Zig Zag Education

Computer Science AS & A Level | AQA | 7516 & 7517

2015 specification
first exams in 2017 (2016 for AS)

AQA

TECHNICAL TOPICS

Presentations and Worksheets

for A Level AQA Computer Science

Includes AS and A Level

AK8/
6063

POD
6063

zigzageducation.co.uk

Contents

Thank You for Choosing ZigZag Education iii

Teacher Feedback Opportunity iv

Terms and Conditions of Use v

Teacher's Introduction 1

Worksheets 2

Sequence and Selection 2

Iteration 3

Subroutines 4

Recursion 5

Arrays 7

Stacks and Queues 9

Linked Lists and Hash Tables 11

Graphs and Trees 12

Vectors 13

Graph and Tree Traversal 14

Reverse Polish Notation 15

Searching Algorithms 16

Sorting Algorithms 17

Shortest Path Algorithm 18

Big O Notation 19

Tracing an Algorithm 20

Finite-State Machines 21

Maths for Regular Expressions 22

Regular Expressions 23

Backus–Naur Form 24

Turing Machine 25

Binary and Hexadecimal 26

Binary Arithmetic 27

Two's Complement 28

Fractions 29

Error Checking 31

Bitmapped Images 32

Vector Graphics 33

Digital Sound 34

Data Compression 35

Encryption 36

Logic Gates 37

Boolean Algebra 38

Assembly Language 39

Relational Databases 40

Structured Query Language 41

Normalisation 43

Answers 44

Sequence and Selection 44
Iteration 44
Subroutines 44
Recursion 45
Arrays 45
Stacks and Queues 46
Linked Lists and Hash Tables 46
Graphs and Trees 47
Vectors 47
Graph and Tree Traversal 48
Reverse Polish Notation 48
Searching Algorithms 48
Sorting Algorithms 49
Shortest Path Algorithm 49
Big O Notation 50
Tracing an Algorithm 50
Finite-State Machines 51
Maths for Regular Expressions 51
Regular Expressions 51
Backus–Naur Form 51
Turing Machine 52
Binary and Hexadecimal 52
Binary Arithmetic 53
Two's Complement 53
Fractions 53
Error Checking 53
Bitmapped Images 54
Vector Graphics 54
Digital Sound 55
Data Compression 55
Encryption 56
Logic Gates 56
Boolean Algebra 57
Assembly Language 57
Relational Databases 58
Structured Query Language 58
Database Normalisation 59

Appendix: Printouts of the Animated Presentations

Teacher's Introduction

This resource is designed to support the delivery of the logical and mathematical topics from the A Level AQA specification (for first teaching in September 2015; first exams from June 2016).

The topics covered are as follows:

1. Sequence and Selection
2. Iteration
3. Subroutines
4. Recursion*
5. Arrays
6. Stacks and Queues*
7. Linked Lists and Hash Tables*
8. Graphs and Trees*
9. Vectors*
10. Graph and Tree Traversal*
11. Reverse Polish Notation*
12. Searching Algorithms*
13. Sorting Algorithms*
14. Shortest Path Algorithm*
15. Big O Notation*
16. Tracing an Algorithm
17. Finite-State Machines++
18. Maths for Regular Expressions*
19. Regular Expressions*
20. Backus-Naur Form*
21. Turing Machine*
22. Binary and Hexadecimal
23. Binary Arithmetic
24. Two's Complement
25. Fractions++
26. Error Checking++
27. Bitmapped Graphics
28. Vector Graphics*
29. Digital Sound
30. Data Compression
31. Encryption
32. Logic Gates++
33. Boolean Algebra
34. Assembly Language
35. Relational Databases*
36. Structured Query Language*
37. Database Normalisation*

* This entire topic is for A Level only

++ This topic is covered at AS but also contains some A Level-only content

For each of the topics above, there is an animated presentation, providing a step-by-step walk-through of the key concept, plus a worksheet giving students the opportunity to demonstrate their understanding.

These presentations and accompanying worksheets can be used in a number of ways:

- ✓ The animated presentations and worksheets can be used in class to introduce topics.
- ✓ The worksheets can be used as homeworks to test understanding.
- ✓ The animated presentations make perfect revision aids.
- ✓ As part of a flipped classroom, where students watch the animated presentations as preparation for the lesson. The students could complete the worksheets in class to test their understanding prior to a more in-depth discussion of the topic.

The animated presentations are provided in PowerPoint (PPTX), HTML5 and PDF formats. The HTML5 versions are included so that students can use the presentations more easily on devices which lack PowerPoint support (such as tablet computers and even smartphones), making them great for revision. Hard copies of the PDF versions have been included at the back of this pack.

Answers are provided for each worksheet which facilitate self and peer assessment.

As this resource also includes all the content needed for the separate AS qualification (for first teaching in September 2015, with the first exams in June 2016), content which is only required for the A Level course is indicated with the icon shown on the right.

A Level Only

The CD-ROM contains the animated presentations in three formats (PPTX, HTML5 and PDF), which are linked together via a HTML frontend **(index.html)**.

If using on a network, it is recommended that you provide a shortcut to the frontend to allow easy access for your students.

Alternatively, you can access the individual files directly (without using the frontend), simply by navigating to the relevant folder on the CD.

Sequence and Selection

Program A

```
01  INPUT A, B
02  Total = A + B
03  Average = Total / 2
04  OUTPUT Average
```

Program B

```
01  INPUT A, B
02  IF A == B THE
03        OUTPUT
04  ELSE IF A > B
05        OUTPUT
06  ELSE
07        OUTPUT
08  END IF
```

1. Which one of the two programs shown above is an example of a seque

2. Give the line number of a statement from the program you identified in question. (1)

3. Identify the different variables that are used in Program A. (3)

4. Which one of the two programs shown above is an example of selectio

5. Give the line numbers of the conditions from the program you identifie question. (2)

6. Write the meaning of each of the comparison operators shown the in ta

Operator	Meaning
>	
<	
>=	
<=	
=	
<>	

Iteration

Program A	Program B
01 INPUT Num 02 FOR i = 1 to Num 03 OUTPUT i 04 NEXT i	01 Password = " 02 WHILE Passwo 03 INPUT 04 END WHILE 05 OUTPUT "Pass

```
Program A
01  INPUT Num
02  FOR i = 1 to Num
03        OUTPUT i
04  NEXT i
```

```
Program B
01  Password = "
02  WHILE Passwo
03        INPUT
04  END WHILE
05  OUTPUT "Pass
```

1. Which one of the two programs shown above features an example of a

2. Give the line number of the condition from the program you identified i question (1)

3. Which one of the two programs shown above features an example of a

4. Describe the purpose of Program A. (3)

5. Describe the purpose of Program B. (3)

6. Explain the difference between REPEAT UNTIL and WHILE loops. (3)

7. Rewrite Program A using a WHILE loop. (3)

Subroutines

Program A

```
__________ areaCalc(W, H)
      Area = W * H
      OUTPUT Area
END __________
areaCalc(10, 8)
```

Program B

```
_________ average(A, B, C)
      Total = A + B + C
      Average = Total / 3
      RETURN Average
END _________
average(4, 3, 4)
```

P

1. Identify a program that contains a function from the three shown above

2. Identify the parameters that are defined in Program B. (3)

3. Identify the arguments that are passed to the areaCalc subroutine in Pr

4. Identify the arguments that are passed to the compare subroutine in Pr

5. What would be the output of the areaCalc subroutine based on the call i

6. What would be the output of the compare subroutine based on the cal

7. Describe the difference between a procedure and a function. (2)

Recursion

Program A

```
01 PROCEDURE count(C)
02     IF A <= 0 THEN
03         OUTPUT 0
04     ELSE
05         OUTPUT C
06         count(C-1)
07     END IF
08 END PROCEDURE
```

Program B

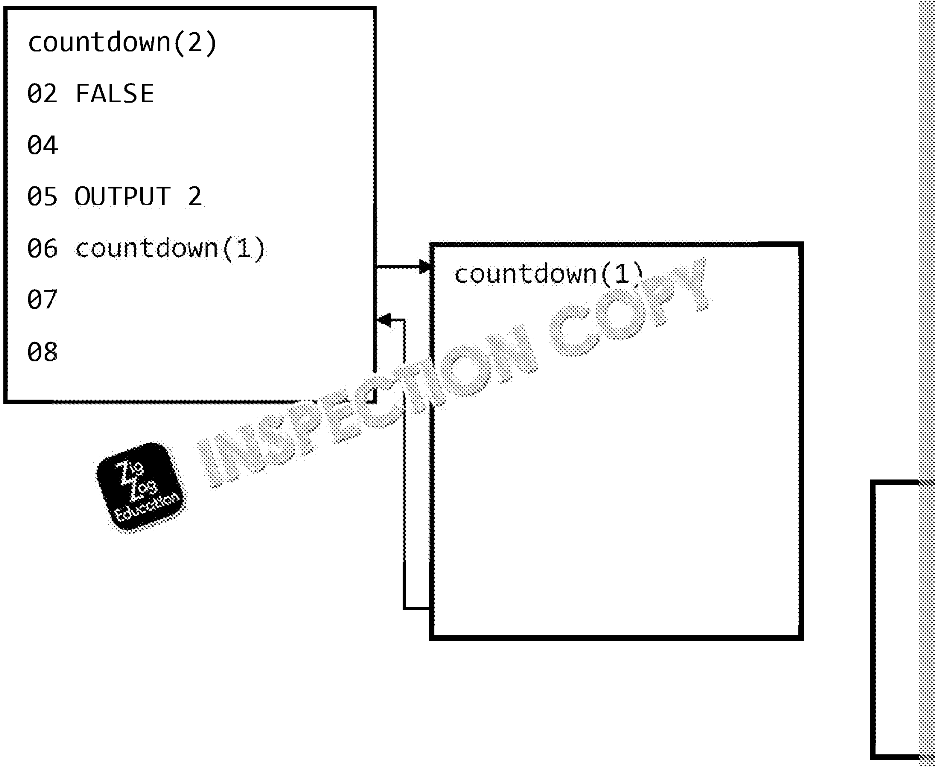
```
01 PROCEDURE count(C)
02     WHILE C >= 0
03         OUTPUT C
04         C = C -1
05     END WHILE
06 END PROCEDURE
```

1. In which of the programs shown above does recursion occur? (1)

2. Give the line number where recursion occurs in the program you identi previous question. (1)

3. Which two programs will produce the same result? (1)

4. Complete the diagram below to show what happens when Program A i countdown(2). (7)



5. Complete the trace table below to show what happens when Program A
countdown(3). (4)

C	OUTPUT
3	
	3
2	
	2
1	
	1
0	
	0

6. What is a disadvantage of using recursion instead of iteration? (3)

Arrays

Use the names array shown below to help you to answer the following ques

Index	0	1	2	3	4
Value	Susan	Ian	Barbara	Steven	Ben

1. State the value of names[3]. (1)

2. State the value of names[illegible]. (1)

3. How can the value 'Ben' be accessed in the names array? (1)

4. How can the value 'Sarah' be accessed in the names array? (1)

5. Create the names array using pseudocode. (2)

6. Write a FOR loop that will cycle through the names array, outputting ea

Use the scores array shown below to help you to answer the following ques

	0	1	2	
0	45	71	34	5
1	23	82	57	3
2	18	31	53	5
3	32	58	97	4

7. State the value of scores[3][1]. (1)

8. State the value of scores[1][2]. (1)

9. How can the value 71 be accessed in the scores array? (1)

10. How can the value 97 be accessed in the scores array? (1)

11. Create the scores array using pseudocode. (3)

Stacks and Queues

1. Draw a diagram below to show the state of a stack after the values Polly and Ben have been added to it. (4)

2. Draw a diagram below to show the state of the stack from question 1 a

3. What is the current value of the variable used to store the position of th

4. Draw a diagram below to show the state of a circular queue after the va have been added to it. Make sure you also show the location of the Fro

5. Draw a diagram below to show the state of the circular queue from que left the queue. Show the location of the Front and Rear pointers. (2)

6. Draw a diagram below to show the state of the circular queue from que been added to the queue. Show the location of the Front and Rear pointers. (2)

Linked Lists and Hash Tables

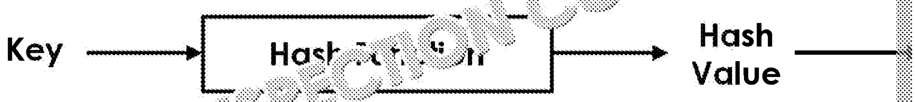
1. Describe the drawback of arrays and how this can be addressed using li

2. Describe the linked list data structure. (2)

3. Describe the drawback of linked lists and how this can be addressed usi

4. Describe the hash table data structure. (2)

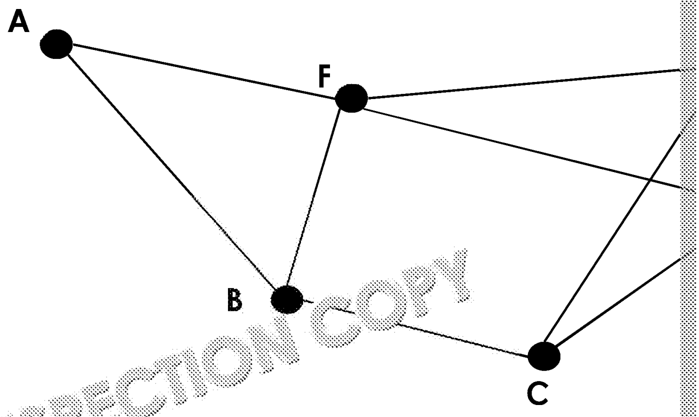
5. The following hash table uses a hash function that generates a hash valu the key. Write the value 'David' in the appropriate place in the table. (1)



6. Show how the value 'Diane' would be added to the hash table above usi

Graphs and Trees

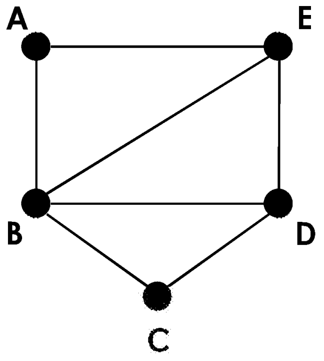
1. Label the graph to show examples of vertices and edges. (2)



2. State two vertices that are neighbours in the graph above. (1)

3. What is the degree of F in the graph above? (1)

4. Complete the adjacency matrix for the graph shown below. (5)



	A	B	C	D
A				
B				
C				
D				
E				

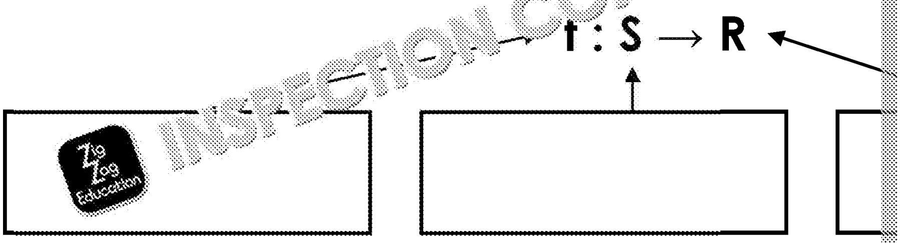
5. Complete the adjacency list for the graph shown above. (5)

Vertex	Adjacent Vertices
A	
B	
C	
D	
E	

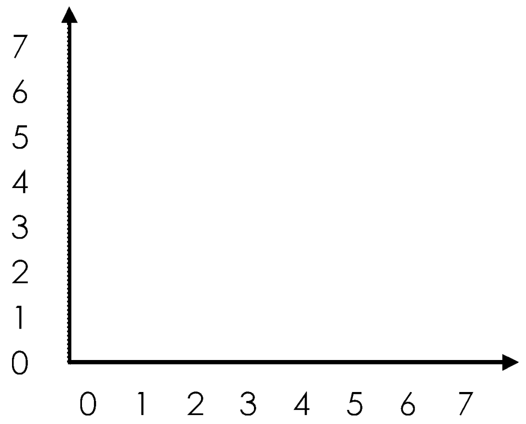
Vectors

1. What is the main difference between arrays and vectors? (2)

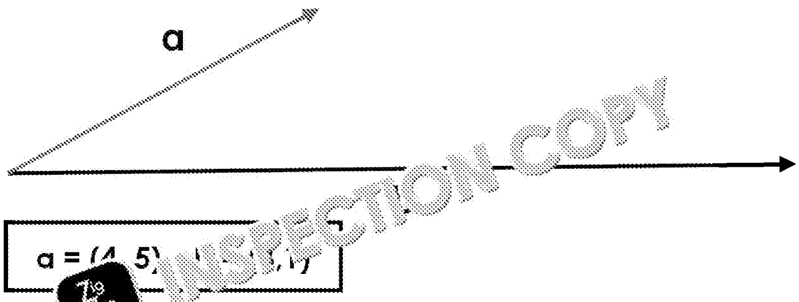
2. Label the representation of a vector as a function below. (3)



3. The position of an arrowhead can be represented by (5,2); show its pos

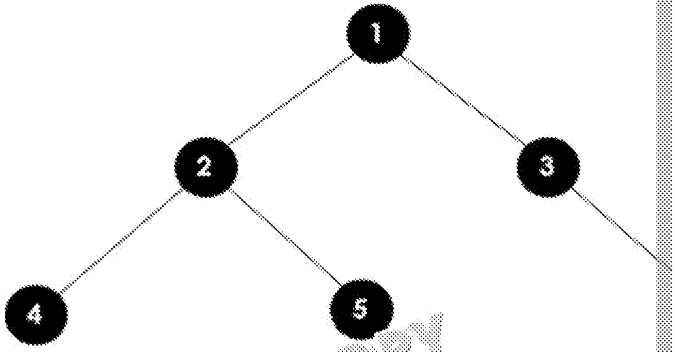


4. Add a new vector between the tail of vector a and the tail of vector b. M



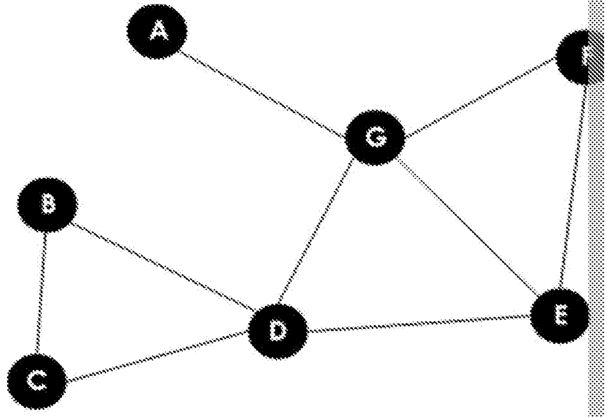
5. Calculate the dot product of vectors a and b from question 4. (2)

Graph and Tree Traversal



1. Write the sequence in which the nodes in the tree below will be visited w

2. Write the sequence in which the nodes in the tree above will be visited wh



3. Write the sequence in which the nodes in the tree above will be visited w

4. Complete a depth-first traversal on the graph above. (7)

Vertex Visited	Stack

Reverse Polish Notation

Convert the following reverse Polish notation expressions to their equivalent

1. 4 6 +

2. 8 2 -

3. 4 9 + 6 *

4. 10 4 7 + *

5. 3 9 + 4 2 - *

Convert the following infix notation expressions to their equivalent reverse

6. 4 + 3

7. 56 – 40

8. 5 * (5 - 3)

9. (6 / 3) + [illegible] 2)

10. (18 – 8) * (30 + 20)

Searching Algorithms

1. Explain how the linear search algorithm works. (2)

2. How many comparisons would be needed to find the value 23 in the lis

   5, 9, 11, 15, 23, 45, 54, 58, 61

3. Create a binary search tree for the list shown below. (3)

   45, 32, [illegible], 9, 38, 23, 6

4. Create a binary search tree for the list shown below. (3)

   56, 12, 4, 7, 98, 32, 65, 86, 26, 77

5. How many comparisons would be needed to find the value 32 in the tre

6. How many comparisons would be needed to find the value 65 in the tre

Sorting Algorithms

1. Show the list below at each stage of sorting using the bubble sort algor

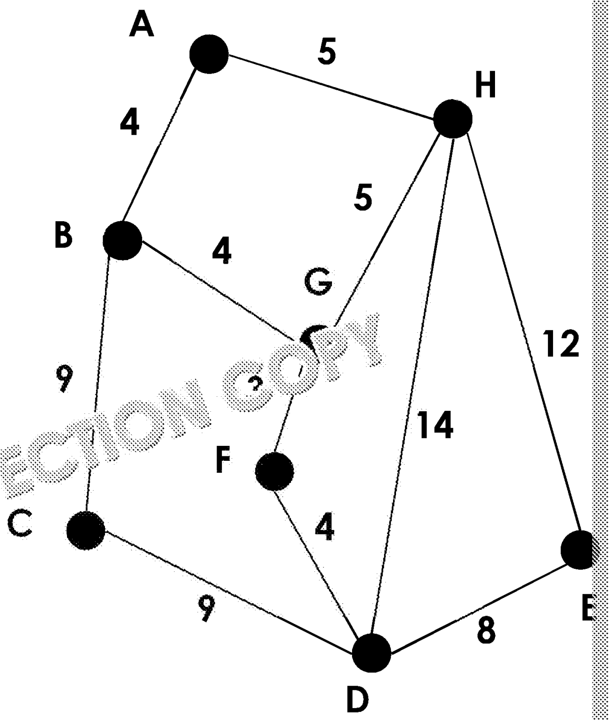
   45, 32, 2, 78, 5, 9

2. Show the list below at each stage of sorting using the merge sort algori

   45, 32, 2, 78, 5, 9, 38, 23

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td colspan="2"></td><td colspan="2"></td><td colspan="2"></td></tr>
<tr><td colspan="4"></td><td colspan="2"></td></tr>
<tr><td colspan="6"></td></tr>
</table>

Shortest Path Algorithm



1. Use Dijksta's shortest path algorithm to find the shortest path between
   on the graph. (8)

Node	Shortest Distance from Vertex A	Previous Node

2. What is the shortest path between vertex A and vertex G? (1)

3. What is the shortest path between vertex A and vertex E? (1)

Big O Notation

1. Complete the table below comparing each type of time complexity. (5)

Complexity	Description
Constant	
Linear	
Polynomial	
Exponential	
Logarithm	

2. Which standard algorithm has linear complexity? (1)

3. Which standard algorithm has polynomial complexity? (1)

4. Which two standard algorithms have logarithmic complexity? (2)

5. Number the below from 1 to 5, 1 being the best performance and 5 be

$O(1)$	
$O(n)$	
$O(2^n)$	
$O(\log_2(n))$	
$O(n^k)$	

Tracing an Algorithm

Program A

```
01  A = 1
02  B = 5
03  IF A > B THEN
04        OUTPUT A
05  ELSE
06        OUTPUT B
07  END IF
```

Program B

```
01  C = 0
02  WHILE C < 3
03        OUTPUT
04        C = C
05  END WHILE
```

1. Trace Program A by completing the trace table below. (6)

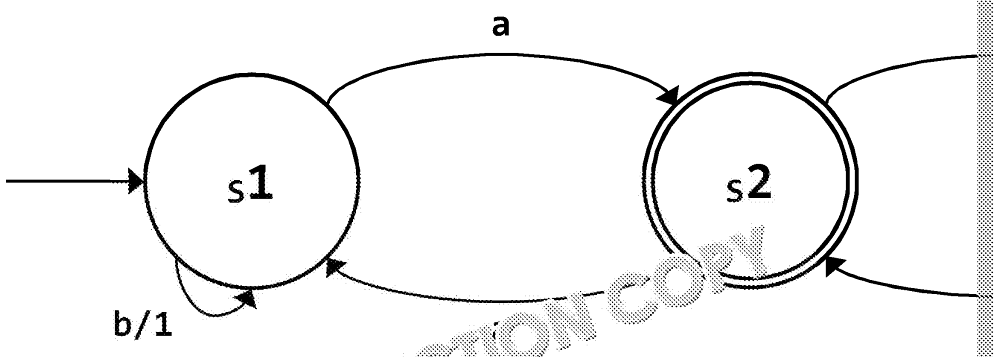
Line	A		OUTPUT	C

2. Trace Program B by completing the trace table below. (14)

Line	C	OUTPUT	Comm

Finite-State Machines

This FSM accepts input sequences consisting of a sequence of letters, for ex



1. Would the sequence aab be a valid or invalid input for the FSM above? (

2. Would the sequence abb be a valid or invalid input for the FSM above? (

3. Give another example of a string of letters that would be rejected by th

4. Give another example of a string of letters that would be accepted by t

5. Complete the state transition table for the FSM shown above. (6)

Current State	Input	Next Stage

6. What output would the sequence babababa produce? (1)

Maths for Regular Expressions

1. Define an empty set called A. (1)

2. What is the cardinality of the set below? (1)

   B = { 1, 2, 3, 4, 5, 6 }

3. Explain the set that this regular expression represents. (2)

   A = { y | y ∈ ℤ ^ [illegible]

4. Which type of set is represented in question 2? (1)

5. What is the Cartesian product of the two sets shown below? (1)

   A = { x, y, z }

   B = { 1, 2, 3 }

6. { 1, 2, 3 } is a subset of { 1, 2, 3, 4, 5, 6 }; what type of subset is this? (1)

7. Complete the table below by entering the correct name for each set op

Operation	
Joining sets together to form a new set keeping only unique elements from each list	
Joining sets together to form a new set containing elements from both	
Joining sets together to form a new set keeping only the elements that are contained in both	

Regular Expressions

1. Which three of the following strings would this regular expression prod

(a|b)c+

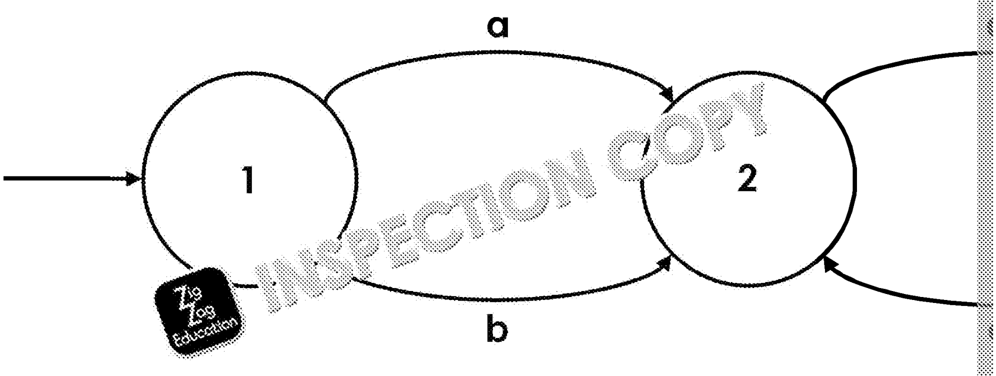
String	Yes/No
ac	
ab	
accccc	
bc	

2. Describe what this regular expression would produce. (4)

a*b+c?

3. Explain why two of the strings would not be produced by the regular ex

4. Write a regular expression that is equivalent to the FSM shown below. (



Backus–Naur Form

This is a set of BNF production rules to define a language of test details:

```
<test> ::= <test-title> <subject> <level> <max-score>
<subject> ::= English | Maths | Science
<max-score> ::= <integer>
<integer> ::= <digit> | <digit><integer>
```

1. Write the rule for the `<level>` element that will accept any one of thes

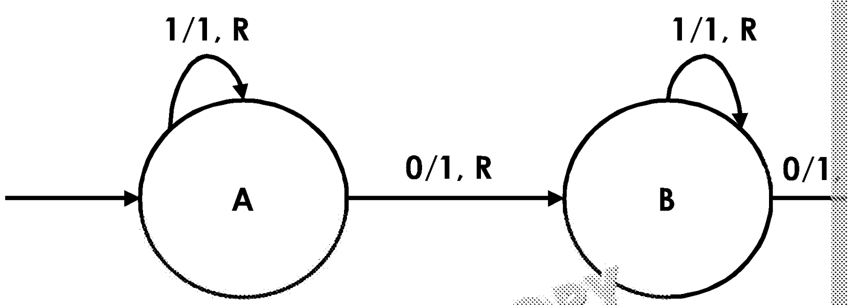
2. Which of the following test details would not be accepted by the B

Test Details	
Test Title: Networking Level: KS4	Subject: Computing Max Score: 50
Test Title: Macbeth Level: KS4	Subject: English Max Score: 25
Test Title: States of Matter Level: KS3	Subject: Science Max Score: 10
Test Title: Fractions Level: KS4	Subject: Maths Max Score: 40.5
Test Title: Percentages Level: KS4	Subject: Maths Max Score: 25

3. Create a syntax diagram to represent the production rules specified abo

Turing Machine

1. The state transition diagram below represents a transition function for a



2. Represent the state transition diagram above using a state table. (4)

State	Read	Write	Move

3. Represent the state transition diagram above using written notation. (4)

4. The paper tape is represented in the top row of the table below. Trace t function depicted below; use an arrow to indicate the position of the re machine is currently in state A. (6)

```
δ (A, 0) = (A, 1, →)
δ (A, 1) = (A, 1, →)
δ (A, □) = (B, □, →)
δ (B, 0) = (B, 0, →)
δ (B, 1) = (B, 0, →)
δ (B, □) = (A, □, →)
```

1	0	□	1	1
↑				

Binary and Hexadecimal

1. Convert the decimal number 45 into binary. (2)

128	64	32	16	8	4	2	1

2. Convert the binary number 01100101 into decimal. (2)

3. Conver[illegible]ec[illegible]nal number 165 into hexadecimal. (2)

4. Convert the hexadecimal number B6 into decimal. (2)

5. Convert the hexadecimal number 9B into binary. (2)

6. Convert the binary number 10010010 into hexadecim[illegible]l. (1)

Binary Arithmetic

Complete the following calculations (give your answers using 5 bits). (24)

1. 0111

0101 +

2. 0101

0111 +

3. 1011

0111 +

4. 1011

0011

5 1[illegible]

1011 +

6. 0111

1010 +

7. 0111

0111 ×

8. 1101

0111 ×

9. 1011

1111 ×

10. 1011

0101 ×

11. 1011

1001 ×

12. 0111

1000 ×

Two's Complement

1. Convert this two's complement binary integer to decimal. (2)

   `10110110`

2. Convert this two's complement binary integer to decimal. (2)

   `00110010`

3. Convert this decimal number to an 8-bit two's complement binary integ

   `-122`

4. Convert this decimal number to an 8-bit two's complement binary integ

   `72`

5. Convert this decimal number to an 8-bit two's complement binary integ

   `-98`

Complete the following calculations. (6)

```
6.    0111          7.    1101          8.    0011

    - 0101              - 0111              - 0111
```

Fractions

Note: you only need to use two's complement with floating-point numbers, not fix

1. Convert this fixed-point binary number to decimal: (2)

1	0	1	1 ●	1	0	0	0

2. Convert this decimal value to fixed-point binary, with 4 bits before and

7.5

3. Convert this fixed point binary number to decimal: (2)

1	0	0	1 ●	1	1	0	0

4. Convert this decimal value to fixed-point binary, with 4 bits before and

12.625

5. Convert this floating-point binary number to decimal: (2)

0 ●	1	0	1	0	0	0	1	0
Mantissa					Exponent			

6. Convert this floating-point binary number to decimal: (2)

1 ●	1	0	1	1	0	0	1	1
Mantissa					Exponent			

7. Represent this decimal number using floating-point binary, using 8 bits
the exponent: (2)

`-4.5`

8. The decimal value 17.27 has been represented as 17.25. Calculate the a

Absolute:	Relative:

9. Normalise this floating-point binary number: (2)

Mantissa				
1 ● 1	0	1	1	0

Exponent		
0	1	0

10. Normalise this floating-point binary number: (2)

Mantissa				
0 ● 0	0	1	1	0

Exponent		
0	0	1

Error Checking

1. Complete these even parity bit patterns. (3)

	0101110
	1101001
	0010110

2. Complete these odd parity bit patterns. (3)

	0101110
	1101001
	0010110

3. These bit patterns have been transferred using even parity; which one c

1001[illegible]
10111011
01001111

4. The following data has been transferred using majority voting. Complet accepted values. (3)

Received Bits	Accepting Value
110	
010	
100	

5. Identify the method of error checking from the description. (2)

Method	Descrip
	Calculated by adding together and sent along with the packet
	Used to validate data entry, add and calculated using the rest of

6. Give one example of the use of check digits in the real world. (1)

7. Calculate the check digit for the number 2651 using the MOD11 metho

Bitmapped Images

1. Complete the table below showing possible representations of rows 2 t

			Grey				→	00 00
			Grey				→	
Green	Green	Green	Blue	Green	Green	Green	→	
			Grey				→	
			Grey				→	

2. What is the resolution of the image above? (1)

3. What is the resolution in pixels of the image above? (1)

4. What is the colour depth of the image above? (1)

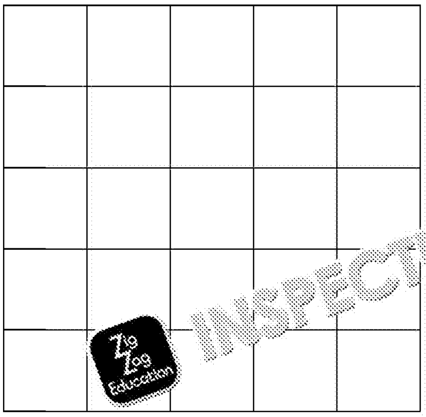
5. What are the storage requirements of the image above? (2)

6. Give an example of the additional data that is included in image metad

Vector Graphics

1. Complete the vector graphic below based on these properties. (3)

   Object type: triangle
   Coordinates: (1,1) (4,1) (1,4)
   Fill colour: grey



2. Name two other properties that the shape above could have. (2)

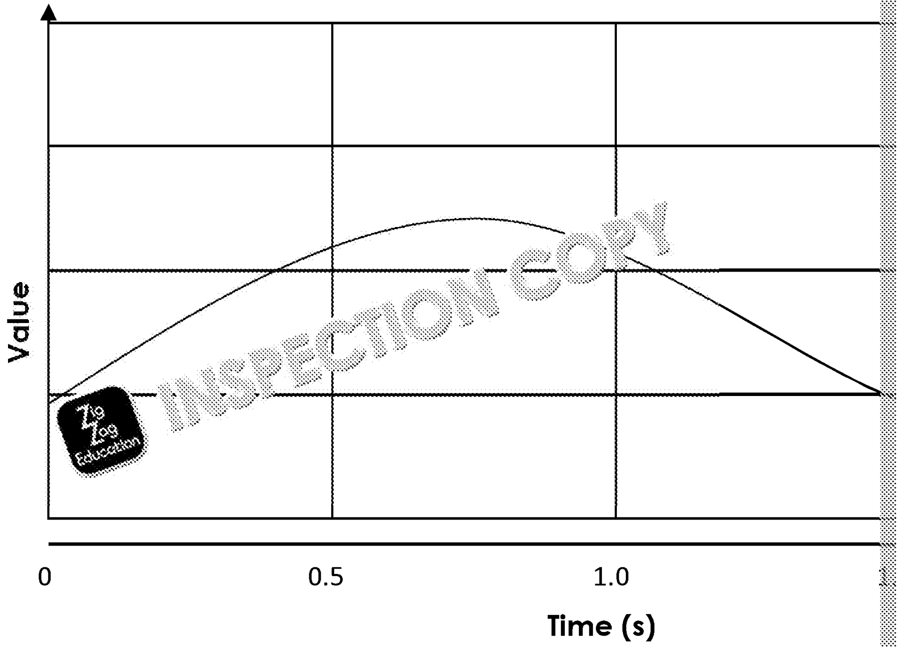
3. State two advantages of the use of vectors over bitmapped images. (2)

4. State two disadvantages of the use of vectors over bitmapped images. (

5. How are the properties of the geometric shapes in vectors stored? (1)

Digital Sound

1. On the following wave, indicate with an X what data would be sampled with a bit depth of 2. (2)



2. A stereo music file is sampled at 44,000 Hz, with each sample having a for 300 seconds, how many kilobytes will the file be? (3)

3. A stereo music file is 600 kB in size. If each sample is 8 bits in depth and 16,000 Hz, how long will the file last? (3)

4. A file has been sampled at 30,000 Hz. State and explain the maximum f should contain. (2)

Data Compression

1. Below is an image encoded in binary. Apply run-length encoding to eac

1	1	0	1	1	→	
1	1	0	1	1	→	
0	0	0	0	0	→	
1	1	0	1	1	→	
1	1	0	1	1	→	

2. Apply run-length encoding (RLE) to this string of text: (1)
   CCCCCLLL[illegible]QQ[illegible]P

3. Suggest possible codes that could be used to represent each line of the dictionary-based compression. (5)

1	1	0	1	1	→	
1	1	0	1	1	→	
0	0	0	0	0	→	
1	1	0	1	1	→	
1	1	0	1	1	→	

4. State one advantage and one disadvantage of lossy compression comp

Absolute:	Relative:

Encryption

1. Encrypt the plain text below using the Caesar cipher with a right shift of

   `COMPUTER SCIENCE ROCKS`

2. Encrypt the plain text below using the Caesar cipher with a right shift of

   `HAIL CAESAR`

3. Decrypt the cipher text below using the Caesar cipher with a right shift

   `UGEWTG`

4. Decrypt the cipher text below using the Caesar cipher with a right shift

   `EOHWFKOHB`

5. Encrypt the plain text below using the Vernam cipher (for this question

   Plaintext: `CO`   One-time pad (OTP): `KITOS`

C			O	
Plaintext	**OTP**	**Ciphertext**	**Plaintext**	**OTP**

Logic Gates

1. Draw the following logic gates: (3)

AND	OR

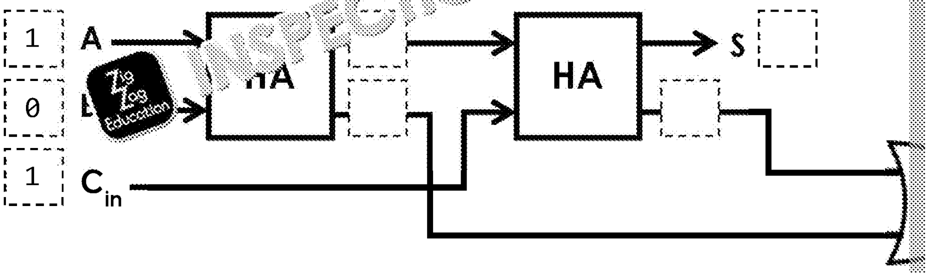
2. Complete this truth table for an OR gate. (4)

Input A	Input B	Output Q

3. Draw the logic diagram for this expression. (3)

$\overline{(A+B).C}$

4. Label this full adder diagram showing the outputs at each stage. (5)



Boolean Algebra

Simplify the following:

1. $(A + B) . (A + \overline{B})$ (2)

2. $\overline{P}.\overline{Q}.R + \overline{P}.Q.R + P.R$ (1)

3. Which of following statements will simplify to the same expression? (2)

   $A.1$ $A.\overline{A}$ $A.A$ $A+1$

4. Which of the following is <u>not</u> a statement of De Morgan's Theory? (1)

   $\overline{A+B} = \overline{A}.\overline{B}$ $\overline{A.B} = \overline{A + B}$ $\overline{A.B} = \overline{A} + \overline{B}$

5. Simplify the following using De Morgan's Theory. (3)

   $\overline{(X + Y)} . (Y + Z)$

Assembly Language

1. Show the contents of the registers as each instruction is executed. (3)

	R0	R1	R2
		5	3
MOV R0, #9			
ADD R0, R0, R2			
SUB R1, R0, R1			

2. Show the contents of the registers and memory locations as each instru

	R0	R1	Memory Location
		3	2
LDR R0, 100			
ADD R0, R0, R1			
STR R0, 101			

3. Show the contents of the registers and memory locations as each instru

	R0	R1	Memory Location
	20		5
LDR R1, 100			
ADD R0, R0, R1			
LSL R1, R0, #2			
STR R1, 101			

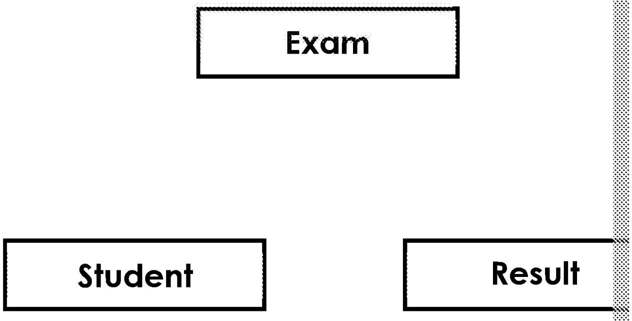
Relational Databases

An exam board uses a database to store the results of each exam for each s
database has three tables: Student, Exam and Result.

1. Details of the three tables are displayed below. Underline the correct fie
for each table. (3)

```
Student (StudentID, Forename, Surname, School, Gender, Da
Exam (ExamID, ExamName, Subject, Level)
Result (ExamID, StudentID, RawMark, Grade)
```

2. Draw the correct relationships between each of the tables. (2)



3. Which part of a database stores all the data about one object or person

4. Underline the foreign keys used in the database. (2)

```
Student (StudentID, Forename, Surname, School, Gender, Da
Exam (ExamID, ExamName, Subject, Level)
Result (ExamID, StudentID, RawMark, Grade)
```

Structured Query Language

An exam board uses a database to store the results of each exam for each s database has three tables: Student, Exam and Result.

```
Student (StudentID, Forename, Surname, School, Gender, DateOf
Exam (ExamID, ExamName, Subject, Level)
Result (ExamID, StudentID, RawMark, Grade)
```

1. Student number 10 has completed exam number 22 with a raw mark o SQL commands to make this entry. (2)

2. Student number 6 has moved to Camden School. Write the SQL comma the student's entry. (3)

3. Student number 8 was entered by mistake. Write the SQL commands to

4. The exam board want to produce a list of all results with an A grade. Th appear for each result: ExamID, StudentID, Grade. Write an SQL query to

5. The exam board want to produce a list of all students that achieved a m
number 17 sorted by raw mark in ascending order. The following inform
student: StudentID, Forename, Surname, School, RawMark. Write an SQ

6. Write the SQL instruction to create the Result table. (4)

Normalisation

1. The following data is non-atomic; convert it into atomic data. (3)

   `Lesson: Computing Tuesday 12:30`

2. State whether or not the following table is in 1NF and give your reasons

MemberID	Name	Telephone no.
001	[illegible]	020 7625 1234 07345 678910

3. Shown below is the structure for a database for an airline. Use the tables database in 2NF. (2)

FlightID*	DestinationID*	DestinationCountry	DepartureTime	A

Flight

Destination

4. Shown below is the structure for a database for an online store. Use the database in 3NF. (2)

OrderID*	ProductID	ProductName	Price	Quantity

Order

Product

Answers

Sequence and Selection

1. Program A
2. *Any line from Program A*
3. A, B and Average
4. Program B
5. 02 and 04 (from program B)
6. *1 mark per correct row:*

Operator	Meaning
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
[illegible]	Equal to
[illegible]	Not equal to

Iteration

1. Program B
2. Line 02
3. Program A
4. It asks the user to input a number (1) and uses a FOR loop (1) to output th
   the user inputted (1).
5. The value of the password variable is set to 'none' (1). A WHILE loop is use
   input their password (1) until they enter the value 'turing' at which point i
6. In a WHILE loop the condition is tested at the start, whereas in a REPEAT U
   (1). A WHILE loop repeats until the condition is TRUE, whereas a REPEAT U
   condition is FALSE (1). In a WHILE loop the statements may never be exec
   loop the statements will always be executed at least once (1).
7. *Award 1 mark for each of the following (also refer to the example below):*
   - The user is asked to input a number.
   - The counter (i) has been initialised correctly (value = 1) before the l
   - The condition is correct.

   Example:

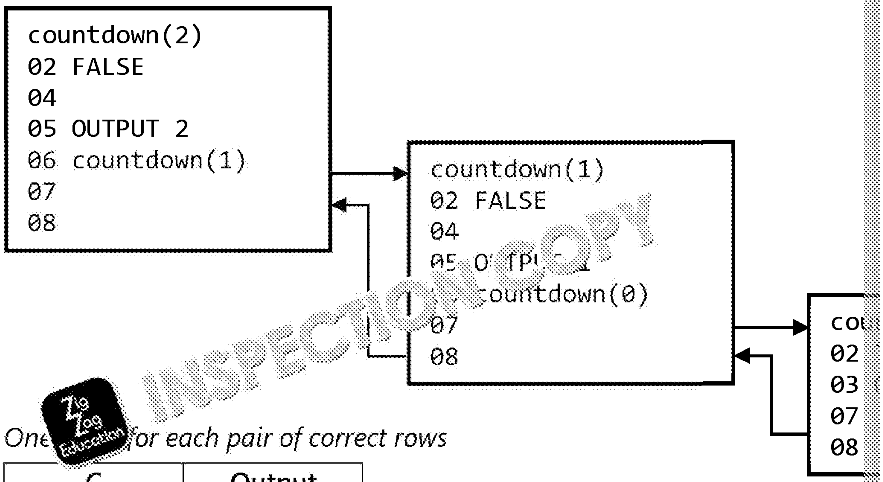
```
INPUT Num
i = 1
WHILE i <= Num
    OUTPUT i
    i = i + 1
END WHILE
```

Subroutines

1. Program B
2. A, B and C
3. 10 and 8
4. 6 and 4
5. 80
6. 6
7. A procedure is a subroutine that does not normally return a value (1), whe
   returns values to the calling routine (1).

Recursion

1. A
2. 06
3. A and B
4. *One mark for each correctly completed box and one mark for each correctly*



```
countdown(2)
02 FALSE
04
05 OUTPUT 2
06 countdown(1)
07
08
```

```
countdown(1)
02 FALSE
04
05 OUTPUT 1
06 countdown(0)
07
08
```

```
cou
02
03 (
07
08
```

5. *One mark for each pair of correct rows*

C	Output
3	
	3
2	
	2
1	
	1
0	
	0

6. There are many calls to the same subroutine (1), each with its own set of v up more memory (1).

Arrays

1. Steven
2. Ian
3. `names[4]`
4. `names[6]`
5. *1 mark initialising the names array correctly, 1 mark for using " " around th*

```
names ← [“Susan”, “Ian”, “Barbara”, “Steven”, ”Ben”, ”Polly
```

6. *1 mark for loop with correct range, 1 mark for using a counter variable (i) value from the names array, 1 mark for correct indentation*

```
FOR i ← 0 to 6
    OUTPUT names[
NEXT i
```

7. 58
8. 57
9. scores[0][1]
10. Scores[3][2]
11. *1 mark for initialising the scores array, 1 mark for initialising it as a 2D arra*

```
scores ← [[45, 71, 34, 55],[23, 82, 57, 37],[18, 31, 53, 5
```

Stacks and Queues

1. *One mark for correct top value, one for correct bottom value and 2 marks f*

Index	Value
4	Ben
3	Zoe
2	Jamie
1	Polly

2. *One mark for the removal of the correct value*

Index	Value
4	
3	Zoe
2	Jamie
1	Polly

3. 3
4. *One mark for each correct row, one for each correctly positioned pointer*

[illegible]	Value	
4	Polly	← Front
3	Jamie	
2	Zoe	
1	Ben	← Rear

5. *One mark for the removal of the correct value and one mark for the correct*

Index	Value	
4		
3	Jamie	← Front
2	Zoe	
1	Ben	← Rear

6. *One mark for the addition of the new value in the correct position and one Rear pointer*

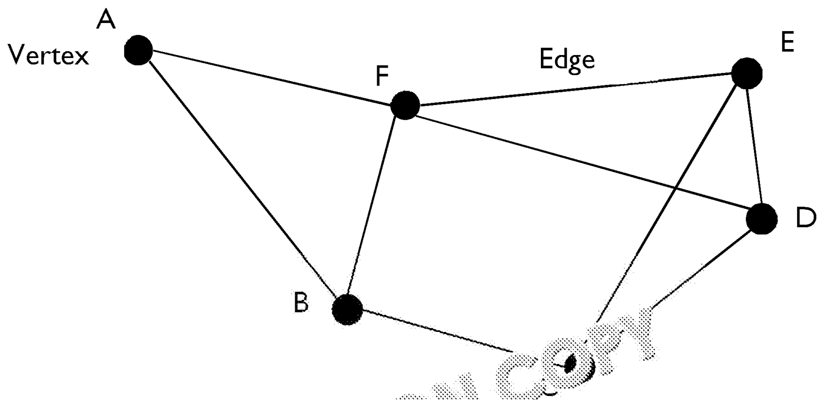
Index	Value	
4	Mel	← Rear
3	Jamie	← Front
2	Zoe	
1	Ben	

Linked Lists and Hash Tables

1. When an array is created its size is declared and it is allocated a section o fixed (1). Linked lists consist of nodes and each node can be stored in any f size flexible (1).
2. Each node in a linked list contains a pointer to the memory location of the node in the list is indicated using the Ø symbol (1). New nodes can easily b the pointers (1)
3. A disadvantage of linked lists is elements can't be directly accessed. The e the element (1). Hash tables offer a solution to this problem, being both f access to elements (1).
4. Hash tables consist of two parts: an array with an associated hash functio piece of data known as a key and generates a hash value; this is used as t
5. The value 'David' should be written in position 3.
6. The value 'Diane' should be written in position 4.

Graphs and Trees

1. *Any edge and vertex correctly labelled; for example:*



2. *Any suitable example, for example [illegible] and F*
3. 4
4. *One mark for each correct line:*

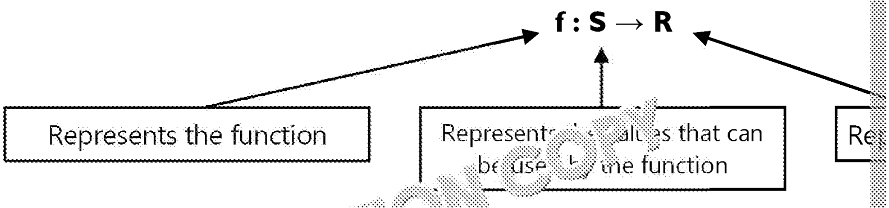
	A	B	C	D	E
A	0	1	0	0	1
B	1	0	1	1	1
C	0	1	0	1	0
D	0	1	1	0	1
E	1	1	0	1	0

5. *One mark for each correct line:*

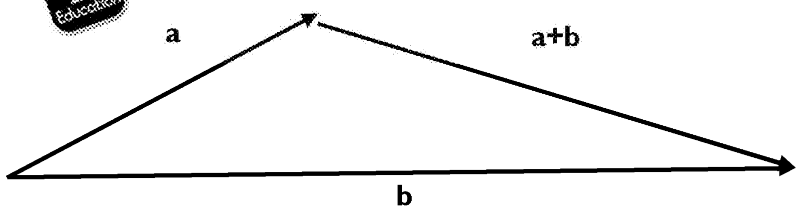
Vertex	Adjacent Vertices
A	B, E
B	A, E, D, C
C	B, D
D	C, B, E
E	A, B, D

Vectors

1. Arrays are fixed in size (1), whereas vectors can grow (1).
2. *One mark per correct label:*



3. *One mark for the line starting at 0,0 and one mark for it ending at 5,2*
4. *One mark for the correct line, one mark for the correct label*



a = (4, 5) b = (8,1)

5. *One mark for correct answer, one mark for working*
(4*8) + (5*1) = 37

Graph and Tree Traversal

1. 1, 2, 4, 5, 3, 6
2. 4, 5, 2, 6, 3, 1
3. 4, 5, 2, 1, 6, 3
4. *Open mark for each correct line where a vertex is visited:*

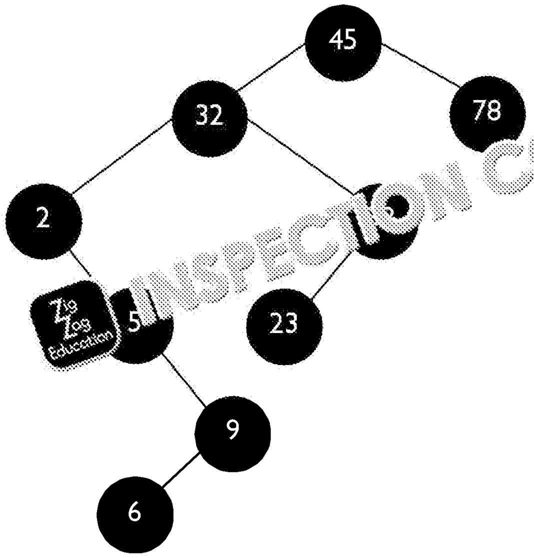
Vertex Visited	Stack
A	A
G	GA
D	DGA
E	EDGA
F	FEDGA
	EDGA
	DGA
B	BDGA
C	CBDGA

Reverse Polish Notation

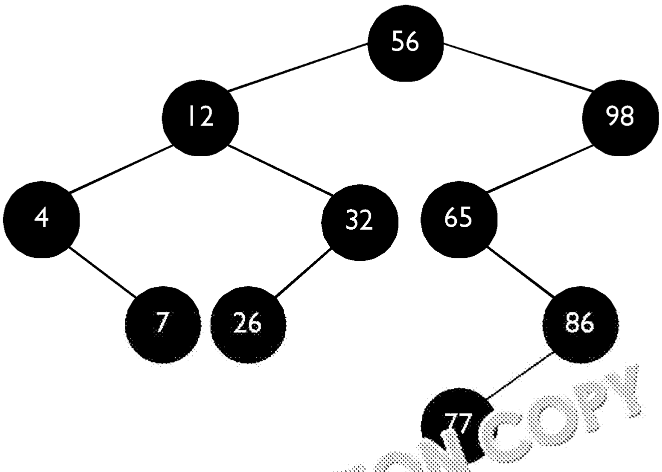
1. 4 + 6
2. 8 - 2
3. (4 + 9) * 6
4. 10 * (4 + 7)
5. (3 + 9) * (4 - 2)
6. 4 3 +
7. 56 40 -
8. 5 5 3 - *
9. 6 3 / 6 2 + +
10. 18 8 - 30 20 + *

Searching Algorithms

1. Searches each element starting at the start of the list (1) until it finds a ma
2. 1
3. *One mark for each correct pair of levels:*



4. *One mark for correct first two levels, one mark for the third level, one mark*



5. 2
6. 3

Sorting Algorithms

1. *One mark per correct row*

45	32	2	78	
32	45	2	78	
32	2	45	78	
32	2	45	5	
32	2	45	5	
2	32	45	5	
2	32	5	45	
2	32	5	9	
2	5	32	9	
2	5	9	32	

2. *One mark per correct row*

45	32	2	78	5	9
32, 45		2, 78		5, 9	
2, 32, 45, 78				5,	
2, 5, 9, 23, 32, 38, 45, 78					

Shortest Path Algorithm

1. *One mark per correct row*

Node	Shortest Distance from vertex A	Previous Node
A	0	
B	4	A
C	[illegible]	B
D	15	F
E	17	H
F	11	G
G	8	B
H	5	A

2. A – B – G
3. A – H – G

Big O Notation

1. *One mark for each correct cell*

Complexity	Description
Constant	The time complexity remains the same regardless of the n
Linear	The time complexity is proportional to the number of iter
Polynomial	The rate at which time complexity rises increases as the n
Exponential	The time complexity increases exponentially as the numb
Logarithmic	The increase in time complexity decreases as the number

2. Linear search
3. Bubble sort
4. Binary search and binary tree search
5. *One mark per correct row:*

O(1)	1
O(n)	3
O(k^n)	5
O(log (n))	2
O(n^k)	4

Tracing an Algorithm

1. *Award 1 mark for each correct line:*

Line	A	B	OUTPUT	Comment
01	1			
02		5		
03				FALSE, go to line 05
05				
06			5	
07				

2. *Award 1 mark for each correct line:*

Line	C	OUTPUT	Comment
01	0		
02			TRUE, go to line 03
03		0	
04	1		
05			Return to line 02
02			TRUE, go to line 03
03		1	
04	2		
05			Return to line 02
02			TRUE, got to line 03
03		2	
04	3		
05			Return to line 02
02			FALSE, end algorithm

Finite-State Machines

1. Invalid
2. Valid
3. *Any valid example – for example, baab*
4. *Any valid example – for example, babb*
5. *1 mark per correct row:*

Current State	Input	Next State
$_s1$	b	$_s1$
$_s1$	a	$_s2$
$_s2$	a	$_s1$
$_s2$	b	$_s3$
$_s3$	a	[illegible]
$_s3$	[illegible]	$_s2$

6. *10[illegible]*

Maths for Regular Expressions

1. A = { } or A = Ø
2. 6
3. y represents all integers (1) greater than or equal to –3 (1)
4. Infinite set
5. {(x,1),(x,2),(x,3),(y,1),(y,2),(y,3),(z,1),(z,2),(z,3)}
6. Proper subset
7. *One mark per correct row*

Joining sets together to form a new set keeping only unique elements fr
Joining sets together to form a new set containing elements from both
Joining sets together to form a new set keeping only the elements that ar

Regular Expressions

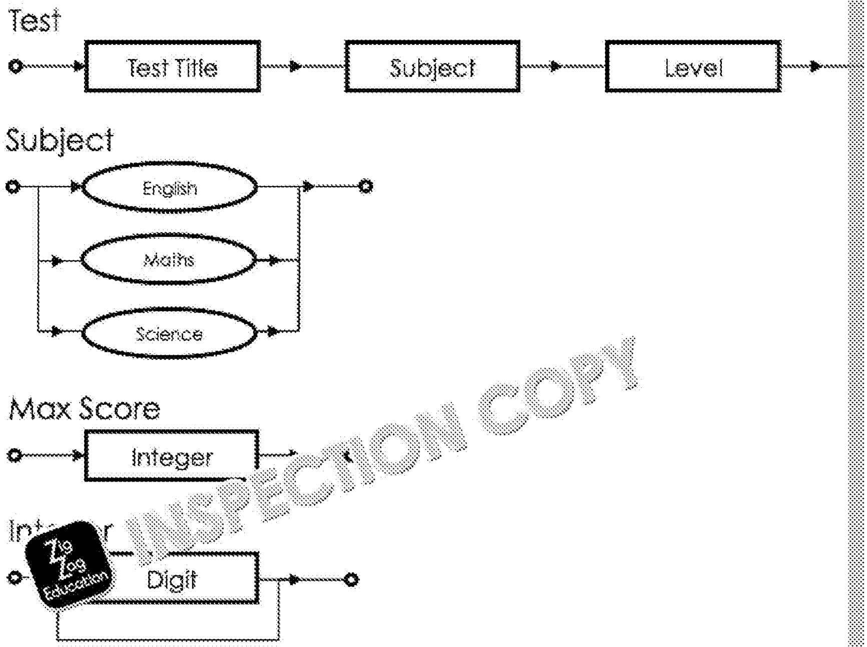
1. 1, 3, 4
2. 0 or more as (1) followed by 1 or more bs (1) followed by 1 or 0 cs (1), fol
3. There should be at least one b in string 1 (1); there should be either 0 or 1
4. (a|b)(ac)*a* 1 mark for (a|b) 1 mark for (ac)* 1 mark fo

Backus–Naur Form

1. <level> ::= KS3 | KS4 | KS5
2. *One mark per row*

Test Details		Accepted (Y/N
Test Title: Networking Level: KS4	Subject: Computing Max Score: 50	N
Test Title: Macbeth Level: KS4	Subject: English Max Score: 25	Y
Test Title: States of Matter Level: KS3	Subject: Science Max Score: 10	Y
Test Title: Fractions Level: KS4	Subject: Maths Max Score: 40.5	N
Test Title: Percentages Level: KS4	Subject: Maths Max Score: 25	Y

3. *2 marks for each correct part of the diagram (Test, Subject, Max Score, Integ*



Turing Machine

1. *One mark for each correct row (doesn't have to be in the same order):*

State	Read	Write	Move Direc
A	0	1	R
A	1	1	R
B	0	1	R
B	1	1	R

2. *One mark for each correct line:*
δ (A, 0) = (B, 1, R)
δ (A, 1) = (A, 1, R)
δ (B, 0) = (H, 1, R)
δ (B, 1) = (B, 1, R)

3. *One mark for each correct state of the tape (excluding first row) + one mar positions*

1	0	□	1	1
↑				
1	0	□	1	1
	↑			
1	1	□	1	1
		↑		
1	1	1	1	1
			↑	
1	1	□	0	1
				↑
1	1	□	0	0

Binary and Hexadecimal

1. 00101101 *(1 mark for the correct answer and 1 mark for working)*
2. 101 *(1 mark for the correct answer and 1 mark for working)*
3. A5 *(1 mark for the correct answer and 1 mark for working)*
4. 182 *(1 mark for the correct answer and 1 mark for working)*
5. 10011011 *(1 mark for the correct answer and 1 mark for working)*

6. 92 *(1 mark for the correct answer and 1 mark for working)*

Binary Arithmetic

(1 mark for the correct answer and 1 mark for working)

1. 01100
2. 01100
3. 010010
4. 01110
5. 010110
6. 010001
7. 110001
8. 1011011
9. 10100101
10. 110111
11. 1100011
12. 111000

Two's Complement

(1 mark for the correct answer and 1 [illegible] working)

1. -74
2. 50
3. 1000[illegible]10
4. 01001000
5. 10011110
6. 0010
7. 0110
8. 1100

Fractions

1. 11.5 *(1 mark for the correct answer and 1 mark for working)*
2. 0111.1000 *(1 mark for the correct answer and 1 mark for working)*
3. 9.75 *(1 mark for the correct answer and 1 mark for working)*
4. 1100.1010 *(1 mark for the correct answer and 1 mark for working)*
5. -2.5 *(1 mark for the correct answer and 1 mark for working)*
6. 13.5 *(1 mark for the correct answer and 1 mark for working)*
7. Mantissa: 1.0111000 Exponent: 0011
8. Absolute: 0.02 Relative: 0.001158
9. Mantissa: 0.110110 Exponent: 0101
10. Mantissa: 0.110 Exponent: 0001

Error Checking

1. *1 mark for each correct row:*

0	0101110
0	[illegible]10[illegible]
1	[illegible]110

2. *1 mark for each correct row:*

1	0101110
1	1101001
0	0010110

3. 01001111 *(the third option)*
4. *1 mark for each correct row:*

Received Bits	Decoded Bit
110	1
010	0
100	0

5. *1 mark for each correct row:*

Method	Description
Checksum	Calculated by adding together all the bytes in the packet a
Check digit	Used to validate data entry, added to the end of a number, and

6. ISBN, credit card number or any other valid example
7. 6 *(1 mark for the correct answer and 1 mark for working)*

Bitmapped Images

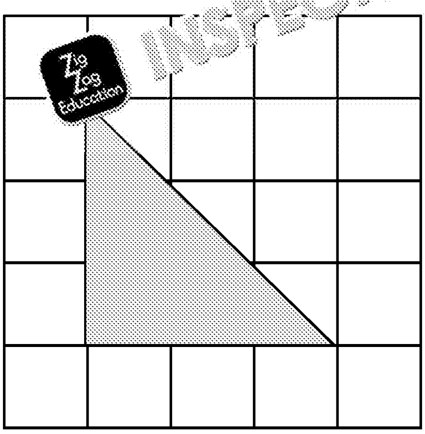
1. *1 mark for each correct row:*

			Grey				→	00 00 00 01 00 00 00
			Grey				→	00 00 00 01 00 00 00
Green	Green	Green	Blue	Green	Green	Green	→	10 10 10 11 10 10 10 *or*
			Grey				→	00 00 00 01 00 00 00
			Grey				→	00 00 00 01 00 00 00

2. 7 × 5
3. 35
4. 2 bits
5. (35 × 2) / 8 (1)
   1.75 bytes (1)
6. *Any one of:* width, height, colour depth, geolocation

Vector Graphics

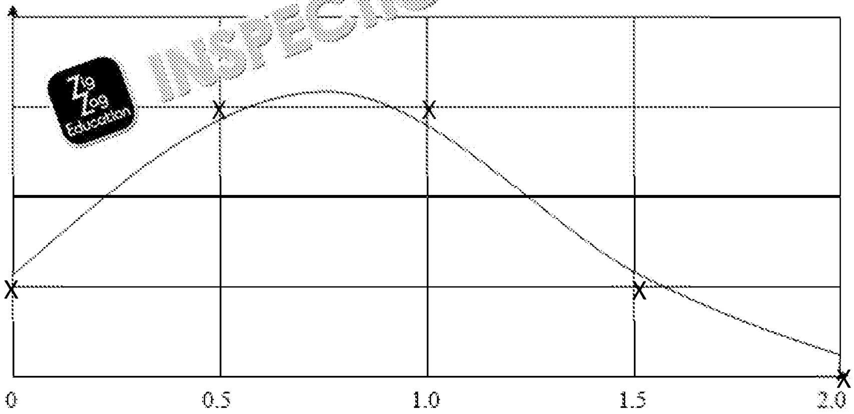
1. *One mark for each correctly drawn property (object type, coordinates, fill c*



2. *Any 2 from:* edge colour, edge width, edge style
3. *Any 2 from:*
   Geometric images stored as vectors require less storage space compared
   Geometric images stored as vectors will load faster from secondary storag
   Vector graphics can be resized without distorting
4. *Any 2 from:*
   Only suitable for images made up of geometric shapes
   Unsuitable to represent photographs as colours can vary greatly within th
   Complex images can take a long time to render
5. Using lists

Digital Sound

1. *An x marked every 0.5 seconds (1) and each one on the closest horizontal li*



2. 44,000 × 16 × 2 × 300 = 422,400,000 bits (1) ÷ 8 = 52,800,000 bytes (1)
3. 600 × 1024 × 8 = 4,915,200 bits (1) ÷ 2 ÷ 8 ÷16000 (1) = 19.2 seconds
4. 15 kHz (1) because Nyquist's theorem says that the sampling rate has to b
   the highest frequency in the sound file (1)

Data Compression

1. *1 mark for each row:*

1	1	0	1	1	→	2(1), 1(0), 2(1)
1	1	0	1	1	→	2(1), 1(0), 2(1)
0	0	0	0	0	→	5(0)
1	1	0	1	1	→	2(1), 1(0), 2(1)
1	1	0	1	1	→	2(1), 1(0), 2(1)

2. 5(C), 5(L), 3(Q), 4(P)
3. *1 mark for each row, accept any two suitable values:*

1	1	0	1	1	→	00
1	1	0	1	1	→	00
0	0	0	0	0	→	01
1	1	0	1	1	→	00
1	1	0	1	1	→	00

4. Advantage: lossy compression usually results in significantly smaller files.
   Disadvantage: permanently destroys some of the data.

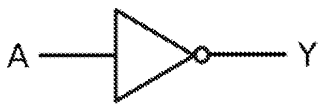
Encryption

1. EQORWVGT UEKGPEG TQEMU
2. LEMP GEIWEV
3. SECURE
4. BLETCHLEY
5. *1 mark for each correct column:*

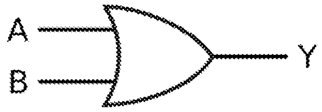
C			O		
Plaintext	**OTP**	**Ciphertext**	**Plaintext**	**OTP**	**Ciphertext**
1	1	0	1	1	0
0	0	0	0	0	0
0	0	0	0	0	0
0	1	1	1	1	0
0	0	[illegible]	1	0	1
1	1	[illegible]	1	0	1
[illegible]	[illegible]	0	1	1	0

Logic Gates

1. *AND gate:*



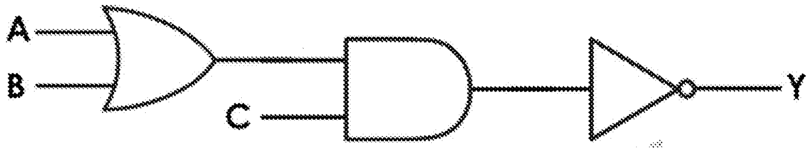
OR gate:



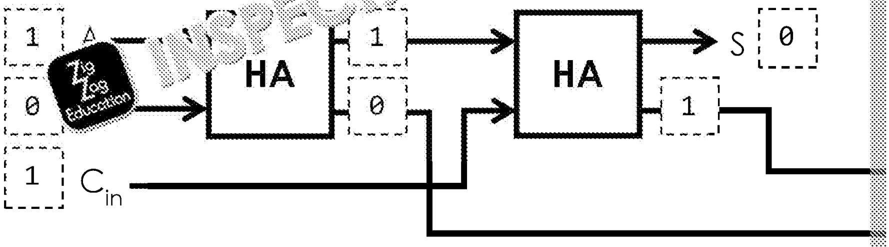
2. *1 mark for each correct line:*

Input A	Input B	Output Q
0	0	0
0	1	1
1	0	1
1	1	1

3. *1 mark for each correctly placed symbol:*



4. *1 mark for each correctly labelled output:*



Boolean Algebra

1. $A.A + A.\overline{B} + B.A + B.\overline{B}$ *(1 mark for expanding the brackets)*
   $A + A.B + \overline{B}.A$
   $A + A$
   A *(1 mark for removing the redundant terms)*

2. The first two expressions simplify to P.R as the first features Q and the sec
   $\overline{P}.R + P.R$ R is the same in both, and the $\overline{P}$/P become redundant
   Simplest form therefore is R *(1 mark for the answer)*

3. A.1 *(1 mark)*   A.A *(1 mark)*

4. $\overline{A.B} = \overline{A + B}$ *(1 mark)*

5. $\overline{(X + Y)} + \overline{(Y + Z)}$ *(1 mark for initial use of De Morgan)*
   $(\overline{X} . \overline{Y}) + (\overline{Y} . \overline{Z})$ *(1 mark for repeated use of De Morgan)*
   $\overline{Y}.(\overline{Y}$ [illegible] *(1 mark for final answer)*

Assembly Language

1. *1 mark for each correct line:*

	R0	R1	R2
		5	3
MOV R0, #9	9		
ADD R0, R0, R2	12		
SUB R1, R0, R1		7	

2. *1 mark for each correct line:*

	R0	R1	Memory Location 100	Memory Locat
		3	2	
LDR R0, 100	2			
ADD R0, R0, R1	5			
STR R0, 101				5

3. *1 mark for each correct line:*

	R0	R1	Memory Location 100	Memory Loca
	20			
LDR R1, 100		5		
ADD R0, R0, R1	25			
LSL R1, R0, 2		100		
STR R1, 101				100

Relational Databases

1. *One mark for each primary key*

```
Student (StudentID, Forename, Surname, School, Gender, Date
Exam (ExamID, ExamName, Subject, Level)
Result (ExamID, StudentID, RawMark, Grade)
```

2. *One mark for each correct relationship*

3. Record

4. *One mark for each correctly identified foreign key*

```
Student (StudentID, Forename, Surname, School, Gender, Date
Exam (ExamID, ExamName, Subject, Level)
Result (ExamID, StudentID, RawMark, Grade)
```

Structured Query Language

1. *One mark for each correct statement*

```
INSERT INTO Result
VALUES (22, 10, 67, "A")
```

2. *One mark for each correct statement*

```
UPDATE Student
SET School = "Camden School"
WHERE StudentID = 6
```

3. *One mark for each correct statement*

```
DELETE FROM Student
WHERE StudentID = 8
```

4. *One mark for each correct statement*

```
SELECT ExamID, StudentID, Grade
FROM Result
WHERE Grade = "A"
```

5. *One mark for each correct statement*

```
SELECT StudentID, Forename, Surname, School, RawMark
FROM Student, Result
WHERE Student.StudentID = Result.StudentID
AND RawMark >= 50 AND ExamID = 17
ORDER BY RawMark
```

6. *One mark for each correctly defined field*

```
CREATE TABLE (
ResultID INT PRIMARY KEY NOT NULL
StudentID INT
RawMark INT
Grade VARCHAR(2) or Grade VARCHAR(1)
)
```

Database Normalisation

1. *One mark for each correctly identified field:*
   Subject: Computing, Day: Tuesday, Time: 12:30

2. No (1) because there are two telephone numbers in the telephone field (1) and there must only be one value per field (1)

3. Flight

FlightID*	**DestinationID***	**DepartureTime**	**ArrivalTime**

Destinations

DestinationID*	**DestinationCountry**

4. Order

OrderID*	**ProductID**	**Quantity**	**TotalPrice**

Product

ProductID	**ProductName**	**Price**