# Suggested Answers

## Number Representation

1. 64

2. 7

3. 80

4. 22

5. 196

6. 0001 0100

7. 1100 0000

8. 0100 0100

9. 0000 1111

10. 1111 1111

11. 1 mark for working (32 + 16 + 4 + 1) and 1 mark for the correct answer (53)

12. 1 mark for working (128 + 32 + 16 + 4 + 2 + 1) and 1 mark for the correct answer (183)

13. 1 mark for working and 1 mark for the correct answer (01001110)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **128** | **64** | **32** | **16** | **8** | **4** | **2** | **1** |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |

14. 1 mark for working and 1 mark for the correct answer (00111111)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **128** | **64** | **32** | **16** | **8** | **4** | **2** | **1** |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |

15. 0100 1011

16. 1111 1010

17. 6F

18. 5D

19. 1 mark for working (e.g. converting to binary first – 1011 1011) and 1 mark for each correct digit (BA)

20. 1 mark for working (e.g. converting to binary first – 0110 0001) and 1 mark for each correct digit (61)

21. 1 mark for working (e.g. converting to binary first - 1010 0111) and 1 mark for correct answer (167)

22. 1 mark for working (e.g. converting to binary first – 1111 1111) and 1 mark for correct answer (255)

## Data Representation & Compression

1. One mark per correct cell:

|  |  |
| --- | --- |
|  | **ASCII Code** |
| **Character** | **Denary** | **Binary** | **Hexadecimal** |
| H | 72 | 01001000 | 48 |
| I | **73** | **01001001** | **49** |
| J | **74** | **01001010** | **4A** |
| K | **75** | **01001011** | **4B** |
| L | **76** | **01001100** | **4C** |

2. 128

3. In lossy compression data is permanently removed (1) whereas lossless compresses data without permanently removing any of the data (1)

4. ZIP file (or any other suitable example)

5. MP3 or JPG/JPEG (or any other suitable example)

## Data Storage Units

1. 8

2. 1 mark per correct cell

|  |  |  |
| --- | --- | --- |
|  | **Abbreviation** | **Number of bytes** |
| **Kilobytes** | KB | 210 or 1024 |
| **Megabytes** | MB | 220 or 1,048,576 |
| **Gigabytes** | GB | 230 or 1,073,741,824 |

3. 10 x 210 or 10,240

4. 5 x 220 or 5,242,880

5. 8 x 230 or 8,589,934,592

6. 1 mark for correct answer (50MB) 1 mark for working (e.g. 51,200 / 1024)

7. 1 mark for correct answer (2GB) 1 mark for working (e.g. 2,048 / 1024)

## Error Detection & Correction

1. 1 mark for working and 1 mark for answer (not valid)

(5x6) + (9x5) + (3x4) + (8x3) + (4x2) + (5x1)

30 + 45 + 12 + 24 + 8 + 5

124/11 = 11.27 (not valid)

2. 1 mark for working and 1 mark for answer (valid)

(2x6) + (5x5) + (1x4) + (0x3) + (3x2) + (8x1)

12 + 25 + 4 + 0 + 6 + 8

55/11 = 5 (valid)

3. 1 mark for working and 1 mark for answer (6)

(7x6) + (9x5) + (3x4) + (4x4) + (0x2)

42 + 45 + 12 + 16 + 0

= 111

4. 1 mark for working and 1 mark for answer (2)

(2x6) + (4x5) + (1x4) + (8x3) + (2x2)

12 + 20 + 4 + 24 + 4

= 64

5. 1 mark for each correct row:

|  |  |  |
| --- | --- | --- |
| **Number** | **Odd Parity** | **Even Parity** |
| 0100100 | 01001001 | 01001000 |
| 1100110 | 11001101 | 11001100 |
| 1110101 | 11101010 | 11101011 |
| 0111010 | 01110101 | 01110100 |

6. 1 mark for each correct row:

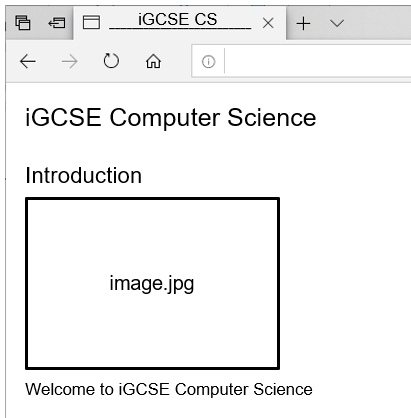
|  |  |  |
| --- | --- | --- |
| **Number** | **Parity** | **Pass?** |
| 01001011 | Even | Yes |
| 01110100 | Odd | No |
| 00010110 | Even | No |
| 00101001 | Odd | Yes |

## HTML

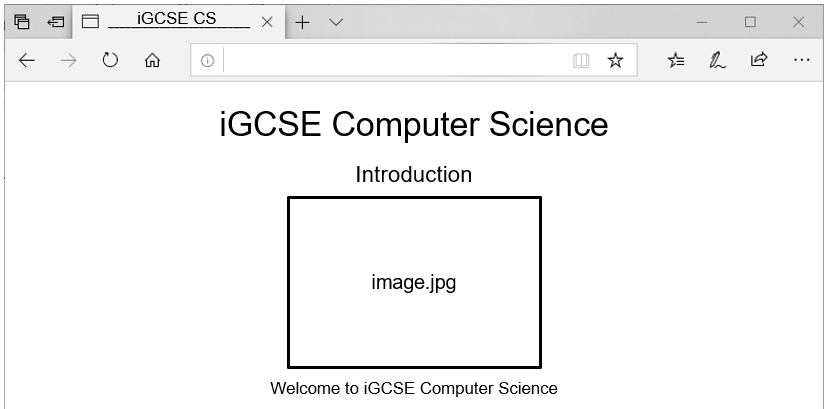
1. File A

2. File B

3. 1 mark for “iGCSE CS” in the title bar, 1 mark for page content being left aligned, 1 mark for the four correct elements appearing, 1 mark for them appearing in the correct order (text size doesn’t matter).



4. 1 mark for the elements being centred on the page, 1 mark for “iGCSE Computer Science” being the largest text, 1 mark for “Introduction” being the second largest text, 1 mark for “Welcome to iGCSE Computer Science” being the smallest text,



5. Any 2 of the following tags (1 mark each):

<html> / <body> / <head> / <h1> / <h2> / <img> / <p>

## Encryption

1. CPC MPWFT QSPHSBNNJOH 6. JULIUS CAESAR

2. DKPCTA KU GCUA 7. CAROL LOVES NETWORKING

3. DQGB OLNHV IRRWEDOO 8. IMAGES ARE MADE UP OF PIXELS

4. DMBQXOSHNM HR BNNK 9. ENCRYPT YOUR DATA TO KEEP IT SAFE

## Logic Gates

1. 1 mark per correct symbol

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| NOT |  |  | AND |  |  | OR |  |

2. 1 mark for two different inputs, 1 mark for the correct outputs:

|  |  |
| --- | --- |
| **INPUT** | **OUTPUT** |
| 0 | 1 |
| 1 | 0 |

3. 1 mark 4 different combinations of inputs, 1 mark for each pair of correct outputs.

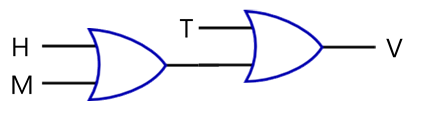
|  |  |  |
| --- | --- | --- |
| **INPUT 1** | **INPUT 2** | **OUTPUT** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

4. 1 mark 4 different combinations of inputs, 1 mark for each pair of correct outputs.

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **X** | **Y** |
| 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

5. 1 mark 4 different combinations of inputs, 1 mark for each pair of correct outputs.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **X** | **Y** | **Z** |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 |



6. 1 mark for two of the inputs going into an OR gate, 1 mark for the third input going into another OR gate, 1 mark for the output of the first OR gate going into the second OR gate.

7. 1 mark for inputs W and M going into an OR gate, 1 mark for the third input going into an AND gate, 1 mark for the output of the OR gate going into the AND gate.

## Designing Solutions

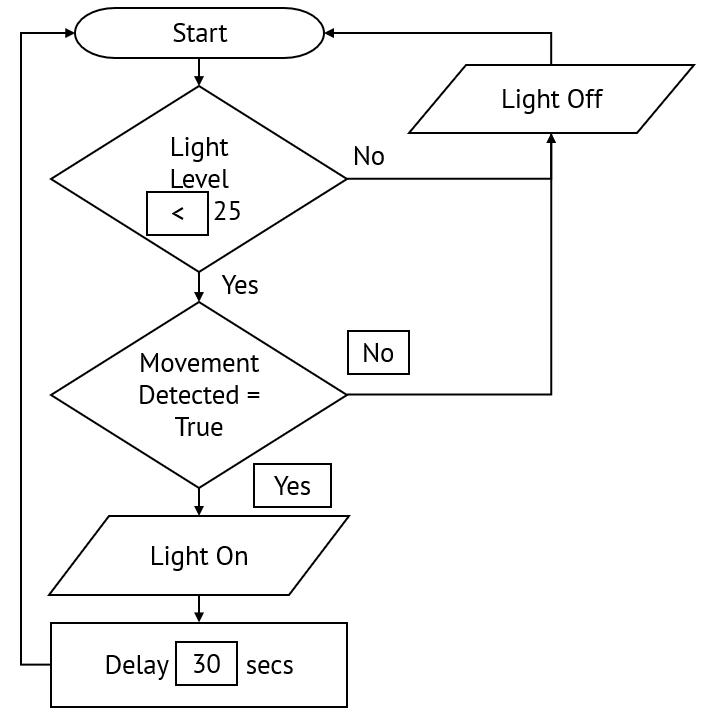
1. Algorithm A

2. It asks the user to input two numbers (1) adds them together (1) and outputs the result (1)

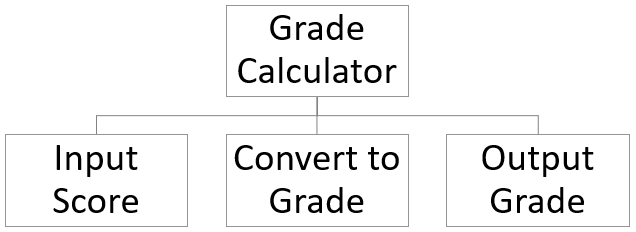
3. 1 mark per correct symbol:

|  |  |
| --- | --- |
|  | Used to control the path taken through an algorithm based on the result of a condition. |
|  | Used to indicate the start or end of an algorithm. |
|  | Used to indicate a process, for example performing a calculation. |
|  | Used when data needs to be inputted or outputted. |
|  | Used to call a pre-defined algorithm. |

4. *One mark for each correctly completed box:*



5a. *1 mark for a single top-level node with an appropriate title that describes the whole program, 1 mark for each 2nd level node that is a logical sub-task (to a maximum of 3).*



5b. *1 mark for correct logic for conversion of A grade, 1 mark for correct logic for conversion of B grade, 1 mark for correct logic for conversion of C grade, 1 mark for correct logic for conversion of D grade*

INPUT Score

IF Score >= 80

THEN

PRINT "A"

ELSE

IF Score >= 70

THEN

PRINT "B"

ELSE

IF Score >= 60

THEN

PRINT "C"

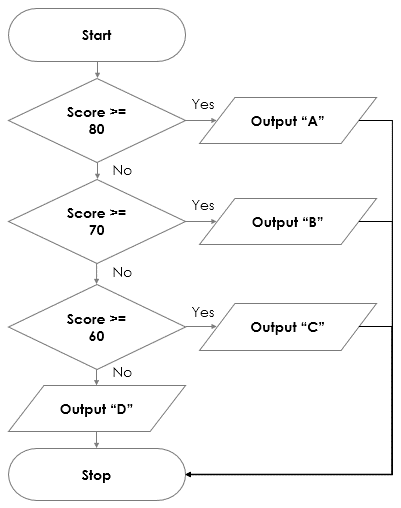
ELSE

PRINT "D"

ENDIF

ENDIF

ENDIF



5c. *4 x* *1 mark for correct logic for conversion of grades A-D; 1 mark for correct use of input/output symbols; 1 mark for correct use of decision symbols.*

*E.g.*

## Tracing Algorithms

1. 1 mark per column with values in the correct order (spacing between rows is not important).

|  |  |  |  |
| --- | --- | --- | --- |
| **Start** | **End** | **Index** | **Output** |
| 1 | 5 |  |  |
|  |  | 1 |  |
|  |  |  | 1 |
|  |  | 2 |  |
|  |  |  | 4 |
|  |  | 3 |  |
|  |  |  | 9 |
|  |  | 4 |  |
|  |  |  | 16 |

2. 1 mark per column with values in the correct order (spacing between rows is not important).

|  |  |  |
| --- | --- | --- |
| **Countdown** | **Target** | **Output** |
| 5 | 0 |  |
|  |  | 5 |
| 4 |  |  |
|  |  | 4 |
| 3 |  |  |
|  |  | 3 |
| 2 |  |  |
|  |  | 2 |
| 1 |  |  |
|  |  | 1 |
| 0 |  |  |
|  |  | “Blast Off!” |

3. 1 mark per column with values in the correct order (spacing between rows is not important).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **i** | **Score** | **Len** | **Answers[i]** | **Responses[i]** |
| 0 | 0 | 5 |  |  |
|  |  |  | TRUE | TRUE |
|  | 1 |  |  |  |
| 1 |  |  |  |  |
|  |  |  | TRUE | FALSE |
| 2 |  |  |  |  |
|  |  |  | FALSE | TRUE |
| 3 |  |  |  |  |
|  |  |  | FALSE | FALSE |
|  | 2 |  |  |  |
| 4 |  |  |  |  |
|  |  |  | TRUE | TRUE |
|  | 3 |  |  |  |
| 5 |  |  |  |  |

## Validation, Verification & Testing

1. Algorithm C

2. Algorithm B

3. Algorithm A

4. Type check or check digit

5. 1 mark for completed second row with test data of 11 characters:

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Test Type** | **Test Data** | **Expected Outcome** |
| Test a value longer than the expected length. | Erroneous | 070000000000000 | “Number Invalid” |
| Test a value of the expected length. | Normal | 0700000000000 | “Number Valid” |

6. *One mark for each correctly completed row. For test one students can choose highest or lowest value but the test data must match the description. For test two students can choose a higher or lower value but the test data must match the description.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Test Type** | **Test Data** | **Expected Outcome** |
| Test the *highest/lowest* allowable integer | Boundary | *11/18* | “Within range” |
| Test a value *higher/lower* than allowable range | Erroneous | *e.g. 5 or 22* | “Out of range” |
| Test a value within the allowable range | Normal | *Any value between 11 and 18* | “Within range” |

7. *1 mark for input of temperature reading, 1 mark for correct logic to test if the reading is within range, 1 mark for suitable message outputted if temperature is in range, 1 mark for suitable message outputted if temperature is not in range.*

|  |  |
| --- | --- |
| **Example Solution 1** | **Example Solution 2** |
| Temp ← INPUT  IF temp < -20 OR temp > 40  THEN  OUTPUT "Temperature out of range"  ELSE  OUTPUT "Temperature within range"  ENDIF | Temp ← INPUT  IF temp => -20 AND temp <= 40  THEN  OUTPUT "Temperature within range"  ELSE  OUTPUT "Temperature out of range"  ENDIF |

8. *One mark for each correctly completed row. For test one students can choose highest or lowest reading but the test data must match the description. For test two students can choose a higher or lower value but the test data must match the description.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Test Type** | **Test Data** | **Expected Outcome** |
| Test the *highest/lowest* allowable reading | Boundary | *-20/40* | “Within range” |
| Test a value *higher/lower* than allowable range | Erroneous | *e.g. -25 or 45* | “Out of range” |
| Test a value within the allowable range | Normal | *Any value between -20 and 40* | “Within range” |

## Programming Concepts

1. Identify any 2 or more lines between 3 and 7

2. Line 2

3. Line 2

4. Line 8

5. Program B

6. Program A

7. Asks the user to input 2 numbers (1) adds them together (1) gives the user the option to repeat the program (1)

8. Asks the user to input a number (1) outputs “Hello” that number of times (1)

9. 1 mark per correct row.

|  |  |
| --- | --- |
| **=** | Equal to |
| **>** | Greater than |
| **<** | Less than |
| **≥** | Greater than or equal to |
| **≤** | Less than or equal to |
| **<>** | Not equal to |

10. The program uses a loop (1) that repeats 20 times (1), starts by outputting 20 (1), and counts down to 1 (1)

Counter = 20

WHILE Counter >= 1

PRINT Counter

Counter ← Counter - 1

ENDWHILE

## Data Types

1. One mark per correctly identified data type.

|  |  |  |
| --- | --- | --- |
| **Variable Name** | **Example Data** | **Data Type** |
| PlayerName | “John Smith” | String |
| BestScore | 5000 | Integer |
| WorstScore | 109 | Integer |
| AverageScore | 3437.5 | Real/Float |
| GameCompleted | FALSE | Boolean |
| Difficulty | “M” | Character |

2. The number has a fractional part / a decimal place.

3. One mark per suitable example and one mark per appropriate data type.

|  |  |  |
| --- | --- | --- |
| **Variable Name** | **Example Data** | **Data Type** |
| FirstName | e.g. “Alex” | String |
| LastName | e.g. “Roberts” | String |
| Gender | e.g. “M” or “Male” | Character / String |
| Age | e.g. 15 | Integer |
| AverageAttendance | e.g. 95.1 | Real / Float |
| CurrentStudent? | e.g. TRUE | Boolean |

## Data Structures

1. Polly

2. names[6]

3. 1 mark for writing the list of values in the correct order separated by commas, 1 mark for assigning them to names.

names = ["Ben","Susan","Polly","Steven","Jamie","Victoria"]

Don’t penalise for not placing the values in speech marks.

4. 16

5. sales[4]

6. 1 mark for writing the list of values in the correct order separated by commas, 1 mark for assigning them to sales.

sales = [87, 16, 58, 29, 93, 73]

7. 1 mark for correct initialisation of the for loop, 1 mark for outputting the current value inside the for loop. Accept any valid form of pseudocode.

FOR i ←1 TO LEN(names)

PRINT names[i]

END FOR

8. students[1:25]

## Databases

1. Any one from: Dog\_ID, Name, Breed, Age, Gender

2. The data from any one of the rows e.g. 1, Daisy, Poodle, 2, F

3. Dog\_ID

4. 1 mark per correctly identified record:

1, Daisy, Poodle, 2, F

2, Jack, Labrador, 5, M

5. 1, Daisy, Poodle, 2, F

6. Gender = “M”

7. Breed = “Labrador”

8. Stock\_Level > 0

9. 1 mark per condition and 1 mark for the AND logical operator:  
(Genre = “Family”) AND (Rating = “U”)

10. 1 mark per condition and 1 mark for the OR logical operator:  
(Genre = “Action”) OR (Genre = “Horror”)