



**AS
COMPUTER SCIENCE**

Paper 1

7516/1

Monday 4 June 2018

Morning

Time allowed: 1 hour 45 minutes

For this paper you must have:

- a computer
- a printer
- appropriate software
- the Electronic Answer Document
- an electronic version and a hard copy of the Skeleton Program
- an electronic version of the Data File
- an electronic version and a hard copy of the Preliminary Material.

You must NOT use a calculator.

[Turn over]

INSTRUCTIONS

- **Type the information required on the front of your Electronic Answer Document.**
- **Before the start of the examination make sure your CENTRE NUMBER, CANDIDATE NAME and CANDIDATE NUMBER are shown clearly IN THE FOOTER of every page (not the front cover) of your Electronic Answer Document.**
- **Enter your answers into the Electronic Answer Document.**
- **Answer ALL questions.**
- **Save your work at regular intervals.**

INFORMATION

- **The marks for questions are shown in brackets.**
- **The maximum mark for this paper is 75.**
- **No extra time is allowed for printing and collating.**
- **The question paper is divided into THREE sections.**

ADVICE

You are advised to allocate time to each section as follows:

SECTION A – 20 minutes;

SECTION B – 20 minutes;

SECTION C – 65 minutes.

AT THE END OF THE EXAMINATION

Tie together all your printed Electronic Answer Document pages and hand them to the Invigilator.

WARNING

It may not be possible to issue a result for this paper if your details are not on every page of your Electronic Answer Document.

DO NOT TURN OVER UNTIL TOLD TO DO SO

SECTION A

You are advised to spend no more than 20 MINUTES on this section.

Enter your answers to SECTION A in your Electronic Answer Document. You MUST SAVE this document at regular intervals.

QUESTION 3 in this section asks you to write program code starting from a NEW program/project/file.

You are advised to save your program at regular intervals.

0	1
---	---

TABLE 1 lists some terms that are important in the theory of computation. These terms are described in TABLE 2, on page 5.

TABLE 1

LABEL	TERM
A	Information hiding
B	Procedural abstraction
C	Functional abstraction
D	Data abstraction
E	Problem abstraction
F	Decomposition
G	Composition
H	Automation

Complete TABLE 2 by filling in the unshaded cells with the appropriate labels (A) to (H) from TABLE 1. On each row write the label that most closely matches the description.

TABLE 2

DESCRIPTION OF TERM	LABEL: A – H
Breaking a problem into a number of sub-problems	
Models are put into action to solve problems	
Combining procedures into compound procedures	
Details are removed until the problem is represented in a way that is possible to solve because the problem reduces to one that has already been solved	

Copy the contents of all the unshaded cells in TABLE 2 into your Electronic Answer Document. [4 marks]

[Turn over]

0	2
---	---

The algorithm, represented using pseudo-code in **FIGURE 1**, describes a method to test whether an integer greater than 2 is prime or not.

FIGURE 1

```
INPUT Number
Root ← 1
WHILE (Root * Root) < Number
    Root ← Root + 1
ENDWHILE
d ← 2
FactorFound ← FALSE
WHILE (FactorFound = FALSE) AND (d <= Root)
    r ← Number MOD d
    IF r = 0 THEN
        FactorFound ← TRUE
    ENDIF
    d ← d + 1
ENDWHILE
IF FactorFound = FALSE THEN
    OUTPUT "Prime"
ELSE
    OUTPUT "Not prime"
ENDIF
```

The MOD operator calculates the remainder resulting from an integer division, for example, $10 \text{ MOD } 3 = 1$.

- 02.1** Complete TABLE 3 by hand-tracing the algorithm in FIGURE 1. Use 5 as the input value. You may not need to use all the rows in TABLE 3.

Copy the contents of all the unshaded cells in TABLE 3 into your Electronic Answer Document.
[3 marks]

TABLE 3

Number	Root	d	Factor Found	r	Output

[Turn over]

BLANK PAGE

- 02.2** Complete TABLE 4 by hand-tracing the algorithm in FIGURE 1. Use 25 as the input value. You may not need to use all the rows in TABLE 4.

Copy the contents of all the unshaded cells in TABLE 4 into your Electronic Answer Document. [3 marks]

TABLE 4

Number	Root	d	Factor Found	r	Output

[Turn over]

0	3
---	---

The algorithm, represented using pseudo-code in FIGURE 2, outputs a series of numbers. The contents of the series depends on the initial value entered by the user.

FIGURE 2

```
Number ← 0
WHILE (Number < 1) OR (Number > 10)
  OUTPUT "Enter a positive whole number: "
  INPUT Number
  IF Number > 10 THEN
    OUTPUT "Number too large."
  ELSE
    IF Number < 1 THEN
      OUTPUT "Not a positive number."
    ENDIF
  ENDIF
ENDWHILE
c ← 1
FOR k ← 0 TO Number - 1
  OUTPUT c
  c ← (c * (Number - 1 - k)) DIV (k + 1)
ENDFOR
```

The DIV operator calculates the result of an integer division, for example, $10 \text{ DIV } 3 = 3$.

WHAT YOU NEED TO DO:**TASK 1**

Write a program to implement the algorithm in **FIGURE 2**.

TASK 2

Test the program by showing the result of entering:

-3

then 11

then 10

EVIDENCE THAT YOU NEED TO PROVIDE

Include the following evidence in your Electronic Answer Document.

03. **1** Your **PROGRAM SOURCE CODE** for **TASK 1**.
[9 marks]

03. **2** **SCREEN CAPTURE(S)** showing the test
described in **TASK 2**. [1 mark]

[Turn over]

SECTION B

You are advised to spend no more than 20 MINUTES on this section.

Enter your answers to SECTION B in your Electronic Answer Document. You MUST SAVE this document at regular intervals.

These questions refer to the PRELIMINARY MATERIAL and the SKELETON PROGRAM, but do NOT require any additional programming.

Refer EITHER to the PRELIMINARY MATERIAL issued with this question paper OR your electronic copy.

0 4 State the name of an identifier for:

0 4 . **1** a variable that is used to store a Boolean value. [1 mark]

0 4 . **2** a user-defined subroutine that returns a single character. [1 mark]

0 5

The program uses different variables to store single characters or the empty string. The variable identifiers are:

`FirstSignal`, `MenuOption`, `PlainTextLetter`, `Signal` **and** `Symbol`.

Using the variable identifiers listed above, complete TABLE 5 by filling in the unshaded cells with the correct variable identifier.

TABLE 5

VARIABLE IDENTIFIER	DESCRIPTION
	uncoded letter, part of <code>PlainText</code>
	single unit of <code>Transmission</code> (= or <code>SPACE</code> or <code>EOL</code>)
	first character in <code>Transmission</code>
	used to build <code>SymbolString</code> (. or -)

Copy the contents of all the unshaded cells in TABLE 5 into your Electronic Answer Document. [4 marks]

[Turn over]

0 6

Several subroutines use a variable

`TransmissionLength`. It could have been declared as a global variable instead of locally in each subroutine.

Why is it good practice to use local variables?
[1 mark]

0 7

This question refers to the subroutine

`ReceiveMorseCode`.

0 7 . 1

What is the purpose of the variable `i` in this subroutine? [2 marks]

0 7 . 2

Explain what happens when the

`GetTransmission` subroutine is called by the `ReceiveMorseCode` subroutine and the file to be read contains only spaces. You must include in your explanation how the subroutine `ReceiveMorseCode` deals with this situation.
[5 marks]

0 8

This question refers to the subroutine

`GetNextSymbol`.

Give an example of a transmission string that would generate the error message

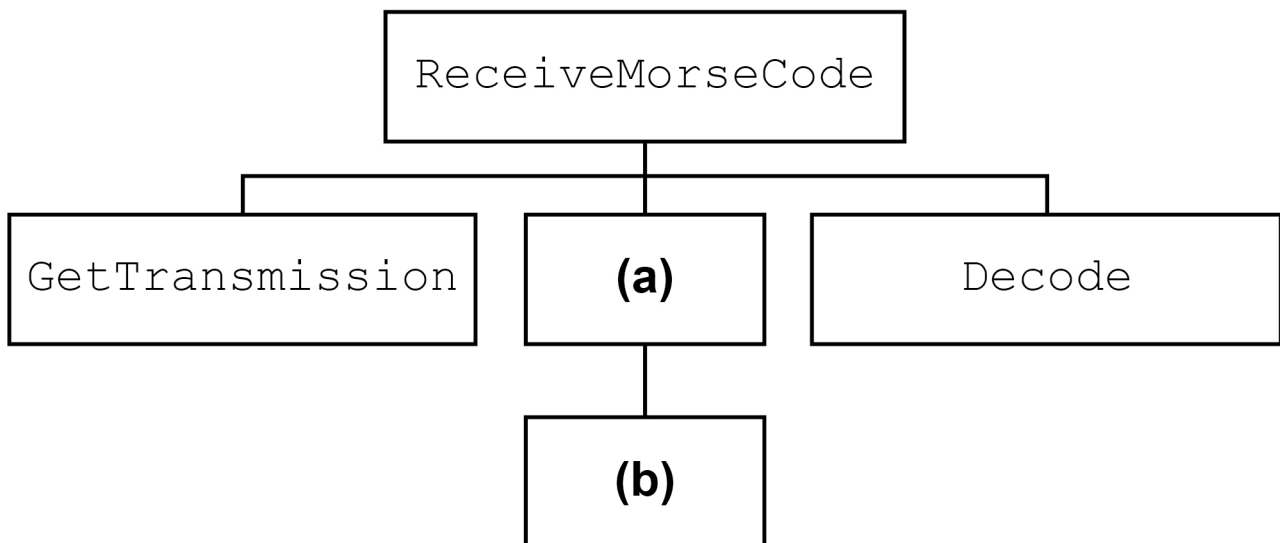
`"Non-standard symbol received"` and explain how the code will detect the error caused by your string. [3 marks]

0 9

FIGURE 3 shows an incomplete hierarchy chart for part of the **SKELETON PROGRAM**.

With reference to the **SKELETON PROGRAM** and **FIGURE 3**, answer questions 9.1 to 9.2.

FIGURE 3



0 9 . 1 What should be written in box (a) in **FIGURE 3**?
[1 mark]

0 9 . 2 What should be written in box (b) in **FIGURE 3**?
[1 mark]

[Turn over]

1	0
---	---

Morse codes also exist for numerals, so that a text message including numbers can also be transmitted. For example, the Morse code for 3 is:

...--

The program is to be extended to include the Morse code for all numerals from 0 to 9.

Describe the changes that would need to be made to the **SKELETON PROGRAM** to achieve this.

In your answer you should discuss the changes that need to be made to the data structures and subroutines.

You are **NOT** expected to actually make the changes. [6 marks]

BLANK PAGE

[Turn over]

SECTION C

You are advised to spend no more than 65 MINUTES on this section.

Enter your answers to SECTION C in your Electronic Answer Document. You MUST SAVE this document at regular intervals.

These questions require you to load the SKELETON PROGRAM and to make programming changes to it.

1	1
---	---

This question refers to the subroutine `SendMorseCode`.

WHAT YOU NEED TO DO:

TASK 1

Modify the subroutine `SendMorseCode` so that it checks each character of the input string entered by the user.

If a character is not an uppercase letter or a space then the program should use the subroutine `ReportError` to output a suitable error message.

The subroutine should then set `MorseCodeString` to the empty string and exit.

TASK 2

Test that the changes you have made work by conducting the following test:

- run the program
- choose option `S`
- enter the string `Help`

EVIDENCE THAT YOU NEED TO PROVIDE

Include the following evidence in your Electronic Answer Document.

1 1 . 1 Your amended **PROGRAM SOURCE CODE** for the subroutine `SendMorseCode`. **[4 marks]**

1 1 . 2 **SCREEN CAPTURE(S)** showing the requested test. **[1 mark]**

[Turn over]

1 2

This question will extend the functionality of the program. To send a Morse code message, the dots and dashes need to be converted into signals.

WHAT YOU NEED TO DO:

TASK 1

Create a new subroutine `SendSignals` that converts a Morse code message such as

`- . . - . . - . . -`

(TEA X) to its equivalent signal

`=== . . = . . = = = = = . = . = = =`

by using the rules

- a dot is converted to an equals sign (=) followed by a space
- a dash is converted to three equals signs (===) followed by a space
- a space is converted to two spaces.
(This results in three spaces after the signals for a letter.)

Note that `△` represents a space and should be displayed as a space on screen.

The subroutine is to output the complete transmission string to the screen.

Add the call `SendSignals (MorseCodeString)` **as the last statement to be executed in the subroutine** `SendMorseCode`.

TASK 2

Test that the changes you have made work by conducting the following test:

- **run the program**
- **choose option** `S`
- **enter the string** `MORSE X`

EVIDENCE THAT YOU NEED TO PROVIDE

Include the following evidence in your Electronic Answer Document.

1 2 . 1 Your **PROGRAM SOURCE CODE** for the subroutine `SendSignals`. [7 marks]

1 2 . 2 **SCREEN CAPTURE(S)** showing the Morse code message and the signals. [1 mark]

[Turn over]

1 3

This question will extend the program to output the letters of the alphabet and their corresponding Morse codes.

WHAT YOU NEED TO DO:

TASK 1

Create a new subroutine

`OutputAlphabetWithCode`. This subroutine is to display each uppercase letter of the alphabet and its corresponding Morse code. The output is to be created using the data in `Letter` and `MorseCode`.

The format of the output should be as shown in FIGURE 4.

- Each uppercase letter is followed by a space and then its Morse code, grouped four to a line.
- Each Morse code should be padded with trailing spaces to a width of 6 characters.

FIGURE 4

A	. -	B	- . . .	C	- . - .	D	- . .
E	.	F	. . - .	G	- - .	H
I	. .	J	. - - -	K	- . -	L	. - . .
M	- -	N	- .	O	- - -	P	. - - .
Q	- - . -	R	. - .	S	. . .	T	-
U	. . -	V	. . . -	W	. - -	X	- . . -
Y	- . - -	Z	- - . .				

TASK 2

Amend the subroutines `DisplayMenu` and `SendReceiveMessages` so that the user can choose a new option `A` that will call `OutputAlphabetWithCode`.

TASK 3

- run the program
- choose new option `A`

EVIDENCE THAT YOU NEED TO PROVIDE

Include the following evidence in your Electronic Answer Document.

13 . 1 Your **PROGRAM SOURCE CODE** for the subroutine `OutputAlphabetWithCode`. Your amended **PROGRAM SOURCE CODE** for the subroutines `DisplayMenu` and `SendReceiveMessages`. **[6 marks]**

13 . 2 **SCREEN CAPTURE(S)** showing the main menu and the output from the test run. **[1 mark]**

[Turn over]

1	4
---	---

This question will further extend the functionality of the program.

Before sending a message in Morse code the message is to be encrypted using a Caesar cipher. To encrypt a character (letter or space) it is shifted a given number of places (the key) in the alphabet (space is taken as the first character, then A, B, C and so on to Z). If a character is required before space or after Z then wrap-round is to be implemented.

For example:

- if the key is 2, then SPACE is encrypted as B, A is encrypted as C, B is encrypted as D, ..., X is encrypted as Z, Y is encrypted as SPACE, and Z is encrypted as A.
- if the key is -1, then SPACE is encrypted as Z, A is encrypted as SPACE, B is encrypted as A, ..., and Z is encrypted as Y.

Three different keys are to be used depending on the position of the character in the message.

The first character in the message is to be encrypted using the first key, the second character is encrypted with the second key and the third character with the third key.

The fourth character is then encrypted using the first key again, the fifth character is encrypted with the second key and the sixth character with the third key and so on.

The three keys are repeatedly used like this until the end of the message.

WHAT YOU NEED TO DO:

TASK 1

At the start of the subroutine

`SendReceiveMessages` **add code to enter three values that MUST be integers (the keys).**

TASK 2

Amend the subroutine `SendMorseCode` to encrypt the message using the substitution method described above and the keys entered by the user.

TASK 3

Test that your code works by conducting the following test:

- run the program
- for the first key enter 17
- for the second key enter 5
- for the third key enter -3
- enter S
- enter the string TEA X

[Turn over]

EVIDENCE THAT YOU NEED TO PROVIDE

Include the following evidence in your Electronic Answer Document.

1 4 . 1 Your amended **PROGRAM SOURCE CODE** for the subroutines `SendMorseCode` and `SendReceiveMessages`. **[9 marks]**

1 4 . 2 **SCREEN CAPTURE(S)** showing the keys entered, the plain text and the encrypted Morse code message. **[1 mark]**

END OF QUESTIONS

There are no questions printed on this page

There are no questions printed on this page

Copyright Information

For confidentiality purposes, from the November 2015 examination series, acknowledgements of third party copyright material will be published in a separate booklet rather than including them on the examination paper or support materials. This booklet is published after each examination series and is available for free download from www.aqa.org.uk after the live examination series.

Permission to reproduce all copyright material has been applied for. In some cases, efforts to contact copyright-holders may have been unsuccessful and AQA will be happy to rectify any omissions of acknowledgements. If you have any queries please contact the Copyright Team, AQA, Stag Hill House, Guildford, GU2 7XJ.

Copyright © 2018 AQA and its licensors. All rights reserved.

IB/M/Jun18/JW/7516/1/E3

