Commencé le	jeudi 16 novembre 2017, 10:24
État	Terminé
Terminé le	jeudi 16 novembre 2017, 11:35
Temps mis	1 heure 10 min
En retard	1 heure 10 min
Points	18,10/19,00
Note	**19,05** sur 20,00 (**95**%)

Description

Some code exists for the simple social network project. It's not particularly great code, but its got ambition! Load it up from here: facebouc_v1.jar.

Oops. There's stuff missing. Well, you've got your first job...

Question 1

Correct

Note de 2,10 sur 3,00

As you can see, the method `MessagePost#display` doesn't really do very much. It should be displaying something like:

```
Leonardo da Vinci
Code this...!
0 seconds ago  -  2 people like this.
   No comments.
```

or

```
Taylor Twit
Party Like It's 1989
(c)1989 Taylor Twit
0 seconds ago
   12345 comment(s). Click here to view.
```

So, make it work.

The tests cover most of the possible cases that can arise, so you need to write your code appropriately. Yes there are (somewhat) hidden tests - you only see one failing test at a time.

You need to paste *just the method* into the Answer, not the whole class.

For example:

Test	Result

`MessagePost mp = new MessagePost("Leonardo da Vinci", "Code this...!");`
`mp.display();`
	`Leonardo da Vinci`
	`Code this...!`
	`0 seconds ago`
	`   No comments.`

Réponse:

```
1  public void display() {
2          System.out.println(username);
3          System.out.println(message);
4          System.out.print(timeString(timestamp));
5          System.out.println((likes > 0) ? ("  -  " + likes + " people like this.") : ' ');
6          System.out.println((comments.size() > 0) ? ("   " + comments.size() + " comment(s). C
7          : "   No comments.");
8
9      }
```

Vérifier

	Test
✓	`MessagePost mp = new MessagePost("Leonardo da Vinci", "Code this...!");` `mp.display();`

✓	`MessagePost mp = new MessagePost("Homer Simpson", "Facts are meaningless. You can use facts`
	`mp.like();`
	`mp.like();`
	`mp.unlike();`
	`mp.display();`

✓	`final MessagePost tsmp = new MessagePost("Taylor Twit", "Party Like It's 1989\\n(c)1989 Tayl`
	`IntStream.range(0, 12345).forEach(i -> tsmp.addComment("That sucks."));`
	`tsmp.display();`
	
	`// note that \\n is an artefact of the CodeRunner test`
	`// you will use the usual single backslash in your testing code instead`

Passed all tests! ✓

Correct

Points pour cet envoi : 3,00/3,00. En tenant compte des tentatives précédentes, cela donne **2,10/3,00**.

Question 2

Correct

Note de 2,00 sur 2,00

As you can see, the method `PhotoPost#display` doesn't really do very much. It should be displaying something like:

```
Alexander Graham Bell
  [handset.png]
  Coming soon: the Samsung Galaxy S13.
0 seconds ago  -  4 people like this.
   1 comment(s). Click here to view.
```

So, make it work.

The tests cover most of the possible cases that can arise, so you need to write your code appropriately. Yes there are (somewhat) hidden tests - you only see one failing test at a time.

You need to paste *just the method* into the Answer, not the whole class.

Réponse:

```
1  public void display() {
2          System.out.println(username);
3          System.out.println("  [" + filename + "]");
4          System.out.println("  " + caption);
5          System.out.print(timeString(timestamp));
6          System.out.println((likes > 0) ? ("  -  " + likes + " people like this.") : ' ');
7          System.out.println((comments.size() > 0) ? ("   " + comments.size() + " comment(s). C
8          : "   No comments.");
9      }
```

Vérifier

Passed all tests! ✓

Correct

Note pour cet envoi : 2,00/2,00.

Question 3

Correct

Note de 1,00 sur 1,00

Test your `MessagePost` and `PhotoPost` classes from the previous questions (where you added the display method to each) with the supplied `NewsFeed` class.

The output should look something like

```
Homer Simpson
Facts are meaningless. You can use facts to prove anything that's even remotely true!
0 seconds ago  -  1 people like this.
   1 comment(s). Click here to view.

Alexander Graham Bell
  [handset.png]
  Coming soon: the Samsung Galaxy S13.
0 seconds ago  -  4 people like this.
   1 comment(s). Click here to view.
```

Paste both your `MessagePost` and `PhotoPost` classes into the Answer.

Réponse:

```
1  package facebouc.v1;
2
3  import java.util.ArrayList;
4
5  /**
6   * This class stores information about a post in a social network. The main part
7   * of the post consists of a (possibly multi-line) text message. Other data,
8   * such as author and time, are also stored.
9   *
10  * @author Michael Kölling and David J. Barnes
11  * @author Peter Sander
12  */
13 public class MessagePost {
14     private String username; // username of the post's author
15     private String message; // an arbitrarily long, multi-line message
16     private long timestamp;
17     private int likes;
18     private ArrayList<String> comments;
```

Vérifier

Passed all tests! ✓

Correct

Note pour cet envoi : 1,00/1,00.

Description

Well, that was fun. You got to add a `display` method with about the same code to both the `MessagePost` and `PhotoPost` classes. You're undoubtedly saying to yourselves, "There's got to be a better way!?". And there is - using inheritance.

This is going to involve quite a modification of the code. We need to take a step back and look at the architecture of our application; then we need to refactor our code. Refactoring means changing the internals of the application but without changing the functionality, and it's a *Really Important*™ concept (in fact, the *Refactor* menu item in Eclipse is one of the most useful). We will apply the principle of DRY to avoid cut-n-paste code development.

Question 4

Correct

Note de 1,00 sur 1,00

Which attributes are duplicated between the `MethodPost` and `PhotoPost` classes:

Veuillez choisir au moins une réponse :

- [x] a. username ✓
- [ ] b. message
- [ ] c. fileName
- [ ] d. caption
- [x] e. timeStamp ✓
- [x] f. likes ✓
- [x] g. comments ✓

Vérifier

Your answer is correct.

Correct

Note pour cet envoi : 1,00/1,00.

Question 5

Correct

Note de 1,00 sur 1,00

Which methods are mostly duplicated between the MethodPost and PhotoPost classes (their implementations are identical or almost identical):

Veuillez choisir au moins une réponse :

- [x] a. like ✓
- [x] b. unlike ✓
- [x] c. addComment ✓
- [ ] d. getText
- [ ] e. getImageFile
- [ ] f. getCaption
- [x] g. getTimeStamp ✓
- [x] h. display ✓ Implementation almost identical for both classes
- [x] i. timeString ✓

Vérifier

Your answer is correct.

Correct

Note pour cet envoi : 1,00/1,00.

Description

We will re-architecture the existing code to get rid of the duplication.

Start a new package by copying everything from `facebouc.v1` into `facebouc.v2`. From now on all changes will be to `v2`.

The basic idea is to move as much as possible that is duplicated between the existing classes into a new `Post` class. This will serve as a *base class* for both *derived classes* `MessagePost` and `PhotoPost`. Stuff in the base class will be declared only once and will be inherited by the derived classes. And voilà - no dupes!

Question 6

Correct

Note de 2,00 sur 2,00

Develop a `facebouc.v2.Post` class by copying all duplicated attributes and methods from `MessagePost`. You'll have to modify `Post#display` since not all attributes used in `MessagePost#display` are visible in `Post`.

Paste your `Post` class into the Answer.

For example:

Test	Result

`Post mp = new Post("Leonardo da Vinci");`
`mp.display();`
	`Leonardo da Vinci`
	`0 seconds ago`
	`  No comments.`

Réponse:

```
1  package facebouc.v2;
2
3  import java.util.ArrayList;
4
5  class Post{
6      private String username; // username of the post's author
7      private long timestamp;
8      private int likes;
9      private ArrayList<String> comments;
10
11     public Post(String author) {
12         username = author;
13         timestamp = System.currentTimeMillis();
14         likes = 0;
15         comments = new ArrayList<>();
16     }
17     /**
18      * Record one more 'Like' indication from a user.
```

Vérifier

	Test	Expected	Got

✓	`Post mp = new Post("Leonardo da Vinci");`
	`mp.display();`
		`Leonardo da Vinci`
		`0 seconds ago`
		`  No comments.`
			`Leonard`
			`0 secon`
			`  No c`

✓	`Post mp = new Post("Homer Simpson");`
	`mp.like();`
	`mp.like();`
	`mp.unlike();`
	`mp.display();`
		`Homer Simpson`
		`0 seconds ago  -  1 people like this.`
		`  No comments.`
			`Homer S`
			`0 secon`
			`  No c`

✓	`final Post tsmp = new Post("Taylor Twit");`
	`IntStream.range(0, 12345).forEach(i -`
	`> tsmp.addComment("That sucks."));`
	`tsmp.display();`
		`Taylor Twit`
		`0 seconds ago`
		`  12345 comment(s). Click here to view.`
			`Taylor`
			`0 secon`
			`  1234`

Passed all tests! ✓

Correct

Note pour cet envoi : 2,00/2,00.

Question 7

Correct

Note de 2,00 sur 2,00

Whatever you copied into `Post` you can now remove from `MessagePost`. But in order to still get access to it, by inheritance, you'll have to indicate that `MessgePost` is derived from `Post` by changing the class signature:

```
public class MessagePost extends Post
```

You'll have to paste both `MessagePost` and `Post` into Answer.

For example:

Test	Result

`MessagePost mp = new MessagePost("Leonardo da Vinci", "Code this...!");`
`mp = new MessagePost("Leonardo da Vinci", "Code this...!");`
`mp.display();`
	`Leonardo da Vinci`
	`0 seconds ago`
	`  No comments.`

Réponse:

```
1  package facebouc.v2;
2
3  import java.util.ArrayList;
4
5  /**
6   * This class stores information about a post in a social network. The main part
7   * of the post consists of a (possibly multi-line) text message. Other data,
8   * such as author and time, are also stored.
9   *
10  * @author Michael Kölling and David J. Barnes
11  * @author Florian
12  * @author Peter Sander
13  */
14 public class MessagePost extends Post{
15      private String username; // username of the post's author
16      private String message; // an arbitrarily long, multi-line message
17      private long timestamp;
18      private int likes;
```

Vérifier

	Test

✓	`MessagePost mp = new MessagePost("Leonardo da Vinci", "Code this...!");`
	`mp = new MessagePost("Leonardo da Vinci", "Code this...!");`
	`mp.display();`

✓	`MessagePost mp = new MessagePost("Leonardo da Vinci", "Code this...!");`
	`mp = new MessagePost("Homer Simpson", "Facts are meaningless. You can use facts to prove any`
	`mp.like();`
	`mp.like();`
	`mp.unlike();`
	`mp.display();`

✓	`final MessagePost tsmp = new MessagePost("Taylor Twit", "Party Like It's 1989\\n(c)1989 Tayl`
	`IntStream.range(0, 12345).forEach(i -> tsmp.addComment("That sucks."));`
	`tsmp.display();`
	
	`// note that \\n is an artefact of the CodeRunner test`
	`// you will use the usual single backslash in your testing code instead`

Passed all tests! ✓

Correct

Note pour cet envoi : 2,00/2,00.

Question 8

Correct

Note de 2,00 sur 2,00

Whatever you copied into `Post` you can now remove from `PhotoPost`. But in order to still get access to it, by inheritance, you'll have to indicate that `PhotoPost` is derived from `Post` by changing the class signature:

```
public class PhotoPost extends Post
```

Paste both `PhotoPost` and `Post` into Answer.

Réponse:

```
1  package facebouc.v2;
2
3  import java.util.ArrayList;
4
5  /**
6   * This class stores information about a post in a social network. The main part
7   * of the post consists of a photo and a caption. Other data, such as author and
8   * time, are also stored.
9   *
10  * @author Michael Kölling and David J. Barnes
11  * @author Florian
12  * @author Peter Sander
13  */
14 public class PhotoPost extends Post{
15      private String username; // username of the post's author
16      private String filename; // the name of the image file
17      private String caption; // a one line image caption
18      private long timestamp;
```

Vérifier

Passed all tests! ✓

Correct

Note pour cet envoi : 2,00/2,00.

Question 9

Correct

Note de 4,00 sur 4,00

Now comes the part where you see more economies of code. Any reference to the specific classes MessagePost and PhotoPost in NewsFeed can be replaced by a reference to the base class Post, eg, you won't need two arrays since a single declaration

```
List<Post> posts = new ArrayList<>();
```

can handle both subtypes.

And you'll only need one `addPost` method.

The result of testing your application should look something like:

```
Homer Simpson
0 seconds ago  -  1 people like this.
   1 comment(s). Click here to view.

0 seconds ago  -  4 people like this.
   1 comment(s). Click here to view.
```

Note that this is a bit different from what v1 displayed.

Paste all your classes `MessagePost` , `PhotoPost`, `Post`, `NewsFeed` into the Answer.

Réponse:

```
1  package facebouc.v2;
2
3  import java.util.ArrayList;
4
5  /**
6   * This class stores information about a post in a social network. The main part
7   * of the post consists of a (possibly multi-line) text message. Other data,
8   * such as author and time, are also stored.
9   *
10  * @author Michael Kölling and David J. Barnes
11  * @author Florian
12  * @author Peter Sander
13  */
14 public class MessagePost extends Post{
15     private String username; // username of the post's author
16     private String message; // an arbitrarily long, multi-line message
17     private long timestamp;
18     private int likes;
```

Vérifier

Passed all tests! ✓

Correct

Note pour cet envoi : 4,00/4,00.

Description

So, your code works, your tests pass and everything is just great. Right? Well, if you're awake, you'll have noticed a slight problem.

In v1 a message post was printed as

```
Leonardo da Vinci
Code this...!
0 seconds ago
   No comments.
```

whereas the same message in v2 looked like

```
Leonardo da Vinci
0 seconds ago
   No comments.
```

A photo post in v1 was

```
Alexander Graham Bell
  [handset.png]
  Coming soon: the Samsung Galaxy S13.
0 seconds ago  -  4 people like this.
   1 comment(s). Click here to view.
```

whereas in v2 it looked like

```
Alexander Graham Bell
0 seconds ago  -  4 people like this.
   1 comment(s). Click here to view.
```

What's going on?

Question 10

Correct

Note de 1,00 sur 1,00

What explains the difference in display between v1 and v2?

Veuillez choisir au moins une réponse :

- [ ] a. Random quantom fluctuation
- [x] b. `Post#display` only has access to attributes in `Post`. Hence it can't display the specific attributes, eg, `MessagePost#message`, that were left in the derived classes. ✓

Vérifier

Your answer is correct.

Correct

Note pour cet envoi : 1,00/1,00.