the program repeats to create the next PC object. If, however, the user responds with "N", the program proceeds to displaying a report about the PC objects.

To **display a report**, the program first generates it and then presents the result on a table that looks like the one shown in Listing 1 below. The report title is displayed in the middle of the top banner. All but the first column correspond to the PC attributes, the rows are data about the PC objects. Thus, the second column corresponds to attribute model, the third corresponds to year, and so on. The last column lists the string representations of the components of the PC objects. The first column sequentially displays the row numbers. Note that the widths of second and fourth columns are lengths of the corresponding attributes. Widths of the first, third and fifth columns are 3, 6 and 50 (respectively). The cell values are properly aligned with the columns and are displayed right-justified. The cell values need not be wrapped. Further, boundaries of two adjacent cells on same row are exactly one space ('␣') apart.

Listing 1: A tabular report for PCs

```
---------------------------------------------------------------------------------------------------
                                           PCPROG REPORT
---------------------------------------------------------------------------------------------------
 1       Vostro 3650MT  2016                Dell [Intel-Core-i3-6100 CPU, 4GB-DDR3L RAM, 500GB-Sata hard disk, Intel-HD graphics card]
---------------------------------------------------------------------------------------------------
```

Immediately below the report is a prompt for whether or not the user wishes to **save the report** to a text file. If the user answers "Y" then the program saves the report text to a file named `pcs.txt`, which is located in the same directory as the program's. Otherwise, the program ends.

2. Task requirements

Below are the requirements of the tasks that you must complete for this assignment. Note that your program will be marked automatically by a program marker, which expects that different related parts of the same class function correctly. A design failure in one part (e.g. a setter operation for an attribute) may lead to failures in other parts (e.g. the getter operation for the same attribute).

1. Create a package named `a3_`*`Sid`* as shown in Figure 1, where *`Sid`* is your student id. For example, if your student id is 123456789 then the package name is `a3_123456789`.

   You will need to use this package to store all the Java class(es) that you create for the program.

   Copy to this package two classes (`PCProg` and `Set`) that are provided in the attached zip file. Fix the import statements in these classes to match your package name. The subsequent tasks will explain what you need to do with these classes. The design of class `Set` was explained in the coursebook.

   IMPORTANT:

   (a) Failure to name the package as described above will result in an invalid program.

   (b) You must use the necessary utility classes in the `utils` package (provided as part of the lecture resources). This package should be created as another top-level package as shown in Figure 1. You *must* *not* create package `utils` as a sub-package of your package. In addition, you *must* *not* include package `utils` as part of your submission.

2. Specify and implement class `PC`.

   Note:

   (a) `PC` must contain the essential state and behaviour spaces.

   (b) `PC` must appropriately use `Set` in its design.

   (c) you must implement `PC.toString()` such that the outputs look like the example shown in Section 1.

    (d) you must override the `equals` method for `Set`, which determines equality of two `Set` objects based on the elements that they have.

3. Specify and implement class `PCFactory` that has a factory method for creating `PC`s. This class must also be a singleton. You must strictly follow the relevant design pattern solutions.
4. Specify and implement class `PCReport` which contains a single operation `displayReport(PC[] objs): String`. Note the following:

    (a) the report format must be as specified in Listing 1.

    (b) this class has no instance variables.

5. A partially completed class named `PCProg` is provided for testing your program. Move it into your `a3_`*`Sid`* package so that you can run it. Class `PCProg` contains the method `main` and some operations for performing the tasks highlighted in Section 1:

    (a) attribute `objs` is typed `Set<PC>` and used to record PC objects.

    (b) `createObjects`: uses `PCFactory` to create a new PC object and record it in `objs`. Method `createObjects` must ***not*** invoke `PC` constructor directly.

    (c) `getObjects`: return the recorded PC objects.

    (d) `displayReport`: uses `PCReport` to generate and display the tabular report about PC objects.

    (e) `saveReport`: save the report text to file.

    (f) `main`: create a new `PCProg` object and run its operations.

    Note:

    - The following procedures are completed and provided for you: `displayReport`, `saveReport` and `main`. You must not change them.

    - In order to make the program work, you must specify and implement the operations `createObjects` and `getObjects`.

3. Submission requirements

Create a zip-compressed file containing **just the folder of the package** specified in Task 1. You must name the file as follows: **a3_*Sid*.zip**, where ***Sid*** is your student id.

Submit your file to the designated submission box.

IMPORTANT: Failure to name the file as described above will result in an invalid program. In particular, ONLY the **ZIP** format is accepted. Other formats (e.g. RAR) are NOT accepted.