M255 Errata & Clarifications – Version 14
April 2012

Software Guide – Java Reference, page 4

In the documentation for the class `OUColour`, the heading **Instance variables**, should be **Class constants**.

Software Guide – Java Reference, page 12

The comment for the `transfer()` method:

*If the `balance` of the receiver is equal to or greater than the argument `anAmount`, the `balance` of the receiver is debited by the argument `anAmount`. The argument **`anAccount`** is then credited by the argument `anAmount` and the method returns `true`, otherwise `false` is returned.*

should be:

*If the `balance` of the receiver is equal to or greater than the argument `anAmount`, the `balance` of the receiver is debited by the argument `anAmount`. The argument **`toAccount`** is then credited by the argument `anAmount` and the method returns `true`, otherwise `false` is returned.*

Note: This mistake can be found in all the `Account.java` project files.

Software Guide – Java Reference, page 21

The comment for the `compareToIgnoreCase(String str)` method states:

Compares the receiver to the argument string character by character as for the `compare()` method, but ignoring case differences.

The comment should read:

*Compares the receiver to the argument string character by character as for the **`compareTo()`** method, but ignoring case differences.*

Software Guide – Java Reference, page 26

The comment for the `removeAll(Collection aCol)` method states:

*Removes **one** occurrence of each of the argument's elements from the receiver. Returns true if the operation was successful, false otherwise.*

The comment should read:

*Removes **every** occurrence of each of the argument's elements from the receiver. Returns true if the operation was successful, false otherwise.*

Software Guide – Java Reference, page 42

The header for the second `FileWriter` constructor is shown as:

```
FileWriter(File file boolean append) throws IOException
```

There is a comma missing, the header should be:

```
FileWriter(File file, boolean append) throws IOException
```

Printed for web publication by
The Open University

Course Guide, page 13

On page 13 of the Course Guide, it states that:

You will not be permitted to take any notes or units into the examination with you, but you will be able to take your printed copy of the Software Guide (with no annotations).

Please note that this information should match that which is stated on Page 3 of the Software Guide - Java Reference, which reads:

You will be allowed to take this booklet into the examination, however the following conditions apply. You can only take the version of this booklet that was printed and sent to you by the Open University. This booklet must be free of any annotations, except for any errata that may appear on the M255 website.

Solution to Specimen Exam, Q22 (ii)

The first line of the solution reads:

```
public class DanceableHoverFrog() extends HoverFrog implements Danceable
```

It should be:

```
public class DanceableHoverFrog extends HoverFrog implements Danceable
```

Unit 1, page 29

Please note that StarOffice is no longer distributed on CD-ROM and has also been superseded by OpenOffice. This is available through the OU Software Downloads page:

http://learn.open.ac.uk/mod/oucontent/view.php?id=371253&section=1.4

Unit 3, page 25

Second paragraph refers to the *"Java Handbook"* when it should refer to the *"Software Guide - Java Reference"*.

Unit 3, page 27, Activity 3

If you copy the expressions numbered 1, 12 and 17 into the OUWorkspace and attempt to evaluate them you will get the following error message:

```
Error at line 1, column 18. Encountered: "\u2013" (8211), after : ""
```

This is because what look like minus signs in the pdf file are actually en-dashes which are slightly longer than minus signs. The error will go away if you replace `"–"` with `"-"`.

Unit 4, page 45, Discussion of Activity 24, point 3

In the method `getHomePosition()`, the line

```
return homePosition;
```

while perfectly correct does not conform to an M255 programming guideline that instance variables should always be prefixed by `this` in the instance methods of the class that declares them. Hence the line would have been better written as:

```
return this.homePosition;
```

Unit 4, page 58, Discussion of Activity 30

In part 4 of this discussion, the body of the method `alignLeftArm()` consists of these lines of code:

```
this.leftArm.setXPos(this.getXPos() - 35);
this.leftArm.setYPos(this.getYPos() + 25);
```

and the body of the method `alignRightArm()` consists of these lines of code:

```
this.rightArm.setXPos(this.getXPos() + 25);
this.rightArm.setYPos(this.getYPos() + 25);
```

While perfectly correct, these lines of code do not conform to an M255 programming guideline that where getter methods have been provided for instance variables, they should always be used to access those instance variables. Hence the lines of code would have been better written as:

```
this.getLeftArm().setXPos(this.getXPos() - 35);
this.getLeftArm().setYPos(this.getYPos() + 25);

this.getRightArm().setXPos(this.getXPos() + 25);
this.getRightArm().setYPos(this.getYPos() + 25);
```

Unit 4, Unit4_Project_3

In Activity 10, you were asked to add a new instance variable (`flyCount`) to the `Frog` class and to initialise this new instance variable to zero in the class's constructor.

In Unit4_Project_3 we failed to provide you with this modified constructor. The modified constructor should be:

```
public Frog()
{
   super();
   this.colour = OUColour.GREEN;
   this.position = 1;
   this.flyCount = 0;
}
```

Note that although the constructor we did provide doesn't initialise `flyCount`, all the methods that rely on `flyCount` will still work because Java automatically initialises `int` instance variables to zero. However, in M255 we encourage you to make the initialisation of each instance variable explicit in the constructor even if Java would automatically give it the value you want.

Unit 4, Unit4_Project_5 & Unit4_Project_5_Completed

The constructor for the `Diamond` class:

```
public Diamond()
{
   colour = OUColour.GREEN;
   xPos = 0;
   yPos = 0;
   width = 20;
   height = 20;
}
```

while perfectly correct does not conform to an M255 programming guideline that instance variables should always be prefixed by `this` in the instance methods of the class that declares them. Hence the code would have been better written as:

```
public Diamond()
{
   this.colour = OUColour.GREEN;
   this.xPos = 0;
   this.yPos = 0;
   this.width = 20;
   this.height = 20;
}
```

Unit 5, page 11, Activity 2

Part 5 of this activity omits to tell you to add to the `Account` class the import statement:

```
import.ou.*;
```

as the very first line in the class. This is necessary in order to use the `OUDialog` class within `Account`.

Unit 5, Page 27

The paragraph before Activity 6 states

"You can test whether an integer `x` is odd using an expression such as `x % 2 == 1`"

However this only works if `x` isn't negative. For example `-3 % 2` evaluates to `-1` not `1`, so `-3` would not be recognised as an odd number.

A better test would be `x % 2 != 0` which would work correctly whether `x` is positive or negative.

Unit 5, page 28, discussion of Activity 7

In the `rightIfGreen()` method, the `if` statement's `Boolean` condition is given as:

```
this.getColour() == OUColour.GREEN
```

While this code is correct and works, a much better `Boolean` condition would have been

```
this.getColour().equals(OUColour.GREEN)
```

This is better is because we are interested in whether the colour returned by `getColour()` and the colour referenced by the class constant `OUColour.GREEN` represent the same colour (i.e. have the same `int` values for their `red`, `green` and `blue` instance variables, rather than whether they are the same object.

Unit 6, page 10

The third margin note states that:

*"the OU class library can be obtained by selecting OU Class Library from the **Tools** menu in the main BlueJ window"*.

This should read:

*"the OU class library can be obtained by selecting OU Class Library from the **Help** menu in the main BlueJ window"*.

Unit 6, page 21

In the blue box, the second line of code in the body of the constructor:

`this.setHeight(0);` should read `this.height = 0;`

Unit 6, page 28, discussion of Activity 6

The last two paragraphs in the discussion of Activity 6 are in the wrong order, the last paragraph should be read first.

Unit 6, page 50

The first line of code in the constructor shown at the bottom of the page has a semi-colon missing after `super()`. In the following line `wclient1` should be `wClient1` and in the final line `wclient2` should be `wClient2`.

Unit 6, Unit6_Project_14

This is the first project file to introduce the `Amphibian` class. The methods `getPosition()`, `setPosition()`, `getColour()` and `setColour()` while being perfectly correct do not

conform to an M255 programming guideline that instance variables should always be prefixed by `this` in the instance methods of the class that declares them. This lapse in style is true for all copies of the `Amphibian` class that can be found in subsequent projects.

Unit 7, page 9

The example statement shown in line 9 of the table is shown as:

```
if (count == 5) total = 42;
```

While this code is perfectly correct, it does not conform to a M255 programming guideline given in page 22 of unit 5, which states:

"Where a statement block contains only a single statement it is permissible to leave the braces out. However we strongly advise against doing so, because it frequently leads to program bugs which are hard to detect."

Therefore the example statement would have been better written as:

```
if (count == 5)
{
   total = 42;
}
```

Unit 7, page 11, discussion of Activity 1

Towards the end of the discussion two lines start with *"sets the* `postion`*"* . These of course should be *"sets the* `position`*"*

Unit 7, page 28

The second paragraph to the Solution to Exercise 5 starts:

"The original version, which used `this.getPosition()`*, is fine, because* ***`getPostion()`****"*

This should be:

"The original version, which used `this.getPosition()`*, is fine, because* ***`getPosition()`****"*

Unit 7, page 32

In the `dance2()` method, 10 lines from bottom of page, a `(` is missing immediately after `||`.

Unit 7, page 47

The second line of code at the top of the page:

```
this.setspeed(90)
```

should be:

```
this.setSpeed(90)
```

Unit 8, page 21

In the discussion of Activity 5 the first line of code `Converter.costOfCurrency(300);` should have been written as:

```
dollarConverter.costOfCurrency(300);
```

Printed for web publication by
The Open University

Unit 8, page 33

About halfway down the page, the statement:

```
dollarConverter costOfCurrency(100);
```

should have been written as:

```
dollarConverter.costOfCurrency(100);
```

Unit 9, page 8

The penultimate word on the page should be **`String`** *not* **`Strong`**.

Unit 9, page 50

The code in the `main()` method:

```
welcome.start();
```

Should have an initial capital letter, i.e.,

```
Welcome.start();
```

Unit9_Activities_6-8 folder

In the QuizMarker.txt file, the code for the first line of code in the body of the `reportResults()` method is incorrect:

```
int percentage = (int)((this.pupilScore * 100.0 / QuizMarker.KEY.length) + 0.5);
```

There should not be a `this` in front on `pupilScore`. The code should be:

```
int percentage = (int)((pupilScore * 100.0 / QuizMarker.KEY.length) + 0.5);
```

There is an error in the 'command prompt' shortcut supplied for Activity 6 of unit 9 in that it contains a couple of superfluous quotation marks.

If you right click on the 'command prompt' shortcut and select Properties you will see that the target is:

```
C:\WINDOWS\system32\cmd.exe /k "set path="C:\Program Files\Java\jdk1.6.0_07\bin";
%path%&&set classpath=."
```

Assuming that you have accepted the default locations when installing the M255 software, this target needs to be edited to remove the quote just before *C:* and the quote just after *bin* so that it reads:

```
C:\WINDOWS\system32\cmd.exe /k "set path=C:\Program Files\Java\jdk1.6.0_07\bin;
%path%&&set classpath=."
```

You should then be able to run the activity as described in the unit.

Unit 10, page 25

The second sentence states:

*"For example, given the **set** created by the method `myDemo()` ..."*

This should be:

*"For example, given the **map** created by the method `myDemo()` ..."*

Unit 11, page 28, discussion of Activity 10

The code for the `setLastName()` method is given as:

```
public void setLastName(String memberLastName)
{
     this.lastName = LastName;
}
```

It should be:

```
public void setLastName(String memberLastName)
{
     this.lastName = memberLastName;
}
```

Unit 12, page 9

End of third paragraph reads:

"... always starts from the root directory of the disk, usually C: in Windows."

It should be:

"... always starts from the root directory of the disk, usually **C:** in Windows."

Unit 12, page 44

Line 14 reads:
"... you can see that second line contains only two tokens ..."
It should be:

"... you can see that the third line contains only two tokens ..."

Unit 12, page 46

Last sentence starts "Theses streams ...", it should be "These streams ..."

Unit 14_Project

The constructor for the `Form` class in this project is written as:

```
public Form(String aName, Teacher aTeacher)
{
   name = aName;
   teacher = aTeacher;
   pupils = new HashSet<Pupil> ();
}
```

While this is perfectly correct it does not conform to an M255 programming guideline that instance variables should always be prefixed by `this` in the instance methods (or constructors) of the class that declares them. Hence the code would have been better written as:

```
public Form(String aName, Teacher aTeacher)
{
   this.name = aName;
   this.teacher = aTeacher;
   this.pupils = new HashSet<Pupil>();
}
```

Printed for web publication by
The Open University