Exercise 2 – Implementing a graph

Help
There are three solutions available with more detail added in each. Use them when you are on your own and need help to continue.

Exercise A:
Write a Java interface that specifies a graph ADT. Call it ***GraphIF***.
Add useful and necessary methods specify PRE and POST conditions. Assume the Vertex class defined below.

```
public class Vertex {
      private boolean mark;
      private String name;

      public Vertex(String name) {
            this.name = name;
            this.mark = false;
      }
      public String getName() {
            return name;
      }
      public boolean isMarked() {
            return mark;
      }
      public void setMarked(boolean mark) {
            this.mark = mark;
      }
}
```

Exercise B:
Implement your Java interface. Use a linked list representation, call it ***LinkedGraph***.
Test the implementation on *Crocodile Airlines*. You may want to use the provided ***Main.java*** class.
The trick is to

Exercise C:
Write a new implementation of the interfadce using a matrix representation (call it ***MatrixGraph***).
The data structure should probably be a two-dimensional array of ***Vertex*** objects.

Exercise D:
Are there common methods in the two implementations? If yes, refactor the common methods into a common **abstract class** (***AbstractGraph***). It should implement the interface. Methods that are different in the two implementations (that access the data internal representation) should be made **abstract**. Let the **LinkedGraph** and **MatrixGraph** inherit from **AbstractGraph**.