NAME DATE STARTED TARGET END DATE

GCE A LEVEL — COMPUTER SCIENCE UNIT 3 QUESTION PACK

1500U30-1 · 2015 spec Unit 3 Topic 4 · A2 unit, first sat 2017, 100 marks, 2h paper

REVISE.wales

COMPUTER SCIENCE — UNIT 3 · Programming Paradigms — OOP, Functional & Declarative

Topic 3.4 — Procedural, object-oriented, event-driven, functional and logic-programming paradigms

Explaining what a programming paradigm is, defining objects, classes and methods in OOP, distinguishing procedural from object-oriented and event-driven approaches, and comparing functional and logic programming as declarative paradigms with example languages.

2015 specification · current

Estimated time for entire question pack: ~1 h 4 min

Derived from the Unit 3 pace of ~1.5 min/mark, padded for written-prose answers (43 marks over 6 questions).

*You are advised to **not** attempt to complete all of this in one sitting.*

ABOUT THIS QUESTION PACK

This is a **comprehensive topic question pack**, not a single mock paper. It contains every question from the WJEC A2 Unit 3 papers (Summer 2017 – Summer 2024, COVID gap) that maps onto Topic 3.4 of the 2015 specification.

Questions are ordered by source paper date.

INSTRUCTIONS

Use black ink or black ball-point pen. Show all working. A calculator is allowed where useful.

For Examiner's use only

Q	Source	Max	Mark
1	S17 Q2	3	
2	S18 Q11	10	
3	S19 Q7	10	
4	S19 Q11	6	
5	S22 Q8	6	
6	S23 Q2	8	
Total		**43**	

Programming Paradigms — OOP, Functional & Declarative — what the spec asks

WJEC GCE A Level Computer Science (from 2015) · Unit 3: Programming & System Development · Topic 3.4.

What is a programming paradigm?

- A style or model of programming — a way of structuring code and reasoning about computation.
- Different paradigms suit different problems: procedural for scripts, OOP for large systems, declarative for queries.
- Some languages support multiple paradigms (Python, C#).
- Paradigm influences how you decompose the problem.

Procedural programming

- Sequence of procedures (subroutines) operating on shared data.
- Top-down design: decompose problem into smaller procedures.
- Examples: C, Pascal, original BASIC.
- Easy for small programs; can become hard to maintain as code grows.

Object-oriented programming (OOP)

- Code organised into objects that combine state (attributes) and behaviour (methods).
- Class = blueprint; object = instance of a class.
- Encapsulation hides internal state behind methods.
- Inheritance allows subclasses to reuse and extend parent classes; polymorphism enables substitutability.

Event-driven programming

- Code (event handlers) executes in response to events: clicks, key presses, sensor inputs, timers.
- Main program is a loop that dispatches events to handlers.
- Ideal for GUIs, games and real-time systems.
- Examples: JavaScript, Visual Basic, modern UI frameworks.

Functional programming

- Treats computation as evaluation of mathematical functions; no side effects.
- Functions are first-class — pass them as arguments, return them from other functions.
- Emphasises immutability, recursion, map/filter/reduce.
- Examples: Haskell, Lisp, F#, parts of Python and JavaScript.

Logic / declarative programming

- You declare facts and rules; the runtime works out how to satisfy queries.
- Programmer states *what* the result should look like, not *how* to compute it.
- Used in AI, expert systems, automated theorem proving.
- Example: Prolog. SQL is also declarative.

Crib sheet »

Programming Paradigms — OOP, Functional & Declarative in one page

Quick-reference notes — revisit before each question.

Paradigm one-liners

Procedural: step-by-step procedures.
OOP: objects with state + behaviour.
Event-driven: handlers react to events.
Functional: composition of pure functions.
Logic: declare facts/rules, query.

Class & object

Class = blueprint (e.g. Car).
Object = instance (e.g. myCar = new Car()).
Many objects can be instantiated from one class.
Each object has its own attribute values.

Method

Function bound to a class — operates on the object's state.
Called via the object: `myCar.start()`.
Hides implementation; provides behaviour interface.

OOP pillars

Encapsulation: hide state, expose methods.
Inheritance: subclass extends superclass.
Polymorphism: same call, different behaviour.
Abstraction: model only what matters.

Functional vs logic

Functional: pure functions, immutability, recursion. E.g. Haskell, Lisp.
Logic: facts + rules + queries; engine searches. E.g. Prolog.
Both declarative — you say *what*, not *how*.

Choosing a paradigm

GUI / games ⇒ event-driven.
Large system with many entities ⇒ OOP.
Short script ⇒ procedural.
Database query ⇒ declarative (SQL).
AI / theorem proving ⇒ logic.

Answer **all** *questions.*

1. Two human computer interfaces (HCI) are voice input and touch screen.

(a) Give **two** benefits of using a touch screen interface on a mobile device. [2]

(b) Describe the difficulties in creating a natural language interface for voice input. [4]

2. Explain the terms object, class and method in object-oriented programming. [3]

3. *(a)* Draw a truth table to show the value of **P** for all possible values of **A**, **B** and **C** for the following Boolean expression:

$$P = \bar{A} \cdot B + C$$ [4]

(b) Using the data in the 8 bit register below, design a mask and use it to demonstrate how a logical operation can be used to extract the least significant 4 bits. [3]

Bit number	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Register contents	1	1	1	0	0	0	1	1

4. *(a)* Describe the processes carried out during the lexical, syntax and semantic analyses stages of compilation. [6]

(b) Describe **one** advantage of using a programming language that requires compiling compared with a programming language that requires interpreting. [2]

(c) Describe **two** advantages of using a programming language that requires interpreting compared with a programming language that requires compiling. [4]

(d) State the purpose of an assembler. Describe the difference between the source code of an assembler and the source code of a compiler. [4]

10. *(a)* Describe the purpose of data validation. [2]

(b) Write an algorithm to validate a date in dd/mm/yyyy format. [11]

11. *(a)* Describe what is meant by Object-Oriented Programming (OOP). [4]

(b) Explain the relationship between classes and instances. [3]

(c) Explain what is meant by a method. [3]

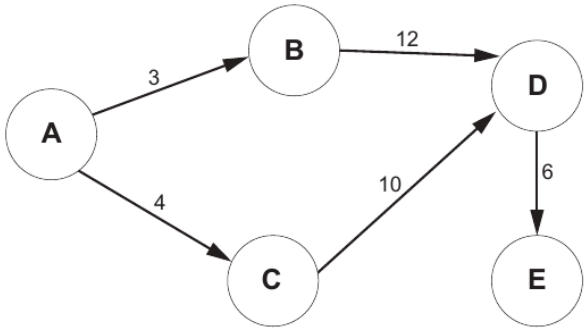
12. Explain the need for standardisation of computer languages and discuss the advantages arising from the use of algorithms and programming languages that have been standardised.

You should draw on your knowledge, skills and understanding from a number of areas across your Computer Science course when answering this question. [10]

END OF PAPER

6. *(a)* Explain the purpose of a shortest path algorithm. [4]

(b) This is a diagram of the costs of traversing a network.



The traversal cost for each node is 2.

(i) Show how this network and its traversal costs can be represented using a two dimensional array. [2]

(ii) State the shortest path from node A to E and calculate its cost. [2]

7. *(a)* Explain what is meant by the term programming paradigm. [2]

(b) Describe the difference between procedural and event-driven programming paradigms. [4]

(c) Using examples, describe which programming paradigms would be most suitable when developing different types of software applications. [4]

8. Draw a truth table to prove the following: [4]

A NOR B = NOT A AND NOT B

9. Below are two algorithms that search for a data item in a one dimensional array. You can assume that the data in the array is in ascending order and that the data item being searched is present. The Search_A algorithm has a time performance of O(n).

```
Algorithm Search_A

declare searchKey, i as integer
declare flag as Boolean
declare myArray[] as integer[]
Input searchKey
Set i = 0
flag = FALSE

repeat
      if myArray[i] = searchKey then
            set flag = TRUE
      end if
      set i = i + 1
until (flag = TRUE)
output myArray[i]
```

```
Algorithm Search_B

declare searchKey, first, last, m as integer
declare myArray[] as integer[]
Input searchKey
Set first = 1
Set last = len(myArray[])

repeat
      set m = (first + last) DIV 2
      if searchKey < myArray[m] then
            set last = m - 1
      else
            set first = m + 1
      end if
until (myArray[m] = searchKey)
output myArray[m]
```

(a) Evaluate the efficiency of the Search_B algorithm and using Big O notation, determine the growth rate for time performance. [5]

(b) Draw a graph of the algorithms above to illustrate their order of time performance. Graph paper is not required. [4]

(c) State which algorithm is more efficient when searching for a data item. [1]

10. Describe the term data compression and explain how data compression algorithms are used.[6]

11. *(a)* Describe what is meant by a class and an object, and describe the relationship between them. [4]

(b) Describe what is meant by the term method, and describe the relationship between object and method. [2]

6. (a) Describe the approach to analysis and design in the following software development methodologies:

(i) Waterfall [4]

(ii) Agile [4]

(b) (i) Describe **one** piece of documentation that should be produced during the analysis stage of software development. [2]

(ii) Describe **one** piece of documentation that should be used during the maintenance stage of software development [2]

7. Every book contains a unique 13-digit International Standard Book Number (ISBN). An ISBN comprises five parts: a GS1 assigned prefix, registration group, publisher, title and a check digit. Each individual part is separated with a space or hyphen.

- The GS1 assigned prefix must be 978 or 979.
- The registration group must be a number between 01 and 99
- The publisher must be a number between 00001 and 99999.
- The title must be a number between 01 and 99.
- The check digit must be a single digit.
- A separator of each part which can be either a space (' ') or hyphen ('-').

Example: 978-11-08412-72-8
Produce a Backus-Naur Form (BNF) definition for a 13-digit ISBN. [6]

8. Functional programming and logic programming are both declarative programming paradigms. Explain these two paradigms, giving an example language in each case:

(a) Functional programming [3]

(b) Logic programming [3]

9. All computer languages should follow the same standards.

(a) Explain the need for standardisation of computer languages. [2]

(b) Describe two potential difficulties involved in agreeing these standards. [2]

10. Draw a truth table to prove the following:

B AND NOT (A NOR B) = B [4]

Answer **all** questions.

1. (a) Explain the operation of a hash table. [4]

(b) This is a diagram of a hash table:

Key	Value
1001	Apple
1002	Berry
1003	Kiwi
1004	Lime
1005	Mango
1006	Pear
1007	Pineapple

(i) Redraw the hash table after "1008, Orange" has been added and "1003, Kiwi" has been deleted. [2]

(ii) Starting from the modified hash table resulting from (b)(i) redraw the hash table after "1006, Peach" has been added. [2]

2. Explain the differences between procedural and object orientated programming paradigms, giving suitable examples for each. [8]

3. Using the following Truth Table express *p* as a Boolean expression:

A	*B*	*C*	*p*
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

[4]

END OF QUESTION PACK

6 questions · 43 marks · ~1 h 4 min
Source: WJEC A2 Computer Science Unit 3 (1500U30-1), Summer 2017–2024, COVID gap
Curated for WJEC Computer Science 2015 spec A2 Unit 3 — Topic 4 (3.4)