COM4011

Introduction to Programming

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 an **Introduction to Programming** assignment. Please refer to your Student Handbook for full details of the programme assessment scheme and general information on preparing and submitting assignments. The assignment brief will specifically give details and instructions for the assignment.

Module grade: Coursework 100%

Description: The assignment is given as three separate tasks. Each task is to be answered individually.

- Task 1 develops an understanding of data structures and programming techniques in the context of a programming language.
- Task 2 demonstrates an ability to write programs using appropriate structure and language rules.
- Task 3 demonstrates an understanding of how programs are developed i.e. from concept to development and testing.

Note: The evidence for this assessment needs to be provided in different ways – elaboration, screenshot, pictures, and java code in text.

Please ensure that you include your fully functional code in text in the appendix so it can be verified.

Learning outcomes:

After completing the module you should be able to:

1. **LO1** Develop an understanding of data structures and programming techniques in context of a programming language.
2. **LO2** Demonstrate an understanding of how programs are developed i.e. from concept to development and testing.
3. **LO3** Demonstrate an ability to write programs using appropriate structure and language rules.

Graduate attributes

4. Discipline Expertise: Knowledge and understanding of chosen field. Possess a range of skills to operate within this sector, have a keen awareness of current developments in working practice being well positioned to respond to change.

All learning outcomes must be met to pass the module.

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(s). 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: 3000 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:

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

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.***

Students who exceed the word count up to a 10% margin will not be penalised. Students should note that no marks will be assigned to work exceeding the specified limit once the maximum assessment size limit has been reached.

Scenario

A small retailer would like a Java program for managing their stock. Some items may be withdrawn from the stock while new items may be added periodically. The retailer wants a convenient way to store this information and manage it effectively. Any errors in the management of the stock should be avoided as they can create issues for sales.
During the working week, when items are sold, the stock is updated and reorders are placed daily to restock items that have fallen below their threshold. The value for the threshold indicated in Table 1 indicates the minimum current stock level needed below which a re-order is triggered. The reorder quantity indicated in Table 1 is used for each item being reordered.

Item#	Stock level	Unit cost	Unit price	Threshold	Reorder Qty
cd5751	21	2.45	4.59	12	30
bc4dd8	22	3.25	5.99	15	20
18b050	35	1.89	2.99	18	50
c47d16	6	9.75	13.79	2	5
6c1af3	14	5.78	8.79	10	10

Table 1. Stock table with typical items.

The retailer would like the programme to display the following menu:

Stock control menu
1. Enter stock data
2. Enter sales data
3. Exit

Figure 1. Menu to be displayed for the program.

Typical sales data for a day has been provided in Table 2.

Item	Qty sold
bc4dd8	8
18b050	20
c47d16	4
6c1af3	5

Table 2. Typical sales for the day.

Based on the stock from Table 1 and sales from Table 2, reorder list would be tabulated as illustrated:

#	Item	Current level	Threshold	Reorder Qty	Unit cost	Subtotal
1	bc4dd8	14	15	20	3.25	65.00
2	18b050	15	18	50	1.89	94.50
3	6c1af3	9	10	10	5.78	57.80
					Total cost	217.30

Table 3. Typical reorder list generated.

Assignment Task

Task 1

1) Provide a flowchart and pseudo-code to display the menu as in Figure 1. Once an option is selected and the code for the option executed, the menu should be displayed again. The program should terminate only when the option for exit is selected.
   a) Explain how you have validated user input (choice selection) and checked for any possible logical errors, while justifying the data types you have used.
   b) Flowchart and pseudocode should use standard conventions and format such as indentations, capitalisation, descriptive names for variables/methods, keywords, operators etc.
   c) The program should take the user input for menu options and execute the appropriate method. When option 1 is selected the method 'StockData' should be executed. When option 2 is selected the method 'SalesData' should be executed.
   d) At this stage each method should only have a print statement such as 'Enter current stock data for each item' or 'Enter sales completed for the day'. No actual choices for the items in stock or sales values are required. The complete Java methods for the selection of option 1 and option 2 will need to be written in task 2 and task 3 respectively.
2) Write a Java program based on the algorithm with comments to illustrate the program flow.
   a) The program should take the user input for menu options and execute the appropriate method.
3) Test your program by displaying the menu and selecting the various options when the menu is displayed. Provide screenshots for the output of the program showing the correct execution of all menu options.
4) Provide the Java code for this task in the Appendix **as text** for verification.

(30 marks)
(LOs: 1,2,3,4)

Task 2

In this task you will write the algorithm, code and test the working of the method 'StockData'. You must also create a suitable data structure to store the details of the stock items to be provided with their current stock among other details. This data structure should be accessible to read and write in both task 2 and task 3.

1) Provide a flowchart and pseudo-code for when option 1 is selected.
   a) The algorithm should demonstrate the acceptance of the user input for the items in stock. The algorithm should accept the data from Table 1, along with appropriate data **for an additional 5 items of your choice**.
   b) After entering all the stock data, the total cost of the stock should be displayed. e.g. for the stock data in Table 1, the message to be displayed would be 'The total cost of stock is 328.52'.
   c) Then the menu should be displayed for further user selection.
   d) Explain how you have validated user input (choice selection) and checked for any possible logical errors, while justifying the data types you have used.
   e) Flowchart and pseudocode should use standard conventions and format such as indentations, capitalisation, descriptive names for variables/methods, keywords, operators etc.
   f) The subroutine should take user input from the keyboard for the choices and data entry.
   g) The user input for items in stock should be stored using a suitable data structure.
2) Write a Java method 'StockData' based on the algorithm with comments to illustrate the program flow.
   a) The method should employ only fixed arrays where necessary – dynamic arrays (i.e. ArrayList, vector, HashMap, etc.) are not to be used.
3) You will need to enter all the stock data from Table 1, and an additional 5 new items with appropriate stock data values of your choice. **All this data should be entered using the keyboard as user input.**
4) Test your program by embedding the complete Java code for the 'StockData' method in the program written in task 1.
5) Provide screenshots for outputs of the Java method for the user input you have used.
6) Provide the Java code for this task in the Appendix **as text** for verification.

(30 marks)
(LOs: 1,2,3.4)

Task 3

In this task you will write the algorithm, code and test the working of the method 'SalesData'. You will need to use the data stored in task 2 to complete task 3.

1) Provide a flowchart and pseudo-code for when option 2 is selected.
   a) The algorithm should take the sales data from Table 2, with appropriate sales values for the additional 5 items you entered in task 2.
   b) Once the sales data is entered, the total sales amount should be displayed. e.g. for the data in Table 1 and Table 2 the message to be displayed would be 'The total sales amount is 206.83'. Then the reorder list should be displayed as in Table 3.
   c) Then the menu should be displayed for further user selection.
   d) Explain how you have validated user input (choice selection) and checked for any possible logical errors, while justifying the data types you have used.
   e) Flowchart and pseudocode should use standard conventions and format such as indentations, capitalisation, descriptive names for variables/methods, keywords, operators etc.
2) Write a Java method 'SalesData' based on the algorithm with comments to illustrate the program flow.
   a) The method should employ only fixed arrays where necessary – dynamic arrays (i.e. ArrayList, vector, HashMap, etc.) are not to be used.
3) You will need to enter all the sales data from Table 2, and appropriate sales values of your choice for the additional 5 items you entered in Task 2. **All this data should be entered using the keyboard as user input.**
4) Test your program by embedding the complete Java code for the 'SalesData' method in the program written in task 1.
5) Provide screenshots for outputs of the Java method.
6) Provide the full Java code in the Appendix **as text** for verification. At this stage the program should include the code for the menu display and the completed methods 'StockData' and 'SalesData'.

(40 marks)
(LOs: 1,2,3,4)

End of questions

As technology and platforms may change, your module tutor will provide you with up-to-date details.

Formative Feedback

You have the opportunity to submit a draft report to receive formative feedback. You are encouraged to submit your assignment for feedback **once** and it is 30% of your entire submission. You, the student, are to choose 30%, **not the tutor**. The last day for guaranteed feedback hand in is up to 2 weeks on Friday before submission week. No formative feedback will be given after the time specified above, either blended, or distance learning.

The Feedback is designed to help you develop areas of your work, encouraging academic skills and independent learning.

If you are a Distance Learning student, then you are encouraged to send 30% of your assignment for feedback by email to your tutor, no later than two weeks before your final submission week. Dates will be given to you by your tutor on a module-by-module basis.

Your work will be assessed according to the rubrics provided at the end of this brief.

Referencing Guidelines

You **MUST** underpin your analysis and evaluation of the key issues with appropriate and wide ranging academic research, ensuring all cited literature is referenced using the AU Harvard system(s).

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 should be submitted as a **single Word document with the Arden cover sheet** (MS Word with your student number as the file name). Do not zip/compress the file in any way. If you submit your work in any other format, the submission **will not be accepted and will be awarded 0 marks.**

The submission for each task should be presented under a relevant heading. Screenshots, algorithm, and inserts should be **legible**, and provided with suitable captions. Flowchart should be presented in an appropriate pictorial format using appropriate page size and orientation to make it legible.

The code should be clearly **presented in text** so that it can be copied and pasted into a suitable coding platform to be executed and verified. For more information, please see the “Submitting an Assignment - Guide” document available on the A-Z key information on iLearn.

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.)

Object-oriented paradigm (abstraction, inheritance, encapsulation and polymorphism) is outside the scope of the module and thus should not be used in completing this assessment. You should only use procedural paradigm when completing the assessment.

Assessment Criteria (Learning objectives covered – all.

Level 4 is the first stage on the student journey into undergraduate study. At Level 4 students will be developing their knowledge and understanding of the discipline and will be expected to demonstrate some of those skills and competences. Student are expected to express their ideas clearly and to structure and develop academic arguments in their work. Students will begin to apply the theory which underpins the subject and will start to explore how this relates to other areas of their learning and any ethical considerations as appropriate. Students will begin to develop self-awareness of their own academic and professional development.		
Grade	**Mark Bands**	**Generic Assessment Criteria**
First (1)	80%+	Outstanding performance which demonstrates the ability to analyse the subject area and to confidently apply theory whilst showing awareness of any relevant ethical considerations. The work shows an outstanding level of competence and confidence managing appropriate sources and materials, initiative and excellent academic writing skills and professional skills (where appropriate). The work shows originality of thought.
	70-79%	Excellent performance which demonstrates the ability to analyse the subject and apply theory whilst showing some awareness any relevant ethical considerations. The work shows a high level of competence in managing sources and materials, initiative ar excellent academic writing skills and professional skills (where appropriate). The work shows originality of thought.
Upper second (2:1)	60-69%	Very good performance which demonstrates the ability to analyse the subject and apply some theory. The work shows a very good level of competence in managing sources and materials and some initiative. Academic writing skill are very good, and expression remains accurate overall. Very good professional skills (where appropriate). The work shows son original thought.
Lower second (2:2)	50-59%	A good performance which begins to analyse the subject and apply some underpinning theory. The work shows a sound level competence in managing basic sources and materials. Academic writing skills are good, and expression remains accurate overa although the piece may lack structure. Good professional skills (where appropriate). The work lacks some original thought.
Third (3)	40-49%	Satisfactory level of performance in which there are some omissions in understanding the subject, its underpinning theory, and ethical considerations. The work shows a satisfactory use of sources and materials. Academic writing skills are limited and there are some errors in expression and the work may lack structure overall. There are some difficulties in developing professional ski (where appropriate). The work lacks original thought and is largely imitative.
Marginal Fail	30-39%	Limited performance in which there are omissions in understanding the subject, its underpinning theory, and ethical considerations. The work shows a limited use of sources and materials. Academic writing 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). T 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 developed. The work is imitative.

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 Algorithm (flowchart AND pseudo code) (40%) Max marks 12	Functional, comprehensive, **efficient** algorithm that is justified with all standard conventions that meets all requirements and **adds appropriate value** toward meeting the requirements.	Functional, **comprehensive** algorithm without logical errors and with error-proofing, justified, with all standard conventions that meets all the requirements.	Functional algorithm justified with all standard conventions and **with error-proofing** and without logical errors/flaws that meets all the requirements.	Functional algorithm **justified** with **all** standard conventions, **without logical errors** that meets **all** the requirements.	Functional algorithm **elaborated,** with **most** standard conventions that meet **most** of the minimum requirements.	**Functional** and/or **reverse-engineered** algorithm **presented** with **some** standard conventions that meet **some of the minimum** requirements.	**Non-functional** algorithm **provided** with **little or no standard conventions** that meets **few or none** of the task requirements. Significant **omission, or logical errors** in algorithm.
Task 1 Code in text (30%) Max marks 9	Effectively commented, succinct, and **efficient** code provided in text that **uses resources providently**, with error-proofing and without logical errors that implements the algorithm exactly and meets all the requirements	**Effectively** commented, **succinct** code provided in text with error-proofing, without logical errors, that implements the algorithm exactly and meets all requirements. Code makes **full** use of language features **productively**.	Fully commented code provided in text that implements the algorithm exactly **with error-proofing** and no logical errors and meets all the requirements. The code makes **some use of language features for productivity**.	**Fully** commented code provided in text that implements the algorithm **exactly** and meets all the requirements. The code has **little or no redundancy** and **no logical errors**.	**Mostly** commented fully functional code, in text, implements the algorithm **closely** and meets **most** of the minimum requirements. The code may have **some** redundancy.	**Functional** code with **some** comments**, in text,** implements the algorithm to **some extent,** meets **some of the minimum** requirements. There may be **significant redundancy** in code.	**Nonfunctional** Code provided **is without comments,** and/or **not in text**; has **errors** (syntax, logical), **does not** implement the algorithm and/or meets **few or none** of the requirements.
Task 1 Sample output screenshots (30%) Max marks 9	**Continuous** and comprehensive sample output illustrates the user interactivity and tests including fool-proofed inputs, while meeting all the requirements, and **providing testing quality assurance**.	**Comprehensive** sample output illustrates the user interactivity and tests invalid user inputs including error messages, while meeting all the requirements and without logical errors indicated through display prompts.	Sample output illustrates user interactivity and tests for all values **including invalid user inputs**, while meeting all requirements. All display prompts **including error messages** are appropriate, clear, unambiguous without logical errors.	Sample output illustrates user interactivity and tests **ALL** relevant values and meets **all** requirements. **All** display prompts included **appropriately**, are clear, unambiguous and **without logical errors**.	Sample output illustrates the user interactivity and tests for **most** relevant values, while meeting **most** of the minimum requirements. **Most** display prompts are included, are **clear** and/or **unambiguous**.	Sample output is **limited** to illustrate user interactivity, tests for **few** input values, while meeting **some** of the **minimum** requirements. **Some** display prompts are included and are **unclear** and/or **ambiguous**.	**Insufficient** sample output provided, which **does not** illustrate the user interactivity or functionality of code**. Limited or no** display prompts in the sample output.

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 Algorithm (flowchart AND pseudo code) (40%) Max marks 12	Functional, comprehensive, **efficient** algorithm that is justified with all standard conventions that meets all requirements and **adds appropriate value** toward meeting the requirements.	Functional, **comprehensive** algorithm without logical errors and with error-proofing, justified, with all standard conventions that meets all the requirements.	Functional algorithm justified with all standard conventions and **with error-proofing** and without logical errors/flaws that meets all the requirements.	Functional algorithm **justified** with **all** standard conventions, **without logical errors** that meets **all** the requirements.	Functional algorithm **elaborated,** with **most** standard conventions that meet **most** of the minimum requirements.	**Functional** and/or **reverse-engineered** algorithm **presented** with **some** standard conventions that meet **some of the minimum** requirements.	**Non-functional** algorithm **provided** with **little or no standard conventions** that meets **few or none** of the task requirements. Significant **omission, or logical errors** in algorithm.
Task 2 Code in text (30%) Max marks 9	Effectively commented, succinct, and **efficient** code provided in text that **uses resources providently**, with error-proofing and without logical errors that implements the algorithm exactly and meets all the requirements	**Effectively** commented, **succinct** code provided in text with error-proofing, without logical errors, that implements the algorithm exactly and meets all requirements. Code makes **full** use of language features **productively**.	Fully commented code provided in text that implements the algorithm exactly **with error-proofing** and no logical errors and meets all the requirements. The code makes **some use of language features for productivity**.	**Fully** commented code provided in text that implements the algorithm **exactly** and meets all the requirements. The code has **little or no redundancy** and **no logical errors**.	**Mostly** commented fully functional code, in text, implements the algorithm **closely** and meets **most** of the minimum requirements. The code may have **some** redundancy.	**Functional** code with **some** comments**, in text,** implements the algorithm to **some extent,** meets **some of the minimum** requirements. There may be **significant redundancy** in code.	**Nonfunctional** Code provided **is without comments,** and/or **not in text**; has **errors** (syntax, logical), **does not** implement the algorithm and/or meets **few or none** of the requirements.
Task 2 Sample output screenshots (30%) Max marks 9	**Continuous** and comprehensive sample output illustrates the user interactivity and tests including fool-proofed inputs, while meeting all the requirements, and **providing testing quality assurance**.	**Comprehensive** sample output illustrates the user interactivity and tests invalid user inputs including error messages, while meeting all the requirements and without logical errors indicated through display prompts.	Sample output illustrates user interactivity and tests for all values **including invalid user inputs**, while meeting all requirements. All display prompts **including error messages** are appropriate, clear, unambiguous without logical errors.	Sample output illustrates user interactivity and tests **ALL** relevant values and meets **all** requirements. **All** display prompts included **appropriately**, are clear, unambiguous and **without logical errors**.	Sample output illustrates the user interactivity and tests for **most** relevant values, while meeting **most** of the minimum requirements. **Most** display prompts are included, are **clear** and/or **unambiguous**.	Sample output is **limited** to illustrate user interactivity, tests for **few** input values, while meeting **some** of the **minimum** requirements. **Some** display prompts are included and are **unclear** and/or **ambiguous**.	**Insufficient** sample output provided, which **does not** illustrate the user interactivity or functionality of code**. Limited or no** display prompts in the sample output.

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 3 Algorithm (flowchart AND pseudo code) (40%) Max marks 16	Functional, comprehensive, **efficient** algorithm that is justified with all standard conventions that meets all requirements and **adds appropriate value** toward meeting the requirements.	Functional, **comprehensive** algorithm without logical errors and with error-proofing, justified, with all standard conventions that meets all the requirements.	Functional algorithm justified with all standard conventions and **with error-proofing** and without logical errors/flaws that meets all the requirements.	Functional algorithm **justified** with **all** standard conventions, **without logical errors** that meets **all** the requirements.	Functional algorithm **elaborated,** with **most** standard conventions that meet **most** of the minimum requirements.	**Functional** and/or **reverse-engineered** algorithm **presented** with **some** standard conventions that meet **some of the minimum** requirements.	**Non-functional** algorithm **provided** with **little or no standard conventions** that meets **few or none** of the task requirements. Significant **omission, or logical errors** in algorithm.
Task 3 Code in text (30%) Max marks 12	Effectively commented, succinct, and **efficient** code provided in text that **uses resources providently**, with error-proofing and without logical errors that implements the algorithm exactly and meets all the requirements	**Effectively** commented, **succinct** code provided in text with error-proofing, without logical errors, that implements the algorithm exactly and meets all requirements. Code makes **full** use of language features **productively**.	Fully commented code provided in text that implements the algorithm exactly **with error-proofing** and no logical errors and meets all the requirements. The code makes **some use of language features for productivity**.	**Fully** commented code provided in text that implements the algorithm **exactly** and meets all the requirements. The code has **little or no redundancy** and **no logical errors**.	**Mostly** commented fully functional code, in text, implements the algorithm **closely** and meets **most** of the minimum requirements. The code may have **some** redundancy.	**Functional** code with **some** comments**, in text,** implements the algorithm to **some extent,** meets **some of the minimum** requirements. There may be **significant redundancy** in code.	**Nonfunctional** Code provided **is without comments,** and/or **not in text**; has **errors** (syntax, logical), **does not** implement the algorithm and/or meets **few or none** of the requirements.
Task 3 Sample output screenshots (30%) Max marks 12	**Continuous** and comprehensive sample output illustrates the user interactivity and tests including fool-proofed inputs, while meeting all the requirements, and **providing testing quality assurance**.	**Comprehensive** sample output illustrates the user interactivity and tests invalid user inputs including error messages, while meeting all the requirements and without logical errors indicated through display prompts.	Sample output illustrates user interactivity and tests for all values **including invalid user inputs**, while meeting all requirements. All display prompts **including error messages** are appropriate, clear, unambiguous without logical errors.	Sample output illustrates user interactivity and tests **ALL** relevant values and meets **all** requirements. **All** display prompts included **appropriately**, are clear, unambiguous and **without logical errors**.	Sample output illustrates the user interactivity and tests for **most** relevant values, while meeting **most** of the minimum requirements. **Most** display prompts are included, are **clear** and/or **unambiguous**.	Sample output is **limited** to illustrate user interactivity, tests for **few** input values, while meeting **some** of the **minimum** requirements. **Some** display prompts are included and are **unclear** and/or **ambiguous**.	**Insufficient** sample output provided, which **does not** illustrate the user interactivity or functionality of code**. Limited or no** display prompts in the sample output.