LEEDS BECKETT UNIVERSITY
SCHOOL OF BUILT ENVIRONMENT, ENGINEERING & COMPUTING

COVER SHEET FOR ALL ASSIGNMENT BRIEFS

Name of Module	Advanced Software Engineering
Name of Module Leader	Dr Duncan Mullier
Main Assessment or Resit?	Main assessment
Semester 1 or 2 or Term 1, 2 or 3	1
CRN	10542
Type of Assessment (Coursework; Presentation; Phase Test etc)	Coursework
Date & deadline time of Submission	9/11/2020
Date for Return of feedback	Within 2 weeks.
Type of Submission (online via My Beckett; Handed in during Seminar; Presentation). It is expected that all assignments will be submitted electronically.	Online Via MyBeckett
Feedback (please specify how will feedback be given to students)	Via marksheet uploaded to myBeckett.
Franchise delivery Is the assessment for campus delivery the same for the franchise partner, if not please provide assessment for franchise partner.	Same.

Component 1 COURSEWORK

Module name and CRN			Advanced Software Engineering10542		
Module Leader			Duncan Mullier		
Term	1	**Level**	6	**Approx No of Students**	60

COMPONENT TITLE: Graphical Programming Language

COMPONENT WEIGHTING: 40 % of Module Marks

HAND-OUT DATE: (Week 1)

SUGGESTED STUDENT EFFORT: 20 hours

SUBMISSION DATE: 9am Monday 9/11/2020 (Week 6)

SUBMISSION INSTRUCTIONS:

Upload to myBeckett submission box. Zip up your entire project directory so you are submitting both your source code and the executable plus a screenshot of your version control commits page

You are required to make a screen recording of you demonstrating your work according to the provided script. This screen recording should also be uploaded in the separately provided upload box. Instructions of how to screen record are provided in detail below.

FEEDBACK MECHANISM:

- *You will receive feedback via the VLE*

LEARNING OUTCOMES ADDRESSED BY THIS COMPONENT:

Evaluate and demonstrate professional engineering style approaches to developing software systems
Develop underpinning and transferrable skills relating to the application of programming languages and environments.

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 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 with a submission date of 22nd May 2021 (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.

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 May 2020 and your marked capped at 40% (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.

DETAILS OF THE ASSESSMENT

Graphical Programming Language Application

This assignment is to use what you are learning in the module to produce a fairly complex program. The idea of the program is to produce a simplified environment for teaching simple programming concepts. You are to create a simple programming language and environment that has the basics of sequence, selection and iteration and allows a student programmer to explore them using graphics.

Note that the assignment has two components. They are marked separately. If you fail either part and your overall module mark is below 40% you are likely to get reassessment in those part/s that you failed. You should read both parts of the assignment to see where you are aiming. You could tackle them completely separately, i.e. worry about part two when it arrives and therefore treat part one as a prototype, but this will involve reworking part one for part two. For example to put design patterns into it which isn't on the marking scheme for part one. Or you could design part 2 elements in from scratch. Both methods are valid software engineering methods. I would at the very least be very familiar with part two whilst you are doing part one so that you can make an informed choice at the time.

Hand In 1 40%

1 Management --10 marks total

Note Version Control needs to be set up and used in part 1 but will be marked in part 2 for 10 marks.

Unit Tests (10 marks)

Fully implemented as unit tests. Note that the code that the tests are testing does not necessarily have to have been written (see Test Driven Development).

Unit Test project set up (2 marks)

Appropriate tests for part 1 set up (2 marks)

Appropriately documented.(3 marks)

Some appropriate tests for part 2 implementation (3 marks)

Note, I am not expecting you to implement code for part 2, just the tests for one or two elements.

2 Implementation -- 30 Marks total

Your implementation must have a proper interface with a window/area for typing a "program" into and a window/area for displaying the output of the "program". You should also have a command line where commands are executed immediately. The actual layout is up to you.

Appropriate UI conforming to above specification (1 mark)

Command parser class

Reads and executes commands on command line one at a time (2 marks)

Reads a program (in the program window) and executes it with a "run" command (typed into the command line). (5 marks)

Saves and loads a program (2 marks)

Syntax checking

Checks for valid commands (2 marks)

Checks for valid parameters (2 marks)

Basic drawing commands (all commands should be case insensitive)

Position pen (moveTo) (2 marks)

pen draw (drawTo) (2 marks)

clear command to clear the drawing area (1 mark)

reset command to move pen to initial position at top left of the screen (1mark)

Draw basic shapes:

rectangle <width>, <height> (2 marks)

circle <radius> (2 marks)

triangle <side>,<side>,<side> (2 marks)

Colours and fills

pen <colour> e.g pen red, or pen green (three or four colours). (2 marks)

fill <on/off> e.g. fill on, makes subsequent shape operations filled and not outline. (2 marks)

The program should be written using inheritance and design patterns (specifically marked in part 2) so that additional commands could easily be added without affecting the rest of the code. Marks may be deducted if this is not the case.

Assessment Brief

COURSEWORK COMPONENT 2

Module name and CRN			Advanced Software Engineering A 10542		
Module Leader			Dr Duncan Mullier		
Term	1	**Level**	6	**Approx No of Students**	60

COMPONENT TITLE: **Graphical Programming Language Application**

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

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

SUGGESTED STUDENT EFFORT: **30 hours**

SUBMISSION DATE: **9am Monday 23/11/2020 9 am (Monday week 9)**

SUBMISSION INSTRUCTIONS:

- *Upload to myBeckett submission box. Zip up your entire project directory so you are submitting both your source code and the executable.*
- *You are required to make a screen recording of you demonstrating your work according to the provided script. This screen recording should also be uploaded in the separately provided upload box. Instructions of how to screen record are provided in detail below.*

FEEDBACK MECHANISM:

- *You will receive feedback via MyBeckett.*

LEARNING OUTCOMES ADDRESSED BY THIS COMPONENT:

Evaluate and demonstrate professional engineering style approaches to developing software systems
Develop underpinning and transferrable skills relating to the application of programming languages and environ
Apply and critically evaluate advanced programming and design concepts.

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 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 with a submission date of 22 May 2021 (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.

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 (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.

Hand In 2 60 marks

Note: If you did not get a working prototype in part 1 then you may continue it for part 2. Your mark for part 1 will stand as it is but you may gain marks for part 2 from the part 1 marking scheme. Your mark cannot exceed the total marks for part 2.

Management 10 marks

Set up version control (10 marks)

Early commit (from part 1), project description this includes part 1 of the assignment.

At least five commits of software for part 1 and at least 5 for part 2 (no marks if less)

Description with each commit about what has been done/changed and what is to be done next.

For high marks I expect a professional standard with many and regular commits each time something significant has been added with comprehensive descriptions.

Further tests for new parts of spec and evidence of completion.

Programming commands – 30 marks

Variables - allows variables to be used in loop and as parameters to draw commands (5 marks)

Loop command (5 marks)

Repeats everything between Loop on the first line and "end" on a later line.

If statement (5 marks)

2 marks for one line

3 marks for block with "endif"

Syntax checking (5 marks)

Syntax of the program is checked before the program is run and reported appropriately.

Methods (10 marks)

This is quite complex and will require some thought.

Define a method with:

```
Method myMethod(parameter list)
     Line 1
     Etc
Endmethod
```

Call a method with:

```
myMethod(<parameter list>)
```

working methods without parameters (5 marks)

working with parameters (+5 marks)

3 Design and Implementation Standard --20 marks total

Use of design patterns - factory class (5 marks)

All shape classes should use appropriate inheritance but should also use the factory design pattern. It should be fairly straightforward to add additional shapes.

Use of additional design pattern/s (5 marks)

Code documented with XML tags, XML documentation produced (I want to see the documentation and not just the comments in the code). For high marks I expect documentation to a professional standard. (5 marks)

Use of exception handling (including user generated exceptions) (5 marks)

4 Additional functionality

Here you can come up with your own functionality. Here are some suggestions but you are free to come up with your own, however you should discuss them with your tutor first.

If you haven't already got 100% you may be able to get additional marks here.

Additional commands, one example might be to transform/rotate shape, more complex shapes and the drawing of shapes.

Command Examples

The pen position is stored in the drawing object. Commands should not be case sensitive.

```
DrawTo x,y
MoveTo x,y
Circle <radius>
Rectangle <width>, <height>
Triangle <base>, <adj>, <hyp>
Polygon [points,...]
```

Complex commands

```
If <variable>=10
      Line 1
      Line 2
Endif


Radius = 20
Width = 20
Height = 20
Count = 1
Loop for Count
      Circle radius
      Radius+10
      Rectangle width, height
      Width+10
      Height + 10
      Count+1
Endloop
```

REASSESSMENT

Part 1 reassessment

You MUST contact your tutor to confirm which part 1 reassessment you are to do. There are two versions of the reassessment for part 1 because you may have affectively done the first one as your part 2 assignment. If this is the case you will be directed to do the part 2 reassessment but CHECK with your tutor.

Part 1 reassessment
This variant is for people who failed part 1 but did not continue to implement it for part 2.

Implementation
To pass the reassessment you program should have ALL of the following facilities.

Your implementation must have a proper interface with a window/area for typing a program into and a window/area for displaying the output of the program. You should also have a command line where commands are executed immediately. The actual layout is up to you.

Appropriate UI conforming to above specification
Command parser class
Reads and executes commands on command line one at a time
Reads a program and executes it with a "run" command
Saves and loads a program
Syntax checking
Checks for valid commands
Checks for valid parameters
Basic drawing commands (all commands should be case insensitive)
Position pen (moveTo)
pen draw (drawTo)
clear command to clear the drawing area
reset command to move pen to initial position at top left of the screen (1mark)
Draw basic shapes:
rectangle(width, height)
circle(radius)

Part 2 Reassessment

To pass the reassessment you program should have ALL of the following facilities.

Management
Set up version control
At least five commits of software
Description with each commit about what has been done/changed and what is to be done next.
Unit Tests with at least three working Unit Tests

Programming commands
Variables - allows variables to be used in loop and as parameters to draw commands (5 marks)

Loop command (5 marks)
Repeats everything between Loop on the first line and "end" on a later line.

3 Design and Implementation Standard

Use of design patterns - factory class (5 marks)

All shape classes should use appropriate inheritance but should also use the factory design pattern.

Code documented with XML tags, XML documentation produced.

Screen Recording Demonstrations

You must produce a screen recording for demonstrating both parts 1 and 2. This will be done according to a separate script for each part. Your recording can be made with any software but I have recommendations for Windows, Linux and Mac here. Your screen recording should include an audio commentary explaining what you are doing (or just reading the script). It should also include showing which part f the marking scheme you are currently demonstrating (by showing the marking scheme).