V1.0

School of Computing, Creative Technologies and Engineering

Assessment Brief

Component 1 COURSEWORK

Module name and CRN		Object Oriented Programming 21540			
Module Leader		Duncan Mullier			
Term	3	**Level**	4	**Approx No of Students**	100

COMPONENT TITLE: **Phase Test**

COMPONENT WEIGHTING: **40% of Module Marks**

HAND-OUT DATE: **at time of individual phase tests**

SUGGESTED STUDENT EFFORT: **1 hour x 2 sessions**

SUBMISSION DATE: **Weeks 6 and Week 10 Scheduled session a second chance with both tests together will be held in week 12.**

SUBMISSION INSTRUCTIONS:

- The phase test will take place on the VLE.

FEEDBACK MECHANISM:
Immediate feedback from the VLE.

LEARNING OUTCOMES ADDRESSED BY THIS COMPONENT:
Develop an understanding of the key object oriented concepts, such as classes, inheritance and polymorphism and have the ability to implement these in a modern object oriented language

NOTES:
The usual University penalties apply for late submission.

This is an individual assessment. Submission of an assessment indicates that you, as a student, have completed the assessment yourself and the work of others has been fully acknowledged and referenced. By submitting this assessed work, you are declaring that you are fit to submit, and you will therefore not normally be eligible to submit a request for mitigation for this work.

If you fail to attend the test at the scheduled date and time (or arrive more than 30 minutes late) without agreed mitigation, your result will be recorded as

Non-Submission. If your result is recorded as Non-Submission or your mark for this assessment and for the whole module is below 40%, you will have opportunity to take reassessment in July 2021 (see Reassessment information below). If you are granted deferral through the mitigation process, you may take the reassessment test with a full range of marks available.

For further information, please refer to your Course Handbook or University Assessment Regulations.

Reassessment

If your result for this assessment is recorded as Non-Submission or your mark for this assessment and for the whole module is below 40%, you will have opportunity to take reassessment phase test during reassessment in July (exact date to be announced nearer the time on OOP myBeckett page). If you are granted deferral through the mitigation process, you may complete the reassessment phase test with a full range of marks available.

DETAILS OF THE ASSESSMENT Task:

The phase test comprising of multiple choice/multiple answer questions of varying degrees of difficulty. You have a total of two phase tests to complete, you must sit them both.

You should take the phase test in a room on your own. You may use your notes no communication systems such as email, Facebook, instant messaging are permitted. If you fail to attend the phase test in your scheduled session and do not present acceptable mitigation to the mitigation pane you will be marked as “not submitted”. Please note: Tutors will follow up any suspected unfair practice as per University policy.

Your tests take place:

· test 1 35% of component 1 marks

· test 2 55% of component 1 marks

Week 12 100% of component 1 marks. This is a second chance where both tests are given together. You will receive the higher of the weeks 6 and 9 test or the week 12 test. To qualify you MUST have attended both the week 6 and 9 tests.

In addition there is a small test for each lecture. You can take each of these tests as many times as you like the combined mark will give 10% of component 1’s mark.

NOTES: The usual University penalties apply for late submission. By submitting this assessed work, you are declaring that you are fit to submit, and you will therefore not normally be eligible to submit a request for mitigation for this work. If you fail to attend the test at the scheduled date and time (or arrive more than 30 minutes late) without agreed mitigation, your result will be recorded as Non-Submission. If your result is recorded as Non-Submission or your mark for this assessment and for the whole module is below 40%,

you will have opportunity to take reassessment (see above). If you are granted deferral through the mitigation process, you may take the reassessment test with a full range of marks available.

Student Instructions for Submission of Phase test

The phase test will be presented to you, once enabled by your tutor in your scheduled session. Your allocated time will be displayed at the top of screen, it is your responsibility to monitor your allocated time and submit your work within the time allocation. Once submitted; the work will be automatically marked and you should be able to see your grade.

Phase test 1 is worth 35% of component 1 (which is worth 40% of the module), the second phase test in week 9 is worth 55% and the remaining 10% comes from the post lecture quizzes.

You will be given 35 questions each worth 1 mark (adding up to 35)

11 Questions from Variables

13 Questions from Selection and Iteration

11 Questions from Classes and Objects

Questions are drawn at random for a larger pool.

Phase 2 test is worth 55% of component 1 (which is worth 40% of the module), the first phase test in week 5 was worth 35% and the remaining 10% comes from the post lecture quizzes.

You will be given 55 questions each worth 1 mark (adding up to 55)

14 Questions from Designing Classes

14 Questions from Inheritance

14 Questions from Polymorphism

13 Questions from Arrays

Questions are drawn at random for a larger pool.

MARKING SCHEME / CRITERIA

Questions will be marked as right or wrong by the VLE.

School of Computing, Creative Technologies and Engineering

Assessment Brief

MAIN Component 2 COURSEWORK

Module name and CRN	*Object Oriented Programming 21540*				
Module Leader	*Duncan Mullier*				
Term	3	***Level***	*4*	***Approx No of Students***	*100*

COMPONENT TITLE: ***Graphics Program***

COMPONENT WEIGHTING: ***60% of Module Marks***

HAND-OUT DATE: ***Week 1***

SUGGESTED STUDENT EFFORT: ***30 hours***

SUBMISSION DATE: **On or before 9am 9th May 2022 (week 13)**

MUST demonstrate your work, via a recorded demo uploaded to YouTube, to receive a mark.

SUBMISSION INSTRUCTIONS: ***Submit via the VLE***

FEEDBACK MECHANISM:

Verbal feedback at demonstration.

LEARNING OUTCOMES ADDRESSED BY THIS COMPONENT:

LO 1	*Develop an understanding of the key object oriented concepts, such as classes, inheritance and polymorphism and have the ability to implement these in a modern object oriented language.*
LO 2	*Gain the ability to develop non-trivial computer programs made up of multiple files and classes in order to reflect modern object oriented based program architectures.*

LO 3	*Apply design concepts using a suitable Object Oriented modelling notation.*

NOTES:

The usual University penalties apply for late submission.

This is an individual assessment. Submission of an assessment indicates that you, as a student, have completed the assessment yourself and the work of others has been fully acknowledged and referenced. By submitting this assessed work, you are declaring that you are fit to submit, and you will therefore not normally be eligible to submit a request for mitigation for this work.

If you fail to attend the demonstration at the scheduled date and time without agreed mitigation, you will be given one further opportunity to demonstrate your work (incurring a 5% late penalty) at a time scheduled by the module team. If you miss this second opportunity, your result will be recorded as Non-Submission. If your result is recorded as Non-Submission or your mark for this assessment and for the whole module is below 40%, you will have opportunity to take reassessment in July 2020 (see Reassessment information below). If you are granted deferral through the mitigation process, you may attend the reassessment demonstration with a full range of marks available.

For further information, please refer to your Course Handbook or University Assessment Regulations.

Reassessment

If your result for this assessment is recorded as Non-Submission or your mark for this assessment and for the whole module is below 40%, you will have opportunity to take reassessment (see Reassessment information below). If you are granted deferral through the mitigation process, you may complete the reassessment with a full range of marks available. You must demonstrate your work during the week of 5th July 2021 (exact date to be announced on OOP myBeckett page nearer the time).

DEMONSTRATION

You are required to produce a demonstration video, using a script provided as a Google form. There are instructions as to how to do this (in terms of what software to use to produce your screen recording and the script you should use) on myBeckett. You MUST follow the instructions carefully and ensure you comply with them all. They will be provided on myBeckett in detail but involve:

1..following the script provided on a google form whilst screen recording your software

2..Submitting your software on myBeckett.

3..Uploading your screen recording demo to your student YouTube Account.

4..Submitting the link to your demo on a separate Google form.

DETAILS OF THE ASSESSMENT MARKING SCHEME / CRITERIA

Your Task:

For this assignment you are required to develop a programmatic solution, using the Java programming language, to the problem below. ***You will be required to demonstrate the completed work to your tutor in order to receive a mark***. During the demonstration your tutor will examine your solution and ask questions about how the work was undertaken and any difficulties that you had. Any awarded mark is for the demonstration and explanation of the work, rather than for the submitted work itself.

Overview

The overall aim of this assignment is to implement a simple graphics tool. This must be built as a graphical application using the Java Swing and/or AWT classes. The software will allow users to type in simple commands which cause a virtual pen (sometimes it is also called a turtle after the Logo programming language which was popular in schools in the 1980s) to move around a virtual canvas area drawing lines as it moves.

Requirement 1 – basic application 20 marks

The first requirement is to produce a Java project which uses LBUGraphics, which is provided in a jar file available on the OOP myBeckett page, under Assessments. This class will do all the drawing and displaying when your class calls its methods (ensure you always have the latest version of the jar file, as with any software it may be updated). There is also documentation for this class in the same place on myBeckett, which has simple instructions on how to use it.

Mark Breakdown

Project complete using the provided Jar file LBUGraphics.jar. Your project should also have a main class (MainClass.java) which sets up the LBUGraphics class correctly. You should call the “about()” method in LBUGraphics which will output a simple graphic.
(10 marks)

You should now make another class called GraphicsSystem.java that uses inheritance to extend the LBUGraphics class and it should function as above (note you can go straight to this and still get the marks for the above).

(5 marks)

Command Processing

The LBUGraphics panel has a text field and a button. You should be able to respond to text typed into this text field. The LBUGraphics system will call your program's "processCommand()" method when the user presses return on the text field or clicks the ok button. You should make it so that your program only calls the "about()" method (to draw the dancing turtle" when the user types "about" into the text field and either presses return or clicks the "ok" button.

(5 marks)

Requirement 2 – command support 15 marks

The second requirement is to implement some basic commands to allow drawing. The user should be able to type in these commands within the text field provided by LBUGraphics. The application should be able to spot invalid commands and report this to the user. The commands to be supported are very explicit and MUST match those shown in the following table. The command MUST be typed in by the user and not selected from a menu as some of them will have parameters which MUST be typed together with the command, for example "forward 90". The parameters MUST not be entered separately, either after the command or in a separate text field. Note that these commands involve calling the methods inside the LBUGraphics object. You should look at the documentation for LBUGraphics.

When the program first runs or the reset command is issued, the turtle/pen should be set to the middle of the canvas and point down the screen and the pen should be set to "down". Hence if the first command was "forward 100" a line from the middle of the screen to nearer the bottom would be drawn.

Command	Description
`penup`	Lifts the pen from the canvas, so that movement does not get shown.

`pendown`	Places the pen down on the canvas so movement gets shown as a drawn line.
`turnleft <degrees>`	Turn <degrees> to the left
`turnright <degrees>`	Turn <degrees> to the right.
`forward <distance>`	Move forward the specified `distance`.
`backward <distance>`	Move backwards the specified `distance`.
`black`	Sets the output pen colour to black.
`green`	Sets the output pen colour to green.
`red`	Sets the output pen colour to red.
`white`	Sets the output pen colour to white.
`reset`	Resets the canvas to its initial state with turtle pointing down.
`clear`	Clears the display.

Note: A `<distance>` is an integer value, e.g. 100. The user does not surround the number with <>.

Turtle/Pen is in the middle of the screen pointing down (5 marks)

“penup” (1 mark)

“pendown” (1 mark1)

“turnleft” (2 marks) (1 mark only if no parameter)

“turnright” (2 marks) (1 mark only if no parameter)

“forward” (2 marks) (1 mark only if no parameter)

“backward” (2 marks) (1 mark only if no parameter)

at least four colour command (such as “red”) (3 marks)

“reset” move the turtle to the initial position and pointing down (1 mark)

‘‘new” clears display, turtle stays in the same position (1 mark)

35

Requirement 3 Validating Commands. 10 Marks

Reports invalid commands (i.e. something not on the commands list above). (2 marks)
Detects valid command with missing parameter. (2 marks)
Detects non numeric data for a parameter. (3 marks)
Correctly bounds parameters (i.e. negative or non sensible values are reported as errors). (3 marks)

45

Requirement 4 – loading, saving and exiting. 15 marks

“Load” and “Save” should allow the user save and load the image and to save and load a set of commands that the user has typed in.

The current image should be saved to a file and also be able to be loaded back into the system and drawing recommenced. If the user attempts to load a new image without the current one been saved first then a warning should be shown to the user, which should provide the opportunity for the current image to be saved first.

Save Image (to suitable image graphics format such as png or jpeg). (2 Marks)
Load Image and redisplayed on the display. (2 marks)
Saves all commands previously typed as a text file. (2 marks)
Loads a text file of commands and executes them. (5 marks)
Warning if the current image is not saved. (2 marks)
Use of GUI dialogues. (2 marks)

60

Requirement 5 – extending LBUGraphics using inheritance. 20 marks

You have so far used LBUGraphics.jar, which is a precompiled file of a class I have written called LBUGraphics.java (technically a jar file is like a zip file and if you open it with a zip program you will see it has LBUGraphics.class inside it, or open it in the project explorer of Eclipse).

You have already used inheritance because you have extended the LBUGraphics class. This is because you have been forced to add a processCommands(String) method. Without putting it you would get a syntax error because LBUGraphics needs to call it when it detects an event on the button or text field. You can do much more with inheritance however. You can add new methods and you can replace existing methods.

Override the about() method so that it still does the same "dance" but appends a message that contains your name**.** (4 marks)

Look at the documentation and you will see that there is an unimplemented circle command that you can replace with an implemented one. The turtle should remain in the original position after this command has been issued.

circle(radius) (2 marks)

A Pen Command that takes three parameters, one for red, one for green and one for blue to make an RGB colour. (4 marks)

A triangle command that draws equilateral triangles with one parameter. (3 marks)

A further triangle command with three parameters that draws any triangle (hint – look at polygons). (2 marks)

A fill command. The user should be able to type "fill" and the system will toggle the fill status (i.e. if it is currently set to not filled, change it to filled and if it is currently set to filled set it to not filled). The status of filled should be "not filled" when the program is first run. If

the status is “filled” then any shapes, such as circle and rectangle, should be drawn with a filled colour. (5 marks)

Overridden about method 4 marks.

Overridden circle method 2 marks.

Additional pen command 4 marks.

Triangle command with one parameter 3 marks.

Triangle command with three parameters 2 marks.

Fill command 5 marks.

80

Requirement 6 Use you imagination 20 marks

If you’ve got this far then you’ve done what I asked and (probably) have a first. Now you can go further than the module and explore other aspects of Java and program development. Here you can add anything you like to make your program more interesting. Try and make it something that is different. i.e. doing additional shapes is fairly easy once you have done one, so you may only get a mark each. Adding a GUI interface is more complicated though and worth more marks. Have a good look at the documentation for LBUGraphics so that you can see what it can do and what it can’t.

100

Deferral

If you have been given a deferral then you should complete the original assignment in full and provide a video recorded demo as described in the original assignment.

Reassessment

If you have been given a reassessment then you are to complete the original assignment to the following reduced specification.

You must use LBUGraphics.jar provided in the Assessment folder and install it correctly and use it to implement the following. Note it may be helpful to read the original assignment specification.

Note: A `<distance>` is an integer value, e.g. 100. The user does not surround the number with <>.

"penup"
"pendown"
"turnright"
"turnleft"
"forward <distance>"
"backward <distance>"
at least four colour command (such as "red")
"reset" move the turtle to the initial position and pointing down
‘‘new" clears display, turtle stays in the same position

Your application needs to additionally:

Reports invalid commands.
Detects and reports valid command with invalid parameter.
Detects and reports non numeric data for a parameter.

You should record your YouTube video as per the original instructions using the original form and upload your video to YouTube and provide the link in the submission box of the assignment.