Component 1 COURSEWORK

Module name and CRN	Advanced Software Engineering				
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 13/11/2023**

SUBMISSION INSTRUCTIONS:

You are to submit three things to the assignment upload box on myBeckett

1 A zip file of your entire project directory (source and executable) (see below)

2 A link to your YourTube demo *

3 A Link to your repository in your chosen version control system

* You are required to make a screen recording of you demonstrating your work according to the provided script. The script will be released closer to submission time.

The script must be in view in the recorded demo.

You must use a microphone and explain your code in the demo.

The script will be in the assessment directory. Instructions of how to screen record are provided in detail below.

Failure to comply with these instructions will result in one request to redo your demo with the penalty for late demo outlined below.

Recommended Software for Screen Recording

You will upload your screen demo to your student YouTube account see https://libanswers.leedsbeckett.ac.uk/faq/179494

You will then add the link to your YouTube video demo to the upload box when you submit your assignment code.

It may be that you are asked to do a live demonstration at a time to be arranged.

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 during the reassessment period (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 record and upload a demonstration by 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 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

Please read this carefully. You MUST produce the program as specified here, so you must be very cler what you are being asked to do. If you produce a different program you will received no marks.

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. You should 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 The Basic Application 40%

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. You will make life hard for yourself in part 2 if this is not the case.

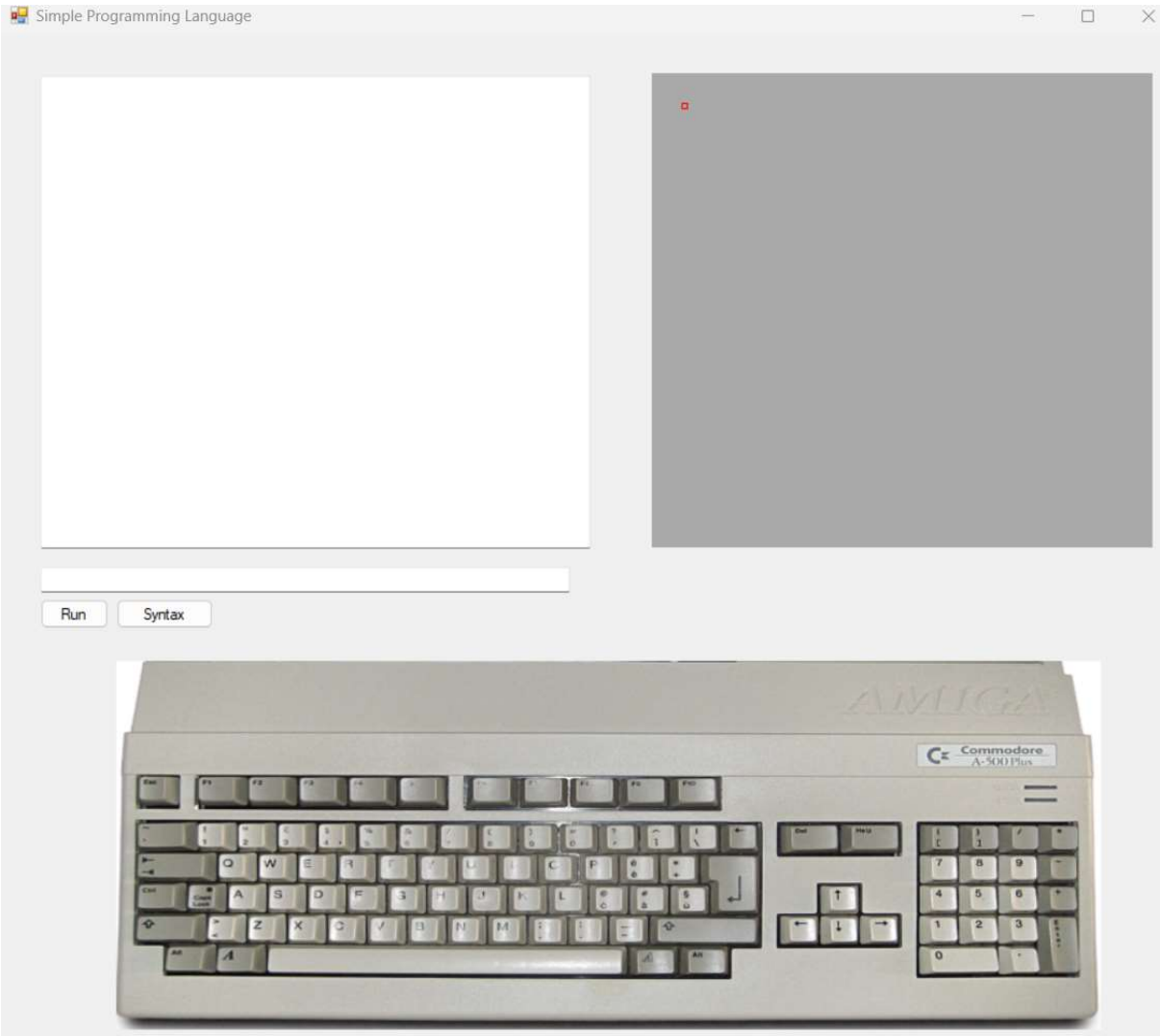
You will be asked to produce a video demonstration of your assignment (script to be provided) you MUST have a microphone and explain and show everything (details in the script). Failure to provide a demo will result in a zero mark.

1 Version Control System Set Up

You should set up your preferred Version Control System and commit the assignment specification and any other useful documents to it. This Must be done within the first two weeks of term (before Monday morning 9am week 3).
(5 marks)

2 Basic Graphical User Interface

Your graphical user interface should be a window/form with an area for receiving individual program commands, an area for receiving a complete program and an area for graphical program output. It should also have a button to run a program and to perform a syntax check.



This is what I have done. It has the three areas and buttons mentioned above and a fairly pointless picture (because I was trying to brighten it up). Yours doesn't have to look like this and you can have more facilities.

Once this has been set up you should minimally have a commit to your version control (feel free to do as many commits as you like).

(5 marks)

3 Command Parser Class
You should have a command parser class (if you have a different design then you must explain why in your demo and there must be a good reason).

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 to a text file (3 marks)
Syntax checking
Checks for valid commands (3 marks)
Checks for valid parameters (3 marks)
Any invalid results should be reported using exceptions (no marks for syntax checking if not).
Each of these facilities should be committed at least once to version control.
You should produce a Unit Test for each facility and the Unit test should have a separate commit to version control.
(16 marks)

4 Basic drawing commands (all commands should be case insensitive)
You will have had to implement a few commands to show part 3 working but here is the complete list of commands for the basic part 1 application.

Each command will only be considered as functional if:
It works

at least one commit with a good description
full XML documentation
a Unit test
XML documentation for the Unit test
At least one commit for the Unit test

4.1 Command List
Position pen (moveTo) 2 marks
pen draw (drawTo) 1 marks
clear command to clear the drawing area 1 mark
reset command to move pen to initial position at top left of the screen 1 mark
Draw basic shapes:
rectangle <width>, <height> 2 marks
circle <radius> 2 marks
triangle (you can do this any way you like) 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. 1 mark

(14 marks)

5 Explain Any Artificial Intelligence and libraries that you have used
You MUST have this section in your video demo. If you miss it out you will be asked to include it as a second chance and have the late demo penalty.

It is not wrong to use libraries and systems that help to produce your code (this is what using a high level language is). However, you must understand what you have done. If you use a library or facility then it frees up time to do more complex things.

You must give an overview of
any non standard libraries you have used
any AI tools you have used

any third party code you have used

If you have produced something that you don’t understand then you will receive no marks for it and are subject to the University’s usual Academic Integrity procedures.

Assessment Brief

COURSEWORK COMPONENT 2

Module name and CRN			Advanced Software Engineering A		
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 8/1/2023**

SUBMISSION INSTRUCTIONS:

You are to submit three things to the assignment upload box on myBeckett

1 A zip file of your entire project directory (source and executable) (see below)

2 A link to your YourTube demo *

3 A Link to your repository in your chosen version control system

* You are required to make a screen recording of you demonstrating your work according to the provided script. The script will be released closer to submission time.

The script must be in view in the recorded demo.

You must use a microphone and explain your code in the demo.

The script will be in the assessment directory. Instructions of how to screen record are provided in detail below.

Failure to comply with these instructions will result in one request to redo your demo with the penalty for late demo outlined below.

Recommended Software for Screen Recording

You will upload your screen demo to your student YouTube account see https://libanswers.leedsbeckett.ac.uk/faq/179494

You will then add the link to your YouTube video demo to the upload box when you submit your assignment code.

It may be that you are asked to do a live demonstration at a time to be arranged.

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 environm
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 during the reassessment period (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 record and upload a demonstration by 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 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.

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.

1 Programming commands – 50 marks

The idea here is that it behaves like a proper programming language with sequence, selection and iteration. See the *Commands Examples* section below.

For each facility you should show
- at least one commit with a good description
- full XML documentation
- a Unit test
- XML documentation for the Unit test
- At least one commit for the Unit test

1.1 Variables

allows variables to be used in loop to determine the number of iterations and as parameters to draw commands
(10 marks)

1.2 If statement

An if statement should have a condition and associated "endif" which denotes the end of a block

(5 marks)

1.3 Loop command

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

1.4 Syntax checking

Syntax of the program is checked before the program is run and reported appropriately. This should be implemented using exceptions (no marks will be given if it is not). You should make your own exception class and throw an object of that class passing an appropriate message object to each layer of your code.
(5 marks)

1.5 Methods

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
(10 marks)

1.6 Multiple Programs
(10 marks)

This facility allows the user to have two program windows and run two separate programs that output to the **same** output window.
In order to do this you will need to use threads and allow thread safe access to a single canvass.

2 Design and Implementation Standard 5 marks

Use of design patterns - factory class (2 marks)
All user created objects from classes that you have designed should use appropriate inheritance but should also use the factory design pattern. It should be fairly straightforward to add additional classes to the factory.

Use of additional design pattern/s (3 marks)

3 Additional functionality – 10 marks

Here you can come up with your own functionality. They need to be substantial functionality that would take you several hours to work out and complete. Here are some suggestions but you are free to come up with your own, however you should discuss them with your tutor first.

Extra simple commands, such as “red” or extra simple GUI components would not attract any extra marks.

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

4 Explain Any Artificial Intelligence and libraries that you have used
You MUST have this section in your video demo. If you miss it out you will be asked to include it as a second chance and have the late demo penalty.

It is not wrong to use libraries and systems that help to produce your code (this is what using a high level language is). However, you must understand what you have done. If you use a library or facility then it frees up time to do more complex things.

You must give an overview of
any non standard libraries you have used
any AI tools you have used
any third party code you have used

If you have produced something that you don’t understand then you will receive no marks for it and are subject to the University’s usual Academic Integrity procedures.

Appendix
Command Examples
The commands MUST be exactly as defined below to receive full marks.

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>
Or you could have
triangle width, height where it defines a box that the triangle is
drawn in
Polygon [points,...]
```

Complex commands

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


Radius = 20
Width = 20
Height = 20
```

Count = 1

```
While  Count < 10
      Circle radius
      Radius = Radius+10
      Rectangle width, height
      Width = Width+10
      Height = Height + 10
      Count = Count+1
Endloop
```

Assessment Brief

REASSESSMENT

Module name and CRN			Advanced Software Engineering A 11732		
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: **Reassessment period, see my**

SUBMISSION INSTRUCTIONS:

You are to submit three things to the assignment upload box on myBeckett

1 A zip file of your entire project directory (source and executable) (see below)

2 A link to your YourTube demo *

3 A Link to your repository in your chosen version control system

* You are required to make a screen recording of you demonstrating your work according to the provided script. The script will be released closer to submission time.

The script must be in view in the recorded demo.

You must use a microphone and explain your code in the demo.

The script will be in the assessment directory. Instructions of how to screen record are provided in detail below.

Failure to comply with these instructions will result in one request to redo your demo with the penalty for late demo outlined below.

Recommended Software for Screen Recording

You will upload your screen demo to your student YouTube account see https://libanswers.leedsbeckett.ac.uk/faq/179494

You will then add the link to your YouTube video demo to the upload box when you submit your assignment code.

It may be that you are asked to do a live demonstration at a time to be arranged.

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 environments..
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 you fail to record and upload a demonstration by 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 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.

REASSESSMENT

The deadline is posted on myBeckett.
You must record a demo using the original script, upload it to your YouTube account and paste the link into the submission. Yu must also provide a link to your repository on your Version Control system.

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 facilities in part 1 fully working (with Version Control, XML comments and Unit Tests). It should be have full version control, XML comments and Unit Tests for everything.

Explain Any Artificial Intelligence and libraries that you have used

You MUST have this section in your video demo. If you miss it out you will be asked o include it as a second chance and have the late demo penalty.

It is not wrong to use libraries and systems that help to produce your code (this is what using a high level language is). However, you must understand what you have done. If you use a library or facility then it frees up time to do more complex things.

You must give an overview of

- any non standard libraries you have used
- any AI tools you have used
- any third party code you have used

If you have produced something that you don't understand then you will receive no marks for it and are subject to the University's usual Academic Integrity procedures.

Part 2 Reassessment

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

Programming commands (with Version Control, XML comments and Unit Tests)

Variables - allows variables to be used in loop and as parameters to draw commands

Loop command

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

Design and Implementation Standard

Use of design patterns - factory class

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

Explain Any Artificial Intelligence and libraries that you have used

You MUST have this section in your video demo. If you miss it out you will be asked o include it as a second chance and have the late demo penalty.

It is not wrong to use libraries and systems that help to produce your code (this is what using a high level language is). However, you must understand what you have done. If you use a library or facility then it frees up time to do more complex things.

You must give an overview of

- any non standard libraries you have used
- any AI tools you have used
- any third party code you have used

If you have produced something that you don't understand then you will receive no marks for it and are subject to the University's usual Academic Integrity procedures.

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).