**Assignment 3:**

**Due Date: Midnight, Sunday 12 June, 2022**

**Total marks: 30**

**Problem 1.**

**[reading from a text/binary file and writing to a text/binary file]**

Write a program that reads a file of numbers of type int and writes all the numbers to another file, but without any duplicate numbers. Assume that the numbers in the input file are already ordered from *smallest to largest* [*Note***:** this makes it easier to find duplicate numbers, as to find a duplicate number, you only need to check whether the number currently read from the file is equal to the number read in previous iteration or not]. After the program is run, the new file will contain all the numbers in the original file, but no number will appear more than once in the file. The numbers in the output file should also be sorted in reverse order from *largest to smallest*. [*Hint*: you can first read the numbers in an ArrayList, reverse the ArrayList, then write back to output file] Your program should obtain both file names from the user. Use either a text file or a binary file. For the text-file version, assume one number per line. For the binary-file version, use numbers of type int that are written using writeInt.

**Problem 2.**

**[Writing/reading serialized objects to/from binary files]**

Revise the class Pet, as shown in Listing 6.1 of Chapter 6 (Ref 2: Walter book), so that it is serializable. The Pet class has attributes: name, age, weight. Write a program that allows you to write and read objects of type Pet to/from a binary file named “pets.dat”. The program should be menu based showing the following options.

1. Select 1 to read all Pet records from the file, and display them. [read all records from the file into an arraylist of pets]
2. Select 2 to write all Pet records to the file. [write the arraylist (unsorted) of pets to the file]
3. Select 3 to write all Pet records sorted by age to the file. [This is the natural order of comparing pets. The Pet class must implement the comparable<Pet> interface. Use Collections.sort method to sort the arraylist.]
4. Select 4 to write all Pet records sorted by name to the file. [This can be implemented by creating comparator classes implementing the Comparator<Pet> interfaces. Use overloaded Collections.sort method to sort the arraylist by name.]]
5. Select 5 to exit.

A user, who has asked to write to a file [option 2, 3 and 4], can write 5 records of Pet added to arraylist at compile time. A user who has asked to read from a file, can read all records in an arraylist and show the records data as he likes, but all data related to each pet must be shown (e.g. name, age, weight).