GCST

1.3.1 COMPRESSION, ENCRYPTION AND HASHING

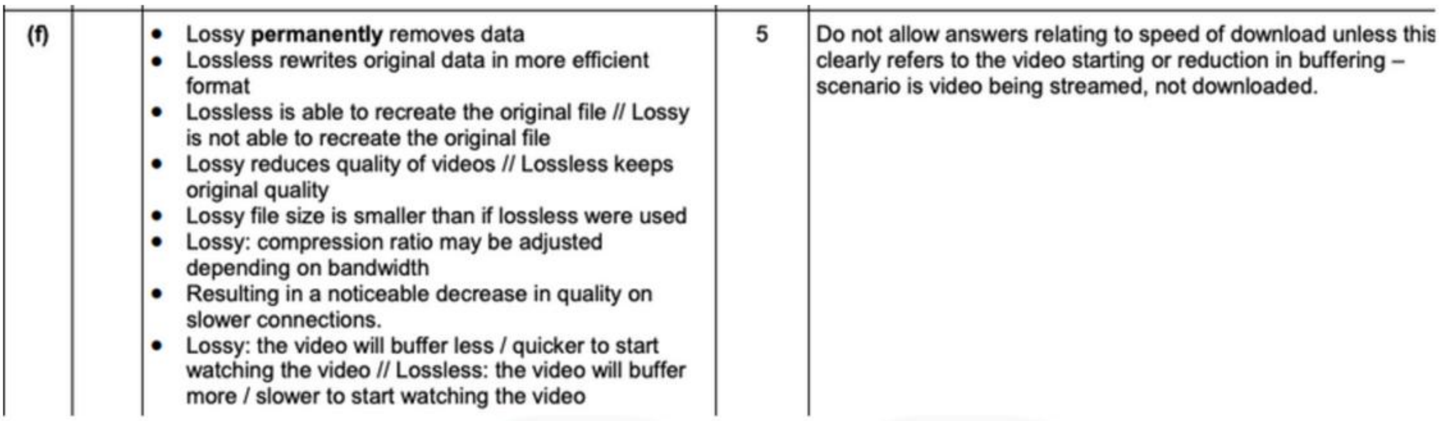
TOPIC WISE EXAM QUESTIONS

ANSWERS

A-LEVEL OCR

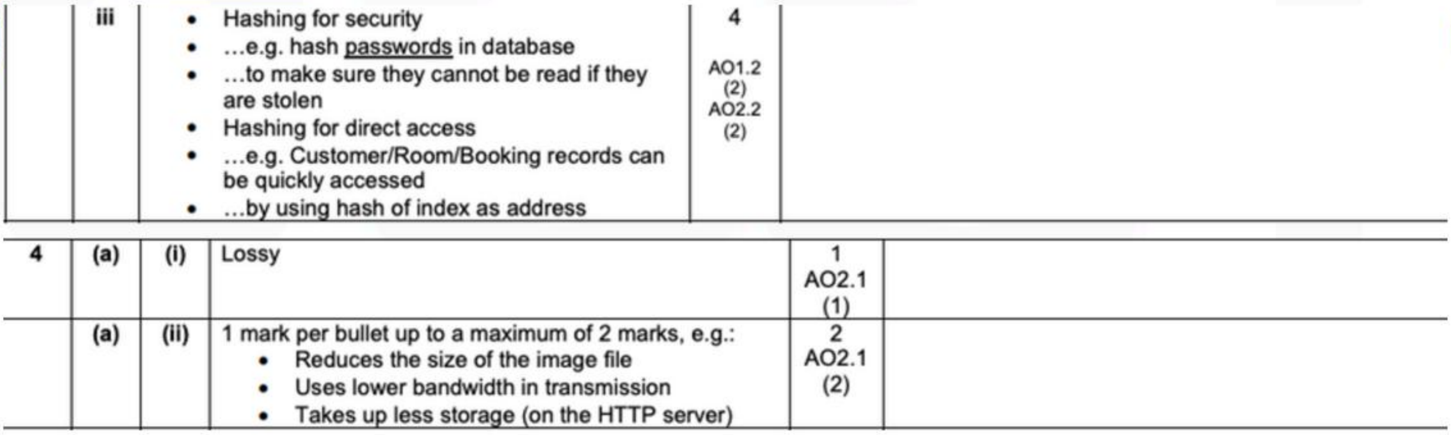
4*	(c)		**Mark Band 3 – High Level (7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of encryption and hashing and how they can be used to store data and communicate securely. The material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate is able to weigh up both technologies which results in a supported and realistic judgement covering when each can be used. This is well balanced. There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated. **Mark Band 2 – Mid Level (4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of encryption **and** hashing and how they can be used to store data and communicate securely; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence / examples are for the most part implicitly relevant to the explanation. The candidate makes a reasonable attempt to come to a conclusion showing some recognition of either technology. This may not be well-balanced, covering one side significantly more than the other, There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence. **Mark Band 1 – Low Level (1-3 marks)** The candidate demonstrates a basic knowledge of encryption or/and hashing and how they can be used to store data and communicate securely; the material is basic and contains some inaccuracies. The candidate makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides nothing more than unsupported assertions. Any discussion will be almost entirely one-sided. The information is basic and communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear. **0 marks** No attempt to answer the question or response is not worthy of credit.	*The following shows example content that may form par of a candidate's answer. It is not intended to be an exhaustive resource, nor should a candidate be expected to specifically cover any particular amount of this.* **Knowledge (AO1)** • Encryption converts data into data that cannot be understood (ciphertext) using a key. • Symmetric encryption uses the same key for both encryption and decryption • Asymmetric encryption uses two keys, one for encryption, one for decryption • Encryption is two-way, so data can be restored to original form, but key is required. • Hashing is a one-way (non-reversible) mathematical process that produces a value from the input value. **Application (AO2)** • For robot's data storage, symmetric encryption is useful as no keys to share/transmit. • For robot-robot/user communication, asymmetric encryption / public key encryption means that only the public key needs to be shared. Data can be encrypted/decrypted with this while the private key is kept secure • Also possible to verify identity of sender / origin of data using asymmetric encryption. • Hashing is useful for information (e.g. password) tha needs to be verified but does not need to be known at any point; once hashed, it is impossible to return to it. **Evaluation (AO3)** • Encryption useful for most data storage as anyone hacking into the robot will not be able to read/understand the data. • Hashing is useful for data storage of password / other items that need to be verified, hash of input compared against hash stored to confirm correctness. • Hashing is not useful for data that needs to be returned to the user as impossible to return to. • Encryption useful for data transmission as data intercepted cannot be decrypted without the key.

2022



(f)		• Lossy **permanently** removes data • Lossless rewrites original data in more efficient format • Lossless is able to recreate the original file // Lossy is not able to recreate the original file • Lossy reduces quality of videos // Lossless keeps original quality • Lossy file size is smaller than if lossless were used • Lossy: compression ratio may be adjusted depending on bandwidth • Resulting in a noticeable decrease in quality on slower connections. • Lossy: the video will buffer less / quicker to start watching the video // Lossless: the video will buffer more / slower to start watching the video	5	Do not allow answers relating to speed of download unless this clearly refers to the video starting or reduction in buffering – scenario is video being streamed, not downloaded.

2020



	iii	• Hashing for security • …e.g. hash passwords in database • …to make sure they cannot be read if they are stolen • Hashing for direct access • …e.g. Customer/Room/Booking records can be quickly accessed • …by using hash of index as address	4 AO1.2 (2) AO2.2 (2)	

4	(a)	(i)	Lossy	1 AO2.1 (1)	
	(a)	(ii)	1 mark per bullet up to a maximum of 2 marks, e.g.: • Reduces the size of the image file • Uses lower bandwidth in transmission • Takes up less storage (on the HTTP server)	2 AO2.1 (2)	

2	a		(The process of) making a file smaller/take up less storage	1 AO1.1	
	b		Full answer CCCMMMCCCC – CCC – ... followed by MMMCCCC (1 per -, max 2)	2 AO1.2	
	c		4C1O3L5C1M1O5C – 4C1O – Followed by 3L5C – Followed by 1M1O5C	3 AO1.2	Accept answer without 1s

	d		– Correct function name and parameter AND the function returns a value.
			– Use of a loop to correctly iterate through the sequence
			– Adds one to a running total when a C is encountered
			– -when character changes from a C if running total is > maximum, overwrites maximum…
			– …correctly reset running total
			1 mark per -, max 5
				5
				AO3.2
					E.g.
					function longest(sequence)
					currentRun = 0
					biggestRun = 0
					for i = 0 To sequence.length - 1
					if sequence.substring(i, 1) == "C" then
					currentRun = currentRun + 1
					else
					if currentRun > biggestRun then
					biggestRun = currentRun
					end if
					currentRun = 0
					endif
					next i
					return biggestRun
					endfunction

	e		– High resolution videos take up large amounts of memory/RAM – Due to the large number of pixels that need to be represented – When streaming, the data being sent is time sensitive/ sufficient data (i.e. the next chunk of video) needs to be received and processed within a given amount of time – Otherwise there will be pauses/buffering. – Compression reduces the amount of data that needs to be sent/bandwidth needed – Compression reduces the cost/data usage for those with download limits	3 AO2.2	

b

Row shift as below (1 Mark)

P	S	E	T	O
E	T	M	C	R
S	A	G	E	S
R	P	L	E	Y
G	G	Q	U	O

Column Shift as below (1 Mark)

G	G	Q	U	O
P	S	E	T	O
E	T	M	C	R
S	A	G	E	S
R	P	L	E	Y

2
(AO1.2)

cao

c

- Procedure correctly defined with parameters.
- Procedure manipulates the correct row of grid.
- Sensible use of for loop to iterate through the array without generating out of bounds exception.
- Correctly shifts each row.

(1 Mark per -, Max 4)

4
(AO3.1)

When checking to see if out of bounds exception keep in mind that in some languages the loop boundaries are exclusive. When unsure give the benefit of the doubt. The final mark is meant to offer stretch and challenge. Be cautious of wrong answers on face value seems to work. For example, the following will **not** work:

```
procedure shiftRow(rowNumber, places)
    for i = 0 to places
        grid[rowNumber,i+1]= grid[rowNumber,i]
    next i
endprocedure
```

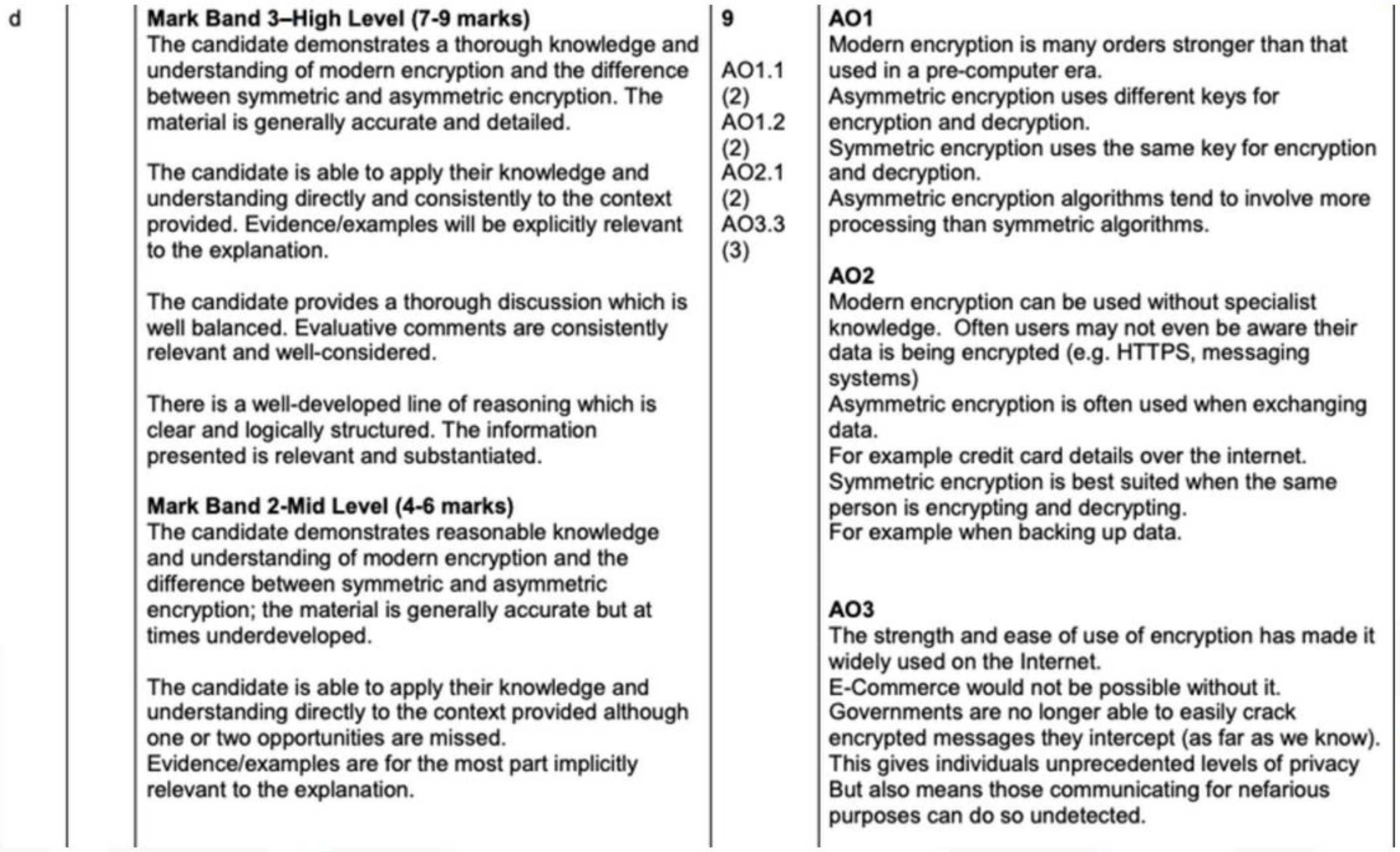
Possible solutions include…

```
procedure shiftRow(rowNumber, places)
    array temp[5]
    for i=0 to 4
        temp[i]=grid[rowNumber,i]
    next i
    for i=0 to 4
        newPos=(i+places)MOD 5 //% is the same
as MOD
        grid[rowNumber,newPos]=temp[i]
    next i
endprocedure
```

And..

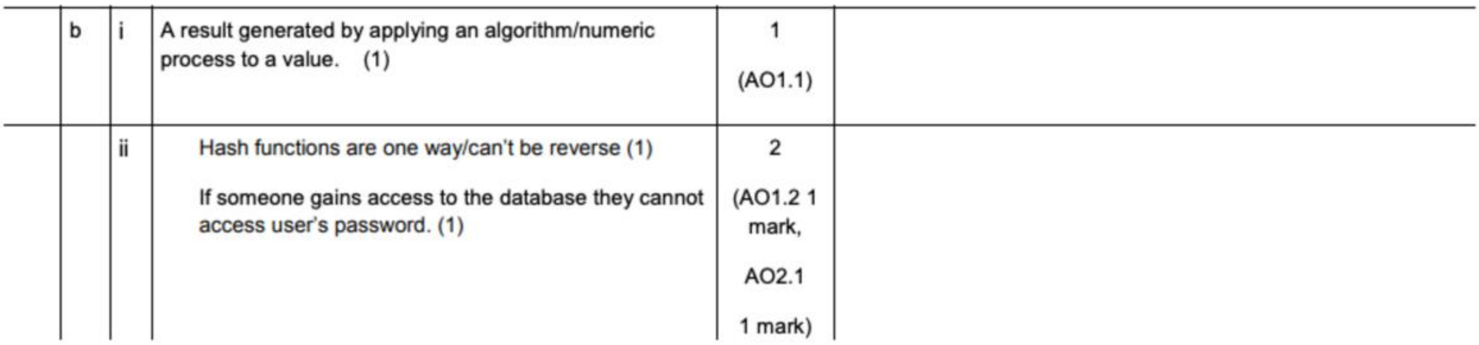
```
procedure shiftRow(rowNumber, places)
    for i=1 to places
        temp1=grid[rowNumber, 4]
        temp2=0
        for j =0 to 4
            temp2=grid[rowNumber,j]
            grid[rowNumber,j]=temp1
            temp1=temp2
        next j
   next i
end procedure
```

Note: within solutions, allow for columns to be referenced first eg `grid[i,rowNumber]`

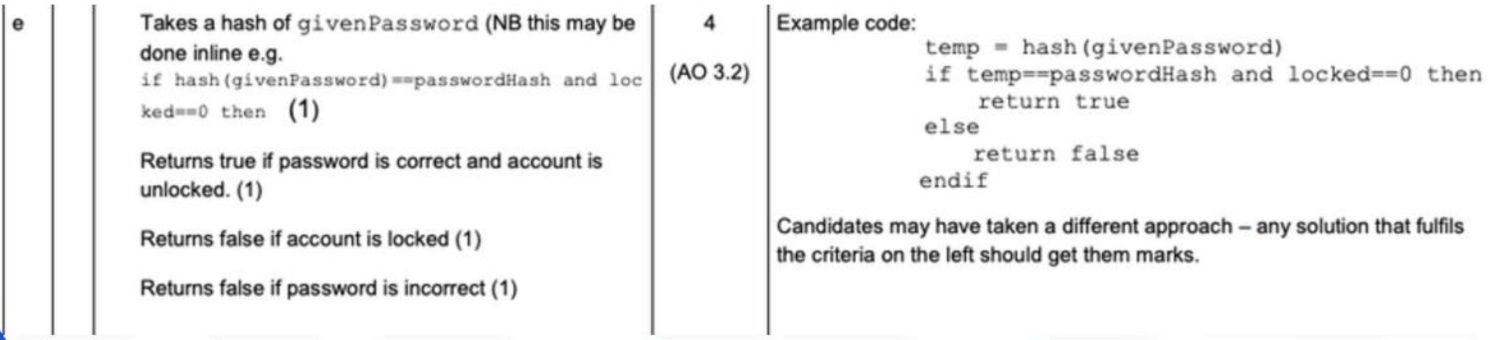


d		**Mark Band 3–High Level (7-9 marks)** The candidate demonstrates a thorough knowledge and understanding of modern encryption and the difference between symmetric and asymmetric encryption. The material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate provides a thorough discussion which is well balanced. Evaluative comments are consistently relevant and well-considered. There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated. **Mark Band 2-Mid Level (4-6 marks)** The candidate demonstrates reasonable knowledge and understanding of modern encryption and the difference between symmetric and asymmetric encryption; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation.	**9** AO1.1 (2) AO1.2 (2) AO2.1 (2) AO3.3 (3)	**AO1** Modern encryption is many orders stronger than that used in a pre-computer era. Asymmetric encryption uses different keys for encryption and decryption. Symmetric encryption uses the same key for encryption and decryption. Asymmetric encryption algorithms tend to involve more processing than symmetric algorithms. **AO2** Modern encryption can be used without specialist knowledge. Often users may not even be aware their data is being encrypted (e.g. HTTPS, messaging systems) Asymmetric encryption is often used when exchanging data. For example credit card details over the internet. Symmetric encryption is best suited when the same person is encrypting and decrypting. For example when backing up data. **AO3** The strength and ease of use of encryption has made it widely used on the Internet. E-Commerce would not be possible without it. Governments are no longer able to easily crack encrypted messages they intercept (as far as we know). This gives individuals unprecedented levels of privacy But also means those communicating for nefarious purposes can do so undetected.

3	a	i	Downloads quicker. (1) Saves user money by using less bandwidth/ on data usage. (1) (Max 1)	1 (AO1.2)	Do not accept 'saves the user space on their device'.
		Ii	Lossy takes away some of the information from the original. (1) Lossless preserves all the information from the original. (1) With text the loss of small amounts of information will make it unreadable. (1)	3 (AO1.1 – 2 marks AO2.1 - 1mark)	
	b		**Mark Band 3–High Level (9-12 marks)** The candidate demonstrates a thorough knowledge and understanding of dictionary and run length encoding for compression. The material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate is able to weigh up both forms of compression and justify dictionary encoding being the better choice. There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated. **Mark Band 2-Mid Level (5-8 marks)** The candidate demonstrates reasonable knowledge and understanding of dictionary and run length encoding for compression; the material is generally accurate but at times underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate makes a reasonable attempt to come to a conclusion as to which form of compression is better suited. There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence. **Mark Band 1-Low Level (1-4 marks)** The candidate demonstrates a basic knowledge of dictionary and run length encoding for compression; the material is basic and contains some inaccuracies. The candidate makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides nothing more than an unsupported assertion. **0 marks** No attempt to answer the question or response is not worthy of credit.	AO1.1 (2) AO1.2 (2) AO2.1 (3) AO3.3 (5) 12	Points may include but aren't limited to: **AO1 Knowledge and Understanding** Run length encoding relies on consecutive pieces of data/characters being the same. Each set of consecutive symbols can be represented by the symbol and its number of occurrences e.g. AAAABBBBBCCC could be represented as 4A5B3C (or A4B5C3 or any sensible RLE encoding) In dictionary encoding frequently occurring pieces of data/groups of characters are replaced by symbols/tokens/smaller groups of characters/indexes. A dictionary is then used to say which symbols/tokens/characters/indexes match which groups of characters. When decompressed the dictionary is used to replace the tokens with the original text. **AO2.1 Application** Run Length Encoding is very unsuitable for the example text There are very few consecutive repeating symbols in the text. only instances being ll and ee these still require 2 characters to represent them 2l and 2e Dictionary encoding is well suited. There are lots of repeating groups of characters For example 'call' 'name' '[SPACE]we' 'Romeo' We could for example have: What's in53? that which2 15 rose By5ny other3 would smell5s sweet; So4would,2re he not41'd 1:call 2:[space]we 3:[space]name 4:[space]Romeo[space] 5:[space]a (NB candidates are unlikely to show full compression, just a demonstration of the principle is sufficient. The best candidates are likely to show an awareness that space is a character that can be used in compression and that upper and lowercase letters are different. Demonstrating this is indicative of but not a requisite of the band.) **AO3.3: Evaluation** Run length encoding is not suited to natural language (more likely to be used in simple images). Applying it to the example the resulting text would be the same size as the original/worse than the original (if we use 1s to represent every individual instance of a character). Dictionary encoding works well. We can already see benefit on small piece of text. Would fare substantially better on full works. Dictionary encoding is the best compression method for this scenario.

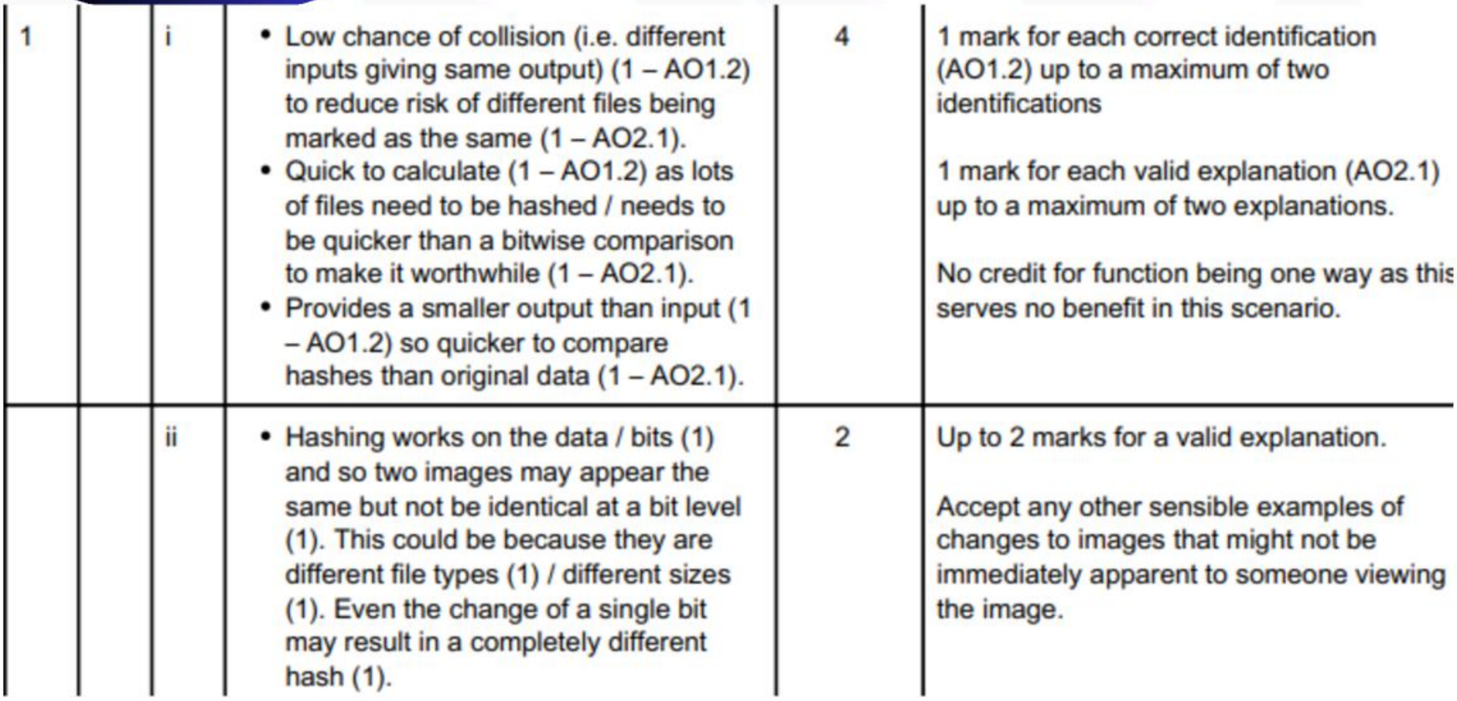


	b	i	A result generated by applying an algorithm/numeric process to a value. (1)	1 (AO1.1)	
		ii	Hash functions are one way/can't be reverse (1) If someone gains access to the database they cannot access user's password. (1)	2 (AO1.2 1 mark, AO2.1 1 mark)	

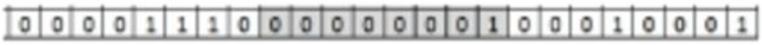
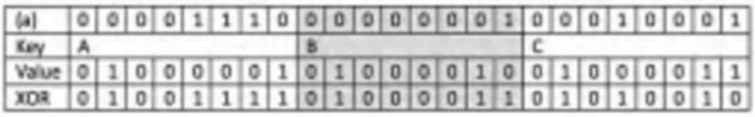


e			Takes a hash of `givenPassword` (NB this may be done inline e.g. `if hash(givenPassword)==passwordHash and locked==0 then` (1)
			Returns true if password is correct and account is unlocked. (1)
			Returns false if account is locked (1)
			Returns false if password is incorrect (1)
				4
				(AO 3.2)
					Example code:
					`temp = hash(givenPassword)`
					`if temp==passwordHash and locked==0 then`
					`    return true`
					`else`
					`    return false`
					`endif`
					Candidates may have taken a different approach – any solution that fulfils the criteria on the left should get them marks.

EXTRA



1		i	• Low chance of collision (i.e. different inputs giving same output) (1 – AO1.2) to reduce risk of different files being marked as the same (1 – AO2.1). • Quick to calculate (1 – AO1.2) as lots of files need to be hashed / needs to be quicker than a bitwise comparison to make it worthwhile (1 – AO2.1). • Provides a smaller output than input (1 – AO1.2) so quicker to compare hashes than original data (1 – AO2.1).	4	1 mark for each correct identification (AO1.2) up to a maximum of two identifications 1 mark for each valid explanation (AO2.1) up to a maximum of two explanations. No credit for function being one way as this serves no benefit in this scenario.
		ii	• Hashing works on the data / bits (1) and so two images may appear the same but not be identical at a bit level (1). This could be because they are different file types (1) / different sizes (1). Even the change of a single bit may result in a completely different hash (1).	2	Up to 2 marks for a valid explanation. Accept any other sensible examples of changes to images that might not be immediately apparent to someone viewing the image.

<table>
<tr><td>3</td><td>a</td><td></td><td>
<table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr></table>
One byte correct (1) all three bytes correct. (1)</td><td>2</td></tr>
<tr><td></td><td>b</td><td></td><td>
<table>
<tr><td>(a)</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>Key</td><td colspan="8">A</td><td colspan="8">B</td><td colspan="8">C</td></tr>
<tr><td>Value</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>XOR</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
</table>
One byte correct (1) all three bytes correct. (1)</td><td>2</td></tr>
<tr><td></td><td>c</td><td></td><td>Symmetric (1) …. as the same key is used to decrypt it as encrypt it (1)</td><td>2</td></tr>
<tr><td></td><td>d</td><td></td><td>Any four from:
Symmetric encryption would require both parties to have copy of the key (1) this couldn't be transmitted over the internet or an eavesdropper monitoring the message may see it (1) Asymmetric gets round this requirement as there are two different keys (1) One key encrypts the data (1) which can be publically distributed (1) and a different key to decrypt it (1) which is kept private (1)</td><td>4</td></tr>
</table>

If you found this useful, drop a follow to help me out!

THANK YOU!

GCST

GCSECOMPUTERSCIENCETUTOR.COM