**Answer to Question1)**

Differences between abstract classes and interfaces

1. Abstract classes can contain member fields that are not initialized, while interfaces fields need to be initialized.
2. A Class can implement many interfaces while a Class can only extend one Abstract class.
3. An interface can extend many interfaces, while an Abstract class can only the extension of one class.
4. Fields in an interface need to be public, while in a class they can be either private or public.
5. An Abstract class can define a constructor, an Interface can't.

**Answer to Question 2)**

1. False. You can have interfaces with no methods. Check the example on folder a.
2. False. An instance field declared by an interface doesn't have to be declared by the implementing class. Check example on folder b.
3. False. Interfaces can't have constructor methods.

\*Refer to the files in the folder exer2 for example.

**Answer to Question 4)**

- A class like WindowAdapter with methods that do nothing can serve as an Abstract Class with a default implementation of all the methods from an interface. We can then extend this Class and only define in the details the methods that we really need or want.

**Answer to Question 5)**

You can prevent this by doing the following:

a - Declaring your constructor method private.

b- Declare a field of the type of the Class, declare it as private and static and instantiate it.

c - Create a public static method that returns the field defined in b. You can use this method to make reference to this unique instance of your class.

\*Refer to the files in the folder exer5 for example.

**Answer to Question 6)**

You might decide to opt for a lazy-initialization in order to improve performance, avoid wasteful computation, and reduce program memory requirements. Don’t create an instance of class until you really need it.

\*Refer to the files in the folder exer6 for example.

**Answer to Question 7)**

**Answer to Question 8)**

True.

\*Refer to the files in the folder exer8 for example.

**Answer to Question 9)**

1. Spring.
2. Dagger.
3. New.

**Answer to Question 10)**

1. There is no new.
2. It is returning a type interface or Abstract, as opposed to a Concrete class.
3. You have multiple classes that implement the same abstract class.

**Answer to Question 11)**

Writer out = new PrintWriter(System.out);

out = new SpecialWriter(out);

or

WrapFiler out = new Wrap (

New Buffer (

New Case (

New PrinWriter(System.out))));

out.setCentre(true);