COM5003

Programming

<u>Portfolio</u>

Date for Submission: Please refer to the timetable on ilearn

(The submission portal on ilearn will close at 14:00 UK time on the date of submission)

Assignment Brief

As part of the formal assessment for the programme you are required to submit a **Programming** assignment. Please refer to your Student Handbook for full details of the programme assessment scheme and general information on preparing and submitting assignments.

Learning Outcomes:

After completing the module, you should be able to:

1. Identify simple algorithms in an Object-Oriented language; along with apply simple testing and debugging techniques in program development.

2. Design systems specifications using object-oriented modelling techniques; and apply these using an appropriate computer assisted software engineering (CASE) tools.

3. Use appropriate techniques to apply inheritance in the design and implementation of classes; along with aggregation techniques.

4. Design and implement solutions to a small number of applications that consist of a limited number of classes within an Object-Oriented language.

5. Formulate solutions to simple programming problems

Guidance

Your assignment should include: a title page containing your student number, the module name, the submission deadline and the exact word count of your submitted document; the appendices if relevant; and a reference list in AU Harvard system. You should address all the elements of the assignment task listed below. Please note that tutors will use the assessment criteria set out below in assessing your work.

You must not include your name in your submission because Arden University operates anonymous marking, which means that markers should not be aware of the identity of the student. However, please do not forget to include your STU number.
Maximum word count: 4000 words

Please refer to the full word count policy which can be found in the Student Policies section here: Arden University | Regulatory Framework

Please note the following:

Students are required to indicate the exact word count on the title page of the assessment.

The word count includes everything in the main body of the assessment (including in text citations and references). The word count excludes ***numerical data in tables***, ***figures, diagrams, footnotes, reference list and appendices. ALL other printed words ARE included in the word count.***

Please note that exceeding the word count by over 10% will result in a 10-percentage point deduction.

Assignment Task

Introduction

This assignment is aimed at allowing you to demonstrate a holistic set of skills in object-oriented programming and systems engineering as required for the design and implementation of real-world Internet applications. This includes the adoption of UML-based systems design concepts and Java-based coding techniques ensuring professional software development. You are required to complete a portfolio of tasks related to a specific case study described below. These tasks constitute a comprehensive design of an electronic system to support the core system's functionality. Consider yourself a senior software engineer delivering this assignment as a stand-alone project for your organisation.

Case study

Walking & Boarding is a start-up business that initially started as a dog-walking business and is growing in leaps and bounds. The founder based on input from their dog-walkers decided to start an online operation to connect stressed and busy dog owners with retired and other people with spare time on their hands to take care of their dogs during the day or board overnight.

They also provide beach adventure days for the more active hounds who are comfortable with travelling longer distances and have brilliant recall. The beach adventure day is a 3-4 hour long beach walk including plenty of play time, swimming and exploring old ship wrecks; a new and exciting environment for busy noses. Gourmet chicken lunch is included. The cost of each beach adventure day is £50 for each dog. Dog owners receive one free adventure day for their dog if they have booked six adventure days.

Walking & Boarding is a nationwide business. The business is online only. They however do have a customer support which comprises of a team of one manager and five customer service consultants. The role of customer support is to analyse feedback, process complaints, refunds and also manage memberships for dog owners and dog walkers/boarders.

Below is the process of how the range of services work.

The dog owner registers on the website and specifies the following information:

- Their name, address, email and mobile number
- The number of dog(s), type, sex and age of each dog to be walked
- Days and times when dog walking is needed
- Frequency (one-off, weekly, daily, multiple walks per day)
- Expected duration and/or length of each walk
- Any medical issues with the dog(s)
- Additional information (local parks, fields, cycleways etc. to be used)

The owner will then be issued with a unique membership number (such as Owner1234) and a unique 'job posting' number (such as Job4321). Each job posting requires a payment of £10. Their personal data is never shared but details of the dog(s) and walking times/preferences are freely available to potential dog walkers and dog minders.

Potential dog walkers/carers/boarders must also register the following data:

- Their name, address, email and mobile number
- Days and times available to walk dogs
- Days and times available to care/board dogs
- Preferred frequency (one-off, weekly, daily, multiple walks per day)
- Limitations (breeds they will not walk, or medical issues they cannot handle)
- Dog walking experience
- Hourly/Daily rate

Once registered and vetted, each 'walker/boarder/carer' is also allocated a membership number (such as Walker2345) and they can see all posted jobs for free (but not the owners' personal data). If they see a job they are interested in, they tick a box on the website and their details (membership number, hourly fee and experience, but not contact details) are emailed to that owner. So, both parties are connected but cannot communicate – yet.

It is then down to the owner to decide if they want to offer the job to that walker. If they reject the offer, they can request up to another three ‘offers’ from other walkers.

If they refuse all four offers, they must pay a ‘re-posting’ fee of £8. If they accept the offer, the website connects the walker and owner (sharing names and contact data) and the company takes 20% from the dog-walker/dog-minder as ‘introduction’ fee for each job that is completed. For example if a dog walker is paid £10 for walking a dog for 30 minutes, Walking & Boarding will receive £2.

Answer the following questions.

Task 1

Draw a UML Class diagram and object diagram to capture the classes/objects, class hierarchy and relationships between the classes for the system described in the case-study. Make sure to specify multiplicities for all the relationships shown in your diagrams.

(20 marks)

(LO2)

Task 2

Implement the class diagram you have created for the case-study in Java. Add appropriate methods to initialize and read the instance variables in the classes. Test your code by printing the variables and sample values.

(20 marks)

(LO1)

Task 3

a. Create a Java menu based program to demonstrate the following options:
    1. Registering as a dog owner
    2. Registering as a dog walker/minder
    3. Sign-in/Sign-out
    4. Posting a job by the dog owner
    5. Posting an offer to take a job by a dog walker/minder
    6. Booking an adventure day by the dog owner for their dog

The program on execution should display the menu as listed above, the user should select an option (input a value). Demonstrate the working of the program by choosing various options (the final output should demonstrate the output for all the methods) and printing sample output for each as mentioned below.

Option1: Their name, address, email and mobile number. Example: Ms. AB; 31, City Centre, New Town, UK; ab@newemail.com; 1234567890
Option 2: Their name, address, email, mobile number, services provided and fee for each.
Option 3: Dog's details and service required (walking, day care etc)
Option 4: Accept and offer to walk/care by replying 'Hello Ms. AB, I would be happy to accept the job posted'
Option 5: Date of adventure day, dog owners details and dog details.

Write suitable methods in the Java classes created in Task 2 to demonstrate the selection and execution of each option.

(30 marks)

b. Using the class containing the method for **option 4** as a base class, extend it to demonstrate inheritance. Write the Java code for this base class and up to three derived classes and **implement this hierarchy in Java.** Test your classes by printing the inherited variables and any calculations conducted in the methods both in the base class and the derived classes.

(15 marks)

c. For this assignment consider the database to be text files. In these text files, create records for a minimum of five dog walkers and three dog owners. Example: Assume dog owner 'A' books dog walker 'Joe Smith' for a 30 minute walk on Monday for £12 and dog walker 'Aaron Bloggs' for a 45 minute walk on Wednesday for their dog for £30. **Implement the calculation of the total amount payable by 'A' to the dog walkers and what is the amount received by Walking and Caring.** Demonstrate this calculation functionality as an abstract method. Test your code by printing the calculation and resulting values.

(15 marks)
(LO3, LO4, LO5)
Task 3 (60 marks)

Additional instructions

- Responses to all the tasks, including UML diagrams and examples of Java code must be put together and submitted as a single Word document.

- Each task involving UML diagrams must be accompanied by a brief critical discussion with references to relevant literature (up to 250 words per task).

- Each example of Java code must be properly documented using comments.

- Screenshot evidence of execution must be included for each example of Java code

- Screenshot of the sample output for each Java program must be included

- A plain-text copy of the Java code must also be appended

Formative Feedback

You have the opportunity to submit a draft to receive formative feedback.

The feedback is designed to help you develop areas of your work and it helps you develop your skills as an independent learner.

If you are a distance learning student, you should submit your work, by email, to your tutor, no later than 2 weeks before the actual submission deadline. If you are a blended learning student, your tutor will give you a deadline for formative feedback and further details.

Formative feedback will not be given to work submitted after the above date or the date specified by your tutor - if a blended learning student.

Referencing Guidance

You **MUST** underpin your analysis and evaluation of the key issues with appropriate and wide ranging academic research and ensure this is referenced using the AU Harvard system.

Follow this link to find the referencing guides for your subject: Arden Library

Submission Guidance

Assignments submitted late will not be accepted and will be marked as a 0% fail.

Your assessment can be submitted as a single Word (MS Word) or PDF file, or, as multiple files.

If you chose to submit multiple files, you must name each document as the question/part you are answering along with your student number ie Q1 Section A STUXXXX. **If you wish to overwrite your submission or one of your submissions, you must ensure that your new submission is named exactly the same as the previous in order for the system to overwrite it.**

You must ensure that the submitted assignment is all your own work and that all sources used are correctly attributed. Penalties apply to assignments which show evidence of academic unfair practice. (See the Student Handbook which is available on the A-Z key information on iLearn.)

ARDEN UNIVERSITY

Assessment Criteria (Learning objectives covered – all.

Level 5 reflects the continuing development in knowledge, understanding and skills from Level 4. At Level 5, students are not expected to be fully autonomous but are able to take responsibility for their own learning with appropriate guidance and direction. Students are expected to further develop their theoretical knowledge within a more intellectual context and to demonstrate this through more complex forms of expression which move beyond the descriptive or imitative domain. Students are expected to demonstrate skills of analysis in both problem-solving and resolution.

Grade	Mark Bands	Generic Assessment Criteria
First (1)	80%+	An outstanding information base exploring and analysing the discipline, its theory and any associated ethical considerations. There is sophisticated use and management of learning resources, and a high degree of autonomy is demonstrated. Writing is outstandingly well structured and accurately referenced throughout. Where appropriate, outstanding professional skills are demonstrated. The work is original and with some additional effort could considered for internal publication.
	70-79%	An excellent knowledge base within which the discipline is explored and analysed. There is a degree of originality in the approach. The work demonstrates confidence and autonomy and extends to consider ethical issues. Learning resources have been managed confidently. Writing is exceptionally well structured and accurately referenced throughout. Where appropriate, an excellent level of professional skills is demonstrated, and the work demonstrates a high level of intellectual and academic skills.
Upper second (2:1)	60-69%	A very good knowledge base which explores and analyses the discipline, its theory, and any associated ethical issues. There is evidence of some originality and independence of thought. A very good range of learning resources underpin the work and there is evidence of growing confidence and self-direction. The work demonstrates the ability to analyse the subject and apply theory with good academic and intellectual skills. Academic writing skills are very good, expression is accurate overall, and the work is consistently referenced throughout.
Lower second (2:2)	50-59%	A good understanding of the discipline which begins to analyse the subject and apply some underpinning theory. There may be reference to some of the ethical considerations. The work shows a sound level of competence in managing basic sources and materials. Academic writing skills are good and accurate overall, and the work is planned and structured with some though. Professional skills are good (where appropriate). The work lacks original thought, but academic and intellectual skills are moving into the critical domain. The work is referenced throughout.
Third (3)	40-49%	Satisfactory level of performance in which there are some omissions in understanding the subject, its underpinning theory, and ethical considerations. There is little evidence of independent thought, and the work shows a basic use of sources and materials. Academic and intellectual skills are limited. The work may lack structure overall. There are some difficulties in developing professional skills (where appropriate). There is an attempt to reference the work.
Marginal Fail	30-39%	A limited piece of work in which there are clear gaps in understanding the subject, its underpinning theory, and ethical considerations. The work shows a limited use of sources and materials. Academic and professional skills are weak and there are errors in expression and the work may lack structure overall. There are difficulties in developing professional skills (where appropriate). The work lacks original thought and is largely imitative.
Clear fail	29% and Below	A poor performance in which there are substantial gaps in knowledge and understanding, underpinning theory and ethical considerations. The work shows little evidence in the use of appropriate sources and materials. Academic writing skills are very weak and there are numerous errors in expression. The work lacks structure overall. Professional skills (where appropriate) are not developed. The work is imitative.

Marking Rubric

Criteria and weighting	Outstanding 80% - 100%	Excellent 70% - 79%	Very Good 60% - 69%	Good 50% - 59%	Pass 40% - 49%	Poor 30 – 39%	Fail 0 – 29%
Task 1 (20%)	An exemplary presentation and implementation using requested UML Class and object diagrams for complex systems e.g. given case-study etc and justified using a highly relevant literature base. Develop a diagram that demonstrates all the advanced relationships such as hierarchy, abstraction, composition that are evident in the case-study. The cardinality is mentioned correctly. The diagram should demonstrate , a high level of complexity, criticality, synthesis and original thought. An exemplary explanation that is free from errors is expected	An Excellent presentation and implementation using requested UML Class and object diagrams for complex systems e.g. given case-study etc and justified using a excellent relevant literature base. An excellent presentation that is free from errors. Advanced concepts such as relationships and cardinality between the classes are mentioned with minor mistakes.	A good presentation and implementation using requested UML Class and object diagrams for complex systems e.g. given case-study etc and justified using relevant literature base. Some of the advanced relationships might be missing or the cardinality might be missing. A wide-ranging use of relevant literature, though there are some minor issues. A very good presentation, which is clear and mostly free from errors.	A satisfactory presentation and implementation using requested Class and object UML diagrams for complex systems e.g. given case-study etc and justified using some relevant literature base. Advanced relationships and cardinality might be slightly wrong. A satisfactory presentation of academic as well as professional skills. Good use of relevant and valid literature appropriately referenced, with some scope for more depth and a good presentation though there are some issues that need to be addressed	Some relevant concepts are presented in terms of Unified Modelling Language (UML) for class and object diagrams. Advanced relationships such as hierarchy etc might not be demonstrated. Cardinality might be missing. Develop some diagrams, demonstrating some level of critical analysis. There is a lack of depth and relevance. Some inclusion of relevant sources with scope for more and a basic presentation which needs further development.	There is limited/no discussion relevant to Unified Modelling Language (UML). The student developed some/no diagrams which do not depict all the classes or there might be 2 -3 classes which might not be clearly described. No object or incorrect diagram. There are significant omissions or a significant lack of depth of content and discussion. There are significant issues with the presentation of the document and limited use of valid references	A very limited or wholly absent level of design.

Criteria and weighting	**Outstanding** **80% - 100%**	**Excellent** **70% - 79%**	**Very Good** **60% - 69%**	**Good** **50% - 59%**	**Pass** **40% - 49%**	**Poor** **30 – 39%**	**Fail** **0 – 29%**
Task 2 (20%)	An outstanding level of documentation, which addresses all aspects of the problem specification. Demonstrates a practitioner level of understanding of coding in Java. The execution of the program is evidenced via screenshots and the output as required by the task is obtained, proof of which is included in the document. A complete copy of the program in text is appended at the end of the document.	An excellent level of documentation, which addresses all aspects of the problem specification. Demonstrates an excellent level of understanding of coding in Java. Evidence of testing the code is included. Copy of the code in plain text included in the appendix.	A very good level of documentation, which addresses most aspects of the problem specification. Demonstrates an very good level of understanding of coding in Java. The code developed should not have any errors. Evidence of testing the code is included. Copy of code included in the appendix.	A good level of documentation and design, which addresses some aspects of the problem specification but there is scope for more depth and/or there are some errors or omissions but the code compiles. Demonstrates a good level of understanding of coding in Java. Evidence of testing the code is included.	A basic level of documentation, which addresses some aspects of the problem specification but there is scope for much more depth and/or there are a number of errors or omissions. Demonstrates a basic level of understanding of coding in Java but code executes without errors and evidence of testing is included.	An insufficient level of documentation and design, which addresses limited aspects of the problem specification. Demonstrates a insufficient level of understanding of coding, the code does not execute and there is no proof of testing included.	A very limited or wholly absent level of documentation and code with lots of errors or code which does not answer the task appropriately. There is no testing demonstrated.

Criteria and weighting	Outstanding 80% - 100%	Excellent 70% - 79%	Very Good 60% - 69%	Good 50% - 59%	Pass 40% - 49%	Poor 30 – 39%	Fail 0 – 29%
Task 3a (30%)	An outstanding level of documentation, which addresses all aspects of the problem specification. Demonstrates a practitioner level of understanding of coding in Java. The execution of the program is evidenced via screenshots and the output as required by the task is obtained, proof of which is included in the document. A complete copy of the program in text is appended at the end of the document.	An excellent level of documentation, which addresses all aspects of the problem specification. Demonstrates an excellent level of understanding of coding in Java. Evidence of testing the code is included. Copy of the code in plain text included in the appendix.	A very good level of documentation, which addresses most aspects of the problem specification. Demonstrates an very good level of understanding of coding in Java. The code developed should not have any errors. Evidence of testing the code is included. Copy of code included in the appendix.	A good level of documentation and design, which addresses some aspects of the problem specification but there is scope for more depth and/or there are some errors or omissions but the code compiles. Demonstrates a good level of understanding of coding in Java. Evidence of testing the code is included.	A basic level of documentation, which addresses some aspects of the problem specification but there is scope for much more depth and/or there are a number of errors or omissions. Demonstrates a basic level of understanding of coding in Java but code executes without errors and evidence of testing is included.	An insufficient level of documentation and design, which addresses limited aspects of the problem specification. Demonstrates a insufficient level of understanding of coding, the code does not execute and there is no proof of testing included.	A very limited or wholly absent level of documentation and code with lots of errors or code which does not answer the task appropriately. There is no testing demonstrated.

Criteria and weighting	**Outstanding** **80% - 100%**	**Excellent** **70% - 79%**	**Very Good** **60% - 69%**	**Good** **50% - 59%**	**Pass** **40% - 49%**	**Poor** **30 – 39%**	**Fail** **0 – 29%**
Task 3b (15%)	An outstanding level of documentation, which addresses all aspects of the problem specification. Demonstrates a practitioner level of understanding of coding in Java. The execution of the program is evidenced via screenshots and the output as required by the task is obtained, proof of which is included in the document. A complete copy of the program in text is appended at the end of the document.	An excellent level of documentation, which addresses all aspects of the problem specification. Demonstrates an excellent level of understanding of coding in Java. Evidence of testing the code is included. Copy of the code in plain text included in the appendix.	A very good level of documentation, which addresses most aspects of the problem specification. Demonstrates an very good level of understanding of coding in Java. The code developed should not have any errors. Evidence of testing the code is included. Copy of code included in the appendix.	A good level of documentation and design, which addresses some aspects of the problem specification but there is scope for more depth and/or there are some errors or omissions but the code compiles. Demonstrates a good level of understanding of coding in Java. Evidence of testing the code is included.	A basic level of documentation, which addresses some aspects of the problem specification but there is scope for much more depth and/or there are a number of errors or omissions. Demonstrates a basic level of understanding of coding in Java but code executes without errors and evidence of testing is included.	An insufficient level of documentation and design, which addresses limited aspects of the problem specification. Demonstrates a insufficient level of understanding of coding, the code does not execute and there is no proof of testing included.	A very limited or wholly absent level of documentation and code with lots of errors or code which does not answer the task appropriately. There is no testing demonstrated.

Criteria and weighting	**Outstanding** **80% - 100%**	**Excellent** **70% - 79%**	**Very Good** **60% - 69%**	**Good** **50% - 59%**	**Pass** **40% - 49%**	**Poor** **30 – 39%**	**Fail** **0 – 29%**
Task 3c (15%)	An outstanding level of documentation, which addresses all aspects of the problem specification. Demonstrates a practitioner level of understanding of coding in Java. The execution of the program is evidenced via screenshots and the output as required by the task is obtained, proof of which is included in the document. A complete copy of the program in text is appended at the end of the document.	An excellent level of documentation, which addresses all aspects of the problem specification. Demonstrates an excellent level of understanding of coding in Java. Evidence of testing the code is included. Copy of the code in plain text included in the appendix.	A very good level of documentation, which addresses most aspects of the problem specification. Demonstrates an very good level of understanding of coding in Java. The code developed should not have any errors. Evidence of testing the code is included. Copy of code included in the appendix.	A good level of documentation and design, which addresses some aspects of the problem specification but there is scope for more depth and/or there are some errors or omissions but the code compiles. Demonstrates a good level of understanding of coding in Java. Evidence of testing the code is included.	A basic level of documentation, which addresses some aspects of the problem specification but there is scope for much more depth and/or there are a number of errors or omissions. Demonstrates a basic level of understanding of coding in Java but code executes without errors and evidence of testing is included.	An insufficient level of documentation and design, which addresses limited aspects of the problem specification. Demonstrates a insufficient level of understanding of coding, the code does not execute and there is no proof of testing included.	A very limited or wholly absent level of documentation and code with lots of errors or code which does not answer the task appropriately. There is no testing demonstrated.