1	1	**All marks for AO3 (programming)** **Mark as follows:** 1) Correct variable declarations for `Number`, `c`, `k;` **Note to examiners** If a language allows variables to be used without explicit declaration (eg Python) then this mark should be awarded if the correct variables exist in the program code and the first value they are assigned is of the correct data type. 2) `WHILE` loop with syntax allowed by the programming language and one correct condition for termination of the loop; 3) Second correct condition for while loop; 4) Correct prompt `"Enter a positive whole number: "` and `Number` assigned value entered by user; 5) Correct syntax for the `IF` statements inside attempt at loop; **A**. `IF … ELSEIF;` 6) correct contents in `IF` statements; 7) `FOR` loop with syntax allowed by the programming language over correct range; 8) Correct assignment to c inside `FOR` loop; 9) Output statement giving correct output; **A.** accept without spaces **I.** Ignore minor differences in case and spelling **R**. real `Number` **Max 8** if code does not function correctly	9

1	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from 03.1, including prompts on screen capture matching those in code.
		Code for 03.1 must be sensible.
		
		Screen capture showing:
		'-3' being entered and the message 'Not a positive number.' displayed
		'11' being entered and the message 'Number too large.' displayed
		'10' being entered and line of numbers displayed
		
		`Enter a positive whole number: -3`
		`Not a positive number.`
		`Enter a positive whole number: 11`
		`Number too large.`
		`Enter a positive whole number: 10`
		`  1  9  36  84  126  126  84  36  9  1`
		
		`>`
		
		A. Alternative layout :
		`Enter a positive whole number: -3`
		`Not a positive number.`
		`Enter a positive whole number: 11`
		`Number too large.`
		`Enter a positive whole number: 10`
		`1`
		`9`
		`36`
		`84`
		`126`
		`126`
		`84`
		`36`
		`9`
		`1`
		`>`
		
		A. input on new line
			1

2	1	**1 mark for AO3 (design) and 3 marks for AO3 (programming)** **Mark as follows:** **AO3 (design) – 1 mark:** 1) Identifying that a selection statement (or equivalent method) is required to test that character is within range of uppercase letters or is a space // identifying that selection statement needs modifying (e.g.`if Char in Letter...);` **AO3 (programming) – 3 marks:** 2) Selection structure is created with correct logic so that if error detected it ensures error message is displayed only once & subroutine exits; 3) calls `ReportError` subroutine with suitable message if error in input string; 4) final value of `MorseCodeString` set to `EMPTYSTRING` (accept `''` or `SPACE`) if error in input string; **A**. accept if `MorseCodeString` set to `EMPTYSTRING` initially and not changed.	4

2	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from 11.1, including prompts on screen capture matching those in code.
		Code for 11.1 must be sensible.
		
		Screen capture showing:
		'S' being entered followed by 'Help' and suitable message displayed
		
		`Main Menu`
		`=========`
		`R - Receive Morse code`
		`S - Send Morse code`
		`X - Exit program`
		
		`Enter your choice: S`
		`Enter your message (uppercase letters and spaces only):`
		`Help`
		`*  Invalid character entered  *`
		
		A. any suitable message, but must be within *s
			1

3	1	**1 mark for AO3 (design) and 6 marks for AO3 (programming)** **Mark as follows:** **AO3 (design) – 1 mark:** 1) Identifying that within an iterative statement a selection statement (or equivalent method) is required to test whether the Morse code is a dot, a dash or a space; **AO3 (programming) – 5 marks:** 2) Correct subroutine heading (`SendSignals`) and ending and correct parameter (`MorseCodeString`) ; 3) loop for each character in `MorseCodeString`; 4) start with empty string and keep adding a symbol string; 5) at least one conversion of dot, dash or space to the correct symbol string; 6) dot, dash and space converted to the correct symbol string; 7) output the signals correctly;	7

3	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from 12.1, including prompts on screen capture matching those in code.
		Code for 12.1 must be sensible.
		
		Screen capture showing:
		`S` being entered followed by `MORSE X`
		and the string **=== === === === === = === = = = = = = === = = ===** ‘ being displayed after the Morse code.
		
		`Main Menu`
		`=========`
		`R - Receive Morse code`
		`S - Send Morse code`
		`X - Exit program`
		
		`Enter your choice: S`
		`Enter your message (uppercase letters and spaces only):`
		`MORSE X`
		`-- --- .-. ... .  -..-`
		`=== ===  === === ===  = === =  = = =  =  === = = ===`
			1

4	1	**2 marks for AO3 (design) and 4 marks for AO3 (programming)** **Note** that AO3 (design) marks are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not regardless of whether the solution works.	6

Level	Description	Mark Range
3	A line of reasoning has been followed to arrive at a logically structured working or almost fully working programmed solution. Code is written to ensure that all letters are output with their corresponding Morse code. The formatting of each line has been considered. A formal interface is used to pass the data structures' data into the subroutine. All of the appropriate design decisions have been taken.	5-6
2	There is evidence that a line of reasoning has been partially followed. The formatting of each line does not fully comply with requirements. There is evidence of some appropriate design work. There is Morse code output for each letter.	3-4
1	An attempt has been made to create `OutputAlphabetWithCode` and some appropriate programming statements have been written. There is insufficient evidence to suggest that a line of reasoning has been followed or that the solution has been designed. The statements written may or may not be syntactically correct and the subroutine will have very little or none of the required functionality. It is unlikely that any of the key design elements of the task have been recognised.	1-2

Marking guidance:

Evidence of AO3 design – 2 points:

Evidence of design to look for in response:

1) identify the need for an iterative statement to act on each letter in turn
2) identify a method to output four letters per line

Evidence of AO3 programming – 7 points:
Evidence of programming to look for in response:

3) add option `A` to `DisplayMenu` subroutine
4) add test for new option and call `OutputAlphabetWithCode` with correct parameters
5) create new subroutine `OutputAlphabetWithCode` with correct parameters
6) loop from A to Z to output each letter and corresponding code separated from

		letter by one space (**A**. two spaces)	

4	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from 13.1, including prompts on screen capture matching those in code.
		Code for 13.1 must be sensible.
		
		Screen capture showing:
		main menu with new option `A`
		'`A`' being entered and alphabet with Morse codes displayed
		
		`Main Menu`
		`=========`
		`R - Receive Morse code`
		`S - Send Morse code`
		`A - Output alphabet with Morse code`
		`X - Exit program`
		
		`Enter your choice: A`
		`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 --..`
		
		If not in columns as shown, do not award screen capture mark
			1

5	1	**3 marks for AO3 (design) and 6 marks for AO3 (programming)** **Note** that AO3 (design) marks are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not regardless of whether the solution works.	9

Level	Description	Mark Range
3	A line of reasoning has been followed to arrive at a logically structured working or almost fully working programmed solution. Code is written to ensure that each letter of the message is encrypted using the user-supplied keys. All of the appropriate design decisions have been taken.	7-9
2	There is evidence that a line of reasoning has been partially followed. The encryption of each character does not fully comply with requirements. There is evidence of some appropriate design work.	4-6
1	An attempt has been made to amend the subroutines. Some appropriate programming statements have been written. There is little evidence to suggest that a line of reasoning has been followed or that the solution has been designed. The statements written may or may not be syntactically correct and the subroutines will have very little or none of the extra required functionality. It is unlikely that any of the key design elements of the task have been recognised.	1-3

Marking guidance:

Evidence of AO3 design – 3 points:

Evidence of design to look for in response:

1) identifying the need to validate a key is an integer
2) identifying a method to encrypt each character with a key
3) identifying suitable method to alternate keys depending on character position in message

Evidence of AO3 programming – 6 points:

Evidence of programming to look for in response:

4) in `SendReceiveMessages` correctly store 3 integer keys entered by the user (in a list or separate variables)
5) amend call and subroutine header of `SendMorseCode` to include keys as parameter(s)
6) correctly encrypt first three characters of message
7) correctly encrypt all characters in message
8) ensure index is within range of array subscripts
9) code to encrypt character inserted in suitable place in `SendMorseCode`

5	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from 14.1, including prompts on screen capture matching those in code.
		Code for 14.1 must be sensible.
		
		Screen capture showing:
		`17, 5 and -3` being entered followed by option `S` and then `TEA X` followed by the output `.--- .--- -.-- --.- -...`
		
		`Enter encryption key (integer): 17`
		`Enter encryption key (integer): 5`
		`Enter encryption key (integer): -3`
		
		`Main Menu`
		`=========`
		`R - Receive Morse code`
		`S - Send Morse code`
		`X - Exit program`
		
		`Enter your choice: S`
		`Enter your message (uppercase letters and spaces only):`
		`TEA X`
		`.--- .--- -.-- --.- -...`
			1

6	1	**All marks for AO3 (programming)** **Mark as follows:** 1) Correct variable declarations for `NumberIn`, `NumberOut`, `Count`, `PartValue`; **Note to examiners** If a language allows variables to be used without explicit declaration (eg Python) then this mark should be awarded if the correct variables exist in the program code and the first value they are assigned is of the correct data type. 2) Correct prompt "`Enter a positive whole number: `" and `NumberIn` assigned value entered by user; 3) Correct initialisation of `NumberOut` and `Count`; 4) `WHILE` loop with syntax allowed by the programming language and correct condition for termination of the loop; 5) Correct incrementation of `Count` within `WHILE` loop; 6) Correct assignment to `PartValue` within `WHILE` loop but before `FOR` loop; 7) Correct updating of `NumberIn` within `WHILE` loop but before `FOR` loop; 8) `FOR` loop with syntax allowed by the programming language over correct range; 9) Correct assignment to `PartValue` inside `FOR` loop; 10) Correct calculation of `NumberOut` after `FOR` loop but within `WHILE` loop; 11) Output statement giving correct output after `WHILE` loop; **I.** Ignore minor differences in case and spelling **Max 10** if code does not function correctly	11

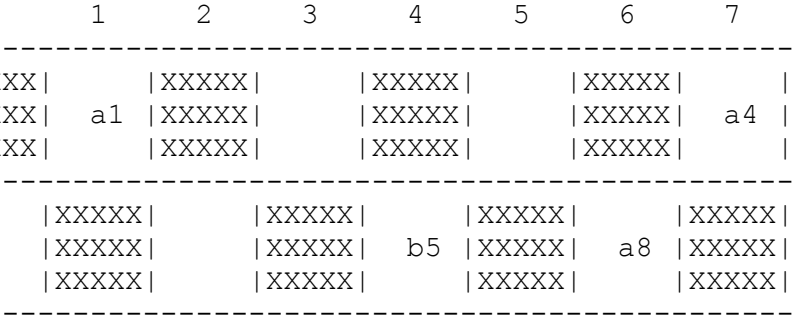
6	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from 03.1, including prompts on screen capture matching those in code.
		Code for 03.1 must be sensible.
		
		Screen capture showing:
		'22' being entered and the message 'The result is: 10110' displayed
		'29' being entered and the message 'The result is: 11101' displayed
		'-1' being entered and the message 'The result is: 0' displayed
		
		`Enter a positive whole number: 22`
		`The result is: 10110`
		`>>>`
		`Enter a positive whole number: 29`
		`The result is: 11101`
		`>>>`
		`Enter a positive whole number: -1`
		`The result is: 0`
		`>>>`
			1

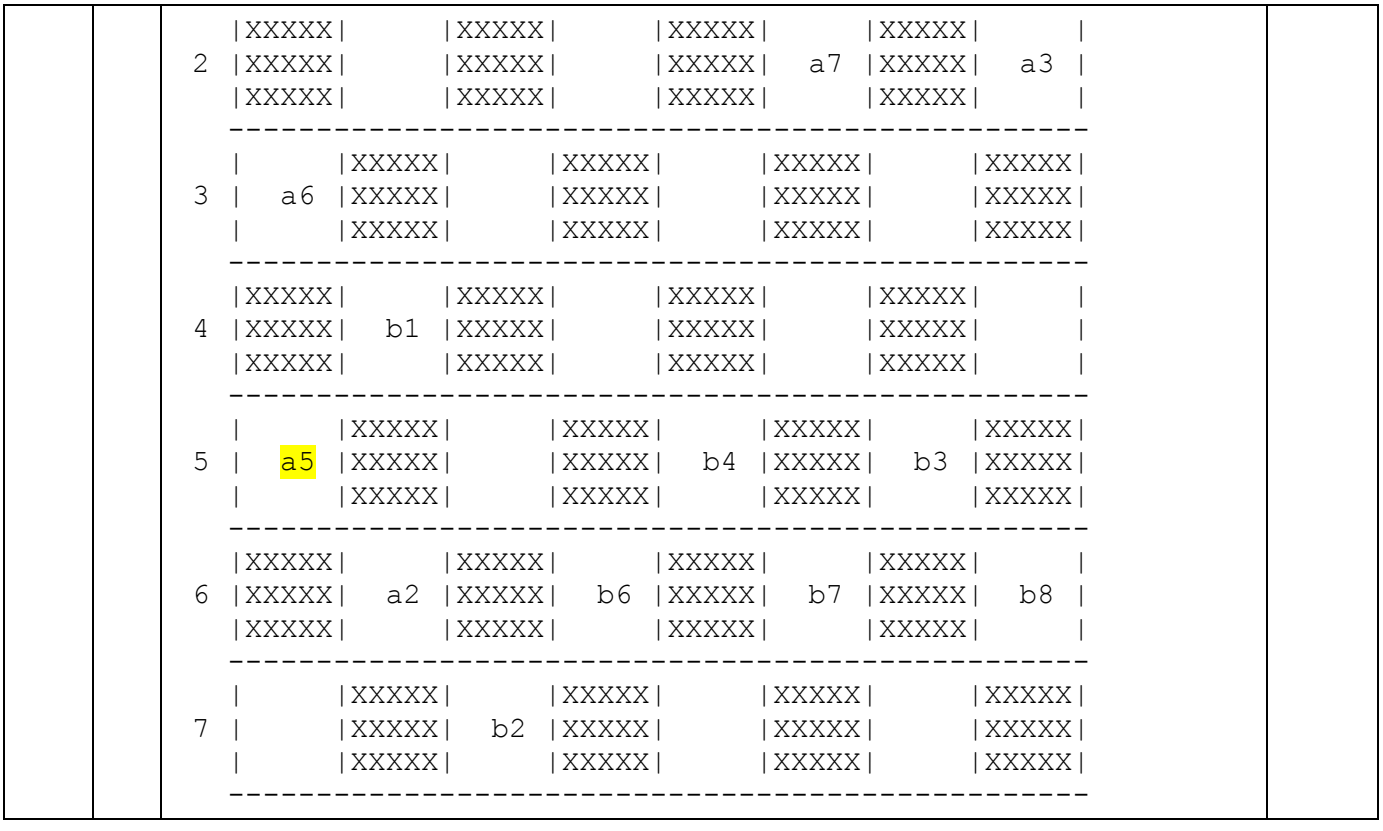
6	3	**Mark is for AO2 (analyse)** converts from (positive) decimal/denary to binary;	1

7	1	**All marks for AO3 (programming)**
		
		Mark as follows:
		1 mark for error codes 1 to 3 tested (using IF, nested IF or CASE)
		A Error messages in a data structure and accessed via error code as index
		1 mark for appropriate error messages **(A** similar wording but same meaning as**):**
		`'Error code 1 - Not a valid piece'`
		`'Error code 2 - Not a valid move'`
		`'Error code 3 - Not a number'`
		1 mark outputting error code (1, 2, 3 or 4)
		
		Note:
		Messages such as "Error Code 1 – not valid" are not detailed enough and are not creditworthy.
			3

7	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from 14.1, including prompts on screen capture matching those in code.
		Code for 14.1 must be sensible.
		
		Screen capture showing:
		`Next Player: a`
		`a5 can jump to 3 , 2`
		`a6 can jump to 3 , 0`
		`a6 can jump to 3 , 4`
		`a7 can jump to 3 , 2`
		`a7 can jump to 3 , 6`
		`a8 can jump to 3 , 4`
		`a9 can move to 3 , 0`
		`a9 can move to 3 , 2`
		`a10 can move to 3 , 2`
		`a10 can move to 3 , 4`
		`a11 can move to 3 , 4`
		`a11 can move to 3 , 6`
		`a12 can move to 3 , 6`
		`There are 13 possible moves`
		`Which piece do you want to move? a4`
		`Error code 1 - not a valid piece`
		`Which piece do you want to move? a9`
		`Which row do you want to move to? 3`
		`Which column do you want to move to? 4`
		`Error code 2 - not a valid move`
		`Which row do you want to move to? a`
		`Which column do you want to move to? 9`
		`Error code 3 - not a number`
		`Which row do you want to move to? 3`
		`Which column do you want to move to? 0`
			1

8	**1**	**1 mark for AO3 (design) and 1 mark for AO3 (programming)** **Mark as follows:** **AO3 (design) – 1 mark:** 1) choosing the final if statement to amend; **AO3 (programming) – 1 mark:** 2) correct logic statement;	**2**

8	**2**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from 15.1, including prompts on screen capture matching those in code.
		Code for 15.1 must be sensible.
		
		Screen capture showing:
		`Next Player: a`
		`a1 can move to 1 , 0`
		`a1 can move to 1 , 2`
		`a2 can move to 7 , 0`
		`a3 can move to 3 , 6`
		`a5 can move to 4 , 3`
		`a5 can jump to 5 , 0`
		`a6 can jump to 5 , 2`
		`a7 can move to 3 , 4`
		`a7 can move to 3 , 6`
		`There are 9 possible moves`
		`Which piece do you want to move? a5`
		`Which row do you want to move to? 5`
		`Which column do you want to move to? 0`
		`jumped over b1`
		
		`Player A:`
		`[[9, 0, 0], [0, 1, 0], [6, 1, 0], [2, 7, 0], [0, 7, 0], [5, 0, 0], [3, 0, 0], [2, 5, 0], [1, 6, 0], [-1, -1, 0], [-1, -1, 0], [-1, -1, 0], [-1, -1, 0]]`
		`Player B:`
		`[[8, 0, 0], [4, 1, 0], [7, 2, 0], [5, 6, 0], [5, 4, 0], [1, 4, 0], [6, 3, 0], [6, 5, 0], [6, 7, 0], [-1, -1, 0], [-1, -1, 0], [-1, -1, 0], [-1, -1, 0]]`
		
		 
			1


2
a7
a3
3
a6
4
b1
5
a5
b4
b3
6
a2
b6
b7
b8
7
b2

<table>
<tr>
<td>9</td>
<td>1</td>
<td>
2 marks for AO3 (design) and 7 marks for AO3 (programming)
<table>
<tr><th>Level</th><th>Description</th><th>Mark Range</th></tr>
<tr><td>3</td><td>A line of reasoning has been followed to arrive at a logically structured working or almost fully working programmed solution.
All of the appropriate design decisions have been taken.</td><td>7–9</td></tr>
<tr><td>2</td><td>There is evidence that a line of reasoning has been partially followed. There is evidence of some appropriate design work.</td><td>4–6</td></tr>
<tr><td>1</td><td>An attempt has been made to write and amend the subroutine <code>PrintResult</code>. Some appropriate programming statements have been written. There is little evidence to suggest that a line of reasoning has been followed or that the solution has been designed. The statements written may or may not be syntactically correct and the subroutines will have very little or none of the extra required functionality. It is unlikely that any of the key design elements of the task have been recognised.</td><td>1–3</td></tr>
</table>
Marking guidance:

Evidence of AO3 design – 2 points:

Evidence of design to look for in response:

1) subroutine <code>CountNumberOfPieces</code> with interface so can be used for both A and B
2) A method for checking piece exists on board

Evidence of AO3 programming – 7 points:

Evidence of programming to look for in response:

3) in <code>CountNumberOfPieces</code> count variable initialised, updated and returned correctly
A counting non-dames only
4) in <code>CountNumberOfPieces</code> loop through A/B/PlayersPieces
5) use value stored in A/B [0,1] as the number of dames
6) formula given in Q correctly programmed
7) comparing the two players' scores and output winner correctly
8) output calculated scores
9) sensible output in case of a draw

Note: output is the same whether or not Question 15 has been attempted.
</td>
<td>9</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>
Mark is for AO3 (evaluate)

**** SCREEN CAPTURE ****
Must match code from 16.1, including prompts on screen capture matching those
</td>
<td>1</td>
</tr>
</table>

in code.
Code for 16.1 must be sensible.

Screen capture showing:

```
Enter the filename: game4.txt

Player A:
[[15, 2, 0], [1, 2, 0], [0, 3, 0], [0, 5, 0], [1, 6, 0], [0,
1, 1], [1, 0, 1], [1, 4, 0], [2, 7, 0], [2, 1, 0], [2, 3, 0],
[2, 5, 0], [3, 6, 0]]
Player B:
[[15, 0, 0], [4, 3, 0], [5, 0, 0], [5, 6, 0], [5, 4, 0], [4,
1, 0], [3, 2, 0], [6, 5, 0], [6, 7, 0], [3, 0, 0], [3, 4, 0],
[4, 5, 0], [4, 7, 0]]

      0     1     2     3     4     5     6     7
   -------------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 0 |XXXXX|  A5 |XXXXX|  a2 |XXXXX|  a3 |XXXXX|     |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -------------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 1 |  A6 |XXXXX|  a1 |XXXXX|  a7 |XXXXX|  a4 |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -------------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 2 |XXXXX|  a9 |XXXXX| a10 |XXXXX| a11 |XXXXX|  a8 |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -------------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 3 |  b9 |XXXXX|  b6 |XXXXX| b10 |XXXXX| a12 |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -------------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 4 |XXXXX|  b5 |XXXXX|  b1 |XXXXX| b11 |XXXXX| b12 |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -------------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 5 |  b2 |XXXXX|     |XXXXX|  b4 |XXXXX|  b3 |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -------------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 6 |XXXXX|     |XXXXX|     |XXXXX|  b7 |XXXXX|  b8 |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -------------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 7 |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -------------------------------------------------
Next Player:  a
There are  0  possible moves
Game ended
A won this game with a score of  -17
B got a score of  3
```

<table>
<tr><td>10</td><td>1</td><td>Mark is for AO2 (analyse)

<code>OpponentsPieces;</code>

R. if any additional code
R. if spelt incorrectly
I. case & spacing</td><td>1</td></tr>
<tr><td>10</td><td>2</td><td>2 marks for AO3 (design) and 7 marks for AO3 (programming)

<table>
<tr><th>Level</th><th>Description</th><th>Mark Range</th></tr>
<tr><td>3</td><td>A line of reasoning has been followed to arrive at a logically structured working or almost fully working programmed solution.
All of the appropriate design decisions have been taken.</td><td>7–9</td></tr>
<tr><td>2</td><td>There is evidence that a line of reasoning has been partially followed. There is evidence of some appropriate design work.</td><td>4–6</td></tr>
<tr><td>1</td><td>An attempt has been made to amend the subroutine <code>MoveDame</code>. Some appropriate programming statements have been written. There is little evidence to suggest that a line of reasoning has been followed or that the solution has been designed. The statements written may or may not be syntactically correct and the subroutines will have very little or none of the extra required functionality. It is unlikely that any of the key design elements of the task have been recognised.</td><td>1–3</td></tr>
</table>

Marking guidance:

Evidence of AO3 design – 2 points:

Evidence of design to look for in response:

1) validate that chosen piece is an opponent’s existing piece
2) return updated <code>OpponentsPieces</code> from subroutine <code>MoveDame</code> (parameter by reference)

Evidence of AO3 programming – 7 points:

Evidence of programming to look for in response:

3) user prompt for which piece to take
4) extracting player letter from chosen piece
5) extracting index from chosen piece
6) retrieving coodinates from <code>OpponentsPieces</code>
7) set opponent’s piece coordinates to -1
8) new dame’s coordinates set to taken piece’s coordinates
9) update parameters in calls to <code>MovePiece</code> in subroutine <code>MakeMove</code> (parameter by reference)

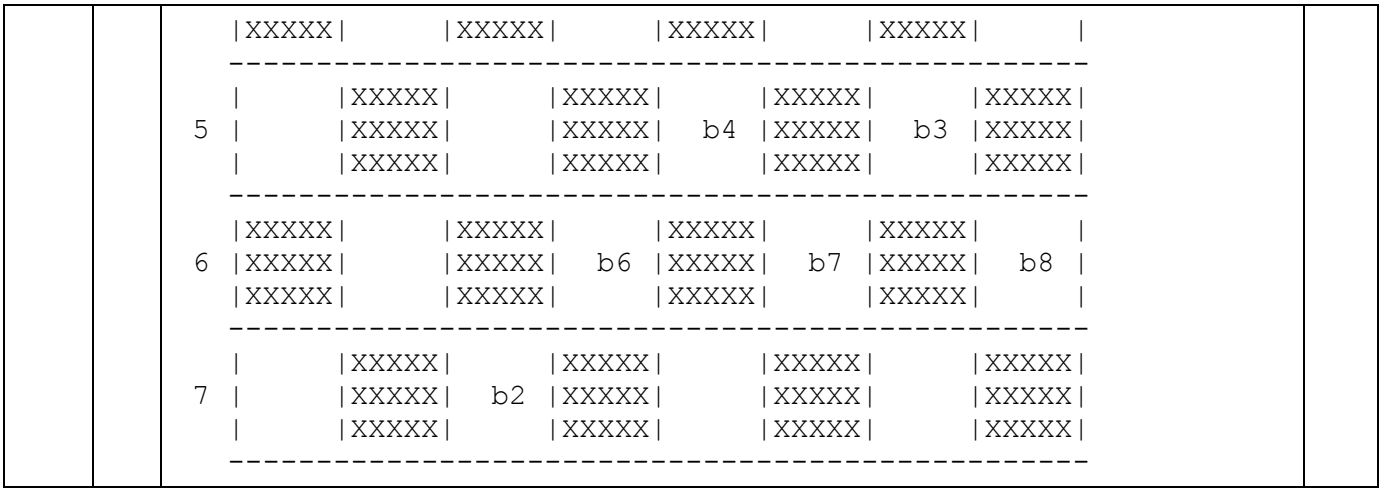
A. solutions that ask the user to input the row and column of the piece to be removed.</td><td>9</td></tr>
</table>

10	**3**	**Mark is for AO3 (evaluate)** ****** SCREEN CAPTURE ****** Must match code from 17.2, including prompts on screen capture matching those in code. Code for 17.2 must be sensible. Screen capture showing: Do you want to load a saved game? (Y/N): y Enter the filename: game3.txt Player A: [[8, 0, 0], [0, 1, 0], [6, 1, 0], [2, 7, 0], [0, 7, 0], [3, 2, 0], [3, 0, 0], [2, 5, 0], [1, 6, 0], [-1, -1, 0], [-1, -1, 0], [-1, -1, 0], [-1, -1, 0]] Player B: [[8, 0, 0], [4, 1, 0], [7, 2, 0], [5, 6, 0], [5, 4, 0], [1, 4, 0], [6, 3, 0], [6, 5, 0], [6, 7, 0], [-1, -1, 0], [-1, -1, 0], [-1, -1, 0], [-1, -1, 0]] 0 1 2 3 4 5 6 7 0 a1 a4 1 b5 a8 2 a7 a3 3 a6 a5 4 b1 5 b4 b3 6 a2 b6 b7 b8 7 b2 Next Player: a a1 can move to 1 , 0	**1**

```
a1  can move to  1  ,  2
a2  can move to  7  ,  0
a3  can move to  3  ,  6
a5  can move to  4  ,  3
a7  can move to  3  ,  4
a7  can move to  3  ,  6
a8  can jump to  3  ,  4
There are  8  possible moves
Which piece do you want to move? a2
Which row do you want to move to? 7
Which column do you want to move to? 0
Which piece do you want to take? b1

Player A:
[[9, 1, 0], [0, 1, 0], [4, 1, 1], [2, 7, 0], [0, 7, 0], [3, 2,
0], [3, 0, 0], [2, 5, 0], [1, 6, 0], [-1, -1, 0], [-1, -1, 0],
[-1, -1, 0], [-1, -1, 0]]
Player B:
[[8, 0, 0], [-1, -1, 0], [7, 2, 0], [5, 6, 0], [5, 4, 0], [1,
4, 0], [6, 3, 0], [6, 5, 0], [6, 7, 0], [-1, -1, 0], [-1, -1,
0], [-1, -1, 0], [-1, -1, 0]]

     0     1     2     3     4     5     6     7
   -----------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 0 |XXXXX|  a1 |XXXXX|     |XXXXX|     |XXXXX|  a4 |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -----------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 1 |     |XXXXX|     |XXXXX|  b5 |XXXXX|  a8 |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -----------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 2 |XXXXX|     |XXXXX|     |XXXXX|  a7 |XXXXX|  a3 |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -----------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 3 |  a6 |XXXXX|  a5 |XXXXX|     |XXXXX|     |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -----------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 4 |XXXXX|  A2 |XXXXX|     |XXXXX|     |XXXXX|     |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -----------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 5 |     |XXXXX|     |XXXXX|  b4 |XXXXX|  b3 |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -----------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 6 |XXXXX|     |XXXXX|  b6 |XXXXX|  b7 |XXXXX|  b8 |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -----------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 7 |     |XXXXX|  b2 |XXXXX|     |XXXXX|     |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -----------------------------------------------
```

10	**4**	**Mark is for AO3 (evaluate)** ****** SCREEN CAPTURE ****** Must match code from 17.2, including prompts on screen capture matching those in code. Code for 17.2 must be sensible. Screen capture showing:	**1**

```
Next Player:  b
b2  can move to  6  ,  1
b3  can move to  4  ,  5
b3  can move to  4  ,  7
b4  can move to  4  ,  3
b4  can move to  4  ,  5
b5  can move to  0  ,  3
b5  can move to  0  ,  5
b6  can move to  5  ,  2
b6  can jump to  4  ,  5
b7  can jump to  4  ,  3
b7  can jump to  4  ,  7
b8  can jump to  4  ,  5
There are  12  possible moves
Which piece do you want to move? b5
Which row do you want to move to? 0
Which column do you want to move to? 3
Which piece do you want to take? a6

Player A:
[[9, 1, 0], [0, 1, 0], [4, 1, 1], [2, 7, 0], [0, 7, 0], [3, 2,
0], [-1, -1, 0], [2, 5, 0], [1, 6, 0], [-1, -1, 0], [-1, -1,
0], [-1, -1, 0], [-1, -1, 0]]
Player B:
[[9, 1, 0], [-1, -1, 0], [7, 2, 0], [5, 6, 0], [5, 4, 0], [3,
0, 1], [6, 3, 0], [6, 5, 0], [6, 7, 0], [-1, -1, 0], [-1, -1,
0], [-1, -1, 0], [-1, -1, 0]]

      0     1     2     3     4     5     6     7
   -------------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 0 |XXXXX|  a1 |XXXXX|     |XXXXX|     |XXXXX|  a4 |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -------------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 1 |     |XXXXX|     |XXXXX|     |XXXXX|  a8 |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -------------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 2 |XXXXX|     |XXXXX|     |XXXXX|  a7 |XXXXX|  a3 |
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
   -------------------------------------------------
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
 3 |  B5 |XXXXX|  a5 |XXXXX|     |XXXXX|     |XXXXX|
   |     |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|
   -------------------------------------------------
   |XXXXX|     |XXXXX|     |XXXXX|     |XXXXX|     |
 4 |XXXXX|  A2 |XXXXX|     |XXXXX|     |XXXXX|     |
```


5
6
7
b4
b3
b6
b7
b8
b2

Qu		Marks	
11		**2 marks for AO1 (knowledge)** Problem definition; Requirements specification // list of objectives; Feedback about requirements specification from end user; Data model / ER diagram; Analysis data dictionary; Interviews; Questionnaires; Observations; Examination of documents; Research existing solutions; Acceptable limitations / constraints; **Max 2**	**2**

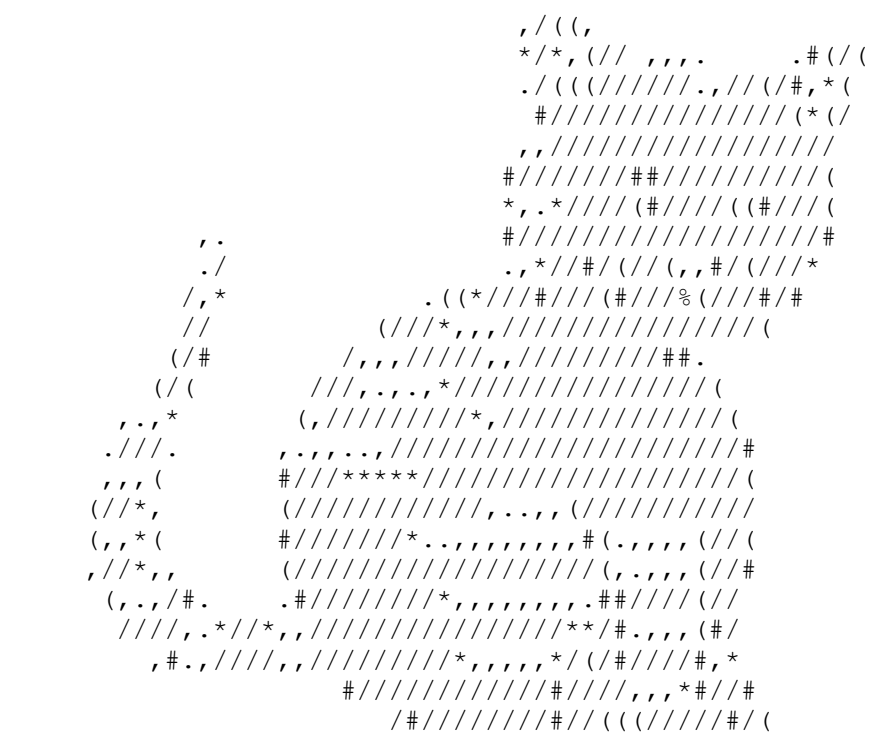
12	**1**	**9 marks for AO3 (programming)** **Mark as follows:** 1) Correct variable declarations for `X`, `Product`, `Factor`, **Note to examiners** If a language allows variables to be used without explicit declaration (eg Python) then this mark should be awarded if the correct variables exist in the program code and the first value they are assigned is of the correct data type. 2) Correct prompt "`Enter an integer greater than 1: `" and `X` assigned integer value entered by user; 3) Correct initialisation of `Product` and `Factor` before `WHILE` loop; 4) `WHILE` loop with syntax allowed by the programming language and correct condition for termination of the loop; 5) Correct incrementation of `Factor` and correct assignment to `Product` within `WHILE` loop; 6) `IF` statement with correct condition and `ELSE` part after the `WHILE` loop; 7) Correct re-initialisation of `Product` within `THEN` part; 8) `FOR` loop with syntax allowed by the programming language over correct range within `THEN` part; 9) Correct assignment to `Product` and output of `N` within `FOR` loop; **I.** minor differences in case and spelling **DPT.** use of incorrect variable name **Max 8** if code does not function correctly	**9**

12	**2**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **03.1**, including prompts on screen capture matching those in code.
		Code for **03.1** must be sensible.
		
		Screen capture showing:
		'720' being entered and 1 2 3 4 5 6 displayed
		(Accept on same or separate lines)
		'600' being entered and the message 'No result' displayed
		
		`Enter an integer greater than 1: 720`
		`1`
		`2`
		`3`
		`4`
		`5`
		`6`
		`>>>`
		`Enter an integer greater than 1: 600`
		`No result`
			1

		>>>	

12	**3**	**Mark is for AO2 (analyse)**
		
		X is equal to the product of a sequence of (consecutive) whole numbers starting at 1
		// X is a factorial number (greater than 1)
		// X is the factorial of a positive integer (greater than 1);
			1

Qu		Marks	
13	1	**1 mark for AO3 (design) and 5 marks for AO3 (programming)** **Mark as follows:** **AO3 (design) – 1 mark:** 1) Declare a new grid to receive mirror image; **AO3 (programming) – 5 marks:** 2) Create subroutine header with required parameters, **I.** extra parameters; 3) Column reference adjusted for mirror image; 4) Nested loops with correct ranges; 5) Add menu option in `DisplayMenu`; 6) Add call to `MirrorImage` in suitable place with parameters that match subroutine definition in code, **A**. call to `MirrorImage` in suitable place with `grid` and `header` parameters if subroutine definition not provided; **Max 5** if code does not function correctly.	**6**

13	**2**	**Mark is for AO3 (evaluate)** ****** SCREEN CAPTURE ****** Must match code from **13.1**, including prompts on screen capture matching those in code. Code for **13.1** must be sensible. Screen capture showing: Main Menu ========= L - Load graphics file D - Display image E - Edit image S - Save image M - Mirror image X - Exit program Enter your choice: M Cat === 	**1**

14	**1**	**3 marks for AO3 (design) and 6 marks for AO3 (programming)**	**9**

Level	Description	Mark Range
3	A line of reasoning has been followed to arrive at a logically structured working or almost fully working programmed solution. All of the appropriate design decisions have been taken. The hidden message may not have been built entirely correctly.	7–9
2	There is evidence that a line of reasoning has been partially followed. There is evidence of some appropriate design work. The subroutine `LoadGreyScaleImage` has been amended with a call to `FindSecretChar` in an appropriate place.	4–6
1	An attempt has been made to write the subroutine `FindSecretChar`. Some appropriate programming statements have been written. There is little evidence to suggest that a line of reasoning has been followed or that the solution has been designed. The statements written may or may not be syntactically correct and the subroutines will have very little or none of the extra required functionality. It is unlikely that any of the key design elements of the task have been recognised.	1–3

Marking guidance:

Evidence of AO3 design – 3 marks:

Evidence of design to look for in response:

1) check whether value of pixel is in the correct range
2) convert a range of integers to a range of letters
3) call `FindSecretChar` with `PixelValue` and `Key` as parameters.

Note: AO3 (design) points are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not and regardless of whether the solution works.

Evidence of AO3 programming – 6 marks:

Evidence of programming to look for in response:

4) correct subroutine header and parameters for `FindSecretChar`, **I.** return type
5) generate underscore if no decrypted character found // generate space if `PixelValue - Key` is zero
6) always returns the correct character
7) extract the key from the file header
8) concatenate hidden message and returned character within `FOR` loop
9) output the hidden message after `FOR` loop.

Max 8 if code does not function correctly.

14	**2**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **14.1**, including prompts on screen capture matching those in code.
		Code for **14.1** must be sensible.
		
		Screen capture showing:
		
		`Enter your choice: L`
		`Enter filename to load: greyscale`
		`H__E___L___P ME`
		
		`TestImage2`
		`==========`
		`##&#.`
		`&:#+&`
		` ####`
		
		A. hyphen instead of underscore character
			1

15	**1**	**3 marks for AO3 (design) and 9 marks for AO3 (programming)**	**12**

Level	Description	Mark Range
3	A line of reasoning has been followed to arrive at a logically structured working or almost fully working programmed solution. All of the appropriate design decisions have been taken. The last value pair may not have been saved.	9–12
2	There is evidence that a line of reasoning has been partially followed. There is evidence of some appropriate design work. Consecutive pixels are counted and most value pairs saved to a new file, delimiter may be missing.	5–8
1	An attempt has been made to write the subroutine `CompressFile`. Some appropriate programming statements have been written. There is little evidence to suggest that a line of reasoning has been followed or that the solution has been designed. The statements written may or may not be syntactically correct and the subroutines will have very little or none of the extra required functionality. It is unlikely that any of the key design elements of the task have been recognised. Some appropriate programming statements from the `LoadFile` subroutine may have been used for reading a file.	1–4

Marking guidance:

Evidence of AO3 design – 3 points:

Evidence of design to look for in response:

1) Attempts to create new file with modified file name
2) Structure that compares current character with previous character
3) Under some circumstances counts consecutive symbols correctly

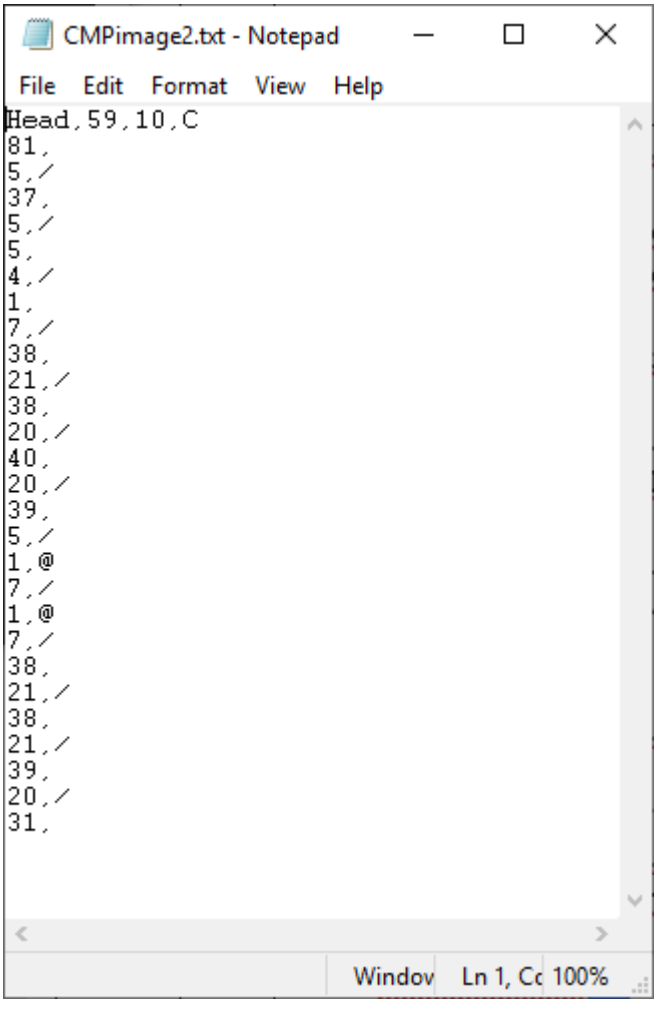
Note: AO3 (design) points are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not and regardless of whether the solution works.

Evidence of AO3 programming – 9 points:

Evidence of programming to look for in response:

4) Ask user for file name
5) Open existing file for reading and new file for writing
6) Edit file header correctly
7) Initialise symbol count for first run of symbols
8) Check each pixel in the file
9) Save first symbol count and symbol to file
10) Save each symbol count and symbol to file (except first and last pairs)

		11) Save last symbol count and symbol to file 12) Reset symbol count for next run of symbols **Max 11** if code does not function correctly.	

15	**2**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **15.1**, including prompts on screen capture matching those in code.
		Code for **15.1** must be sensible.
		
		Screen capture showing:
		
		`Enter your choice: C`
		`Which graphics file do you want to compress? image2`
		
		 
		
			1

Qu		Marks	
16	1	**8 marks for AO3 (programming)** **Mark as follows:** 1) Correct variable declarations for `C`, `D`, `S`, `T` and initialisation; **Note to examiners:** If a language allows variables to be used without explicit declaration, (eg Python), then this mark should be awarded if the correct variables exist in the program code and the first value they are assigned is of the correct data type. 2) Correct `WHILE` loop syntax allowed by the programming language and correct condition for termination of the loop; 3) Correct generation of two random numbers between 1 and 6 and output within loop; 4) Correct running total assigned to `S` and updating of `T`; 5) Correct condition to increment `C` within the loop; 6) Correct condition to increment `D` with the loop; 7) Correct calculation of `A` outside the loop; 8) Correct output outside loop; **I**. case **Max 7** if code does not function correctly	**8**

16	**2**	**Mark is for AO3 (evaluate)** ****** SCREEN CAPTURE ****** Must match code from **03.1**. Code for **03.1** must be sensible. As this output is from a random number generator, the output from candidates will not be the values below. However: Output should show two digits between 1 and 6 on each line except the final line. On the final line it should show the correct number of lines containing at least one 6, number of doubles and another integer. Number of lines containing at least one 6 and/or number of doubles must be 3 Screen capture showing: 1 5 4 3 3 1 4 2 2 2 3 3 1 6 6 5 3 5 6 3 3 2 3 >>> **I**. missing spaces **A**. each digit on a new line	**1**

Qu		Marks	
17	1	**7 marks for AO3 (programming)** **Marking guidance:** **Evidence of AO3 programming – 7 points:** Evidence of programming to look for in response: 1. Loop structure; 2. Check through each element of `Puzzle` // check `Puzzle` until end of entries reached // check until a protected cell is found; 3. Compare first two characters of `CellInfo` with first two characters of `Puzzle` line // compare the string values of `Row` and `Column` with the first two characters of the `Puzzle` line; 4. If protected cell found …; 5. … give appropriate error message to user **A.** 'Invalid input' <u>also</u> being output; 6. Only if it is not a protected cell update content of cell; **DPT.** Incorrect identification of a protected cell 7. Code in correct place in `SolvePuzzle`; **Max 6** if any errors	**7**

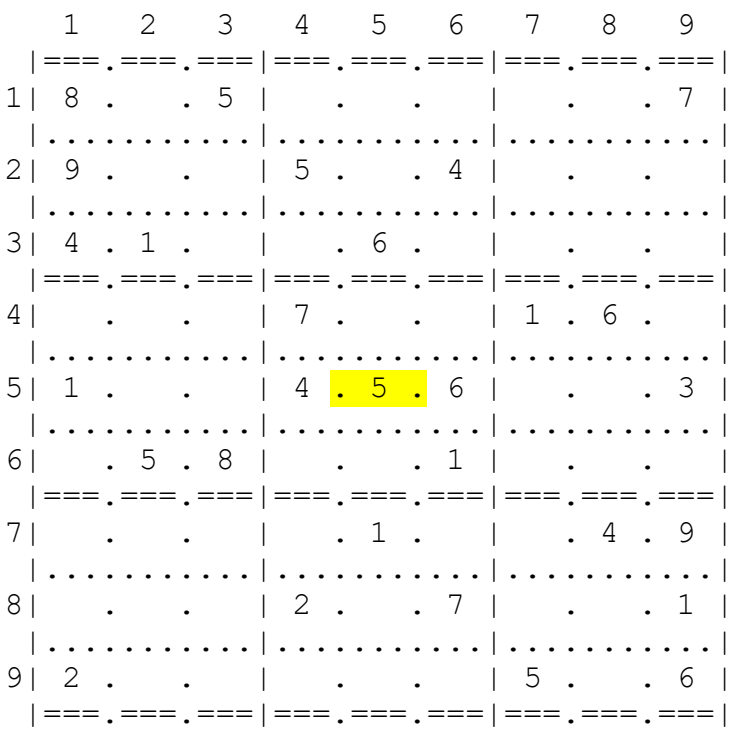
17	**2**	**Mark is for AO3 (evaluate)** ****** SCREEN CAPTURE ****** Must match code from **12.1**, including prompts on screen capture matching those in code. Code for **12.1** must be sensible. Screen capture showing: Enter your choice: S 1 2 3 4 5 6 7 8 9 |===.===.===|===.===.===|===.===.===| 1| 8 . . 5 | . . | . . 7 | |.....|.....|.....| 2| 9 . . | 5 . 7 . 4 | . . | |.....|.....|.....| 3| 4 . 1 . 7 | . 6 . | . . | |===.===.===|===.===.===|===.===.===| 4| . . | 7 . . | 1 . 6 . | |.....|.....|.....| 5| 1 . 7 . | 4 . . 6 | . . 3 | |.....|.....|.....| 6| 6 . 5 . 8 | . . 1 | . . | |===.===.===|===.===.===|===.===.===| 7| . . | . 1 . | . 4 . 9 | |.....|.....|.....| 8| . . | 2 . . 7 | . . 1 | |.....|.....|.....| 9| 2 . . | . . | 5 . . 6 | |===.===.===|===.===.===|===.===.===| Enter row column digit: (Press Enter to stop) 117 You can't change a protected cell Enter row column digit: (Press Enter to stop) 323 You can't change a protected cell Enter row column digit: (Press Enter to stop) 993 You can't change a protected cell Enter row column digit: (Press Enter to stop) 853		**1**

```
   1   2   3   4   5   6   7   8   9
 |===.===.===|===.===.===|===.===.===|
1| 8 .   . 5 |   .   .   |   .   . 7 |
 |...........|...........|...........|
2| 9 .   .   | 5 . 7 . 4 |   .   .   |
 |...........|...........|...........|
3| 4 . 1 . 7 |   . 6 .   |   .   .   |
 |===.===.===|===.===.===|===.===.===|
4|   .   .   | 7 .   .   | 1 . 6 .   |
 |...........|...........|...........|
5| 1 . 7 .   | 4 .   . 6 |   .   . 3 |
 |...........|...........|...........|
6| 6 . 5 . 8 |   .   . 1 |   .   .   |
 |===.===.===|===.===.===|===.===.===|
7|   .   .   |   . 1 .   |   . 4 . 9 |
 |...........|...........|...........|
8|   .   .   | 2 . 3 . 7 |   .   . 1 |
 |...........|...........|...........|
9| 2 .   .   |   .   .   | 5 .   . 6 |
 |===.===.===|===.===.===|===.===.===|

Enter row column digit:
(Press Enter to stop)
854

   1   2   3   4   5   6   7   8   9
 |===.===.===|===.===.===|===.===.===|
1| 8 .   . 5 |   .   .   |   .   . 7 |
 |...........|...........|...........|
2| 9 .   .   | 5 . 7 . 4 |   .   .   |
 |...........|...........|...........|
3| 4 . 1 . 7 |   . 6 .   |   .   .   |
 |===.===.===|===.===.===|===.===.===|
4|   .   .   | 7 .   .   | 1 . 6 .   |
 |...........|...........|...........|
5| 1 . 7 .   | 4 .   . 6 |   .   . 3 |
 |...........|...........|...........|
6| 6 . 5 . 8 |   .   . 1 |   .   .   |
 |===.===.===|===.===.===|===.===.===|
7|   .   .   |   . 1 .   |   . 4 . 9 |
 |...........|...........|...........|
8|   .   .   | 2 . 4 . 7 |   .   . 1 |
 |...........|...........|...........|
9| 2 .   .   |   .   .   | 5 .   . 6 |
 |===.===.===|===.===.===|===.===.===|
```

A. grid displaying between incorrect attempts

Qu		Marks	
18	1	**2 marks for AO3 (design) and 6 marks for AO3 (programming)** **Marking guidance:** **Evidence of AO3 design – 2 points:** Evidence of design to look for in response: 1. Identify the need for a loop containing a conditional statement; 2. Recognise that subgrid boundaries need to be considered; **Note: AO3** (design) points are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not and regardless of whether the solution works. **Evidence of AO3 programming – 6 points:** Evidence of programming to look for in response: 3. Subroutine heading with correct parameters and return value; 4. Check both row and column for duplicate **I.** failing to check location at which digit is to be placed; 5. Correctly calculate all subgrid boundaries; 6. Check each digit in any subgrid for duplicate *(does not depend on MP5 above)*; 7. Call in correct place in `SolvePuzzle` so that grid updated only under correct circumstances / if digit is valid / if subroutine returns true **R.** if returned value not used; 8. Give appropriate error message (under at least some correct circumstances and is never displayed when it shouldn't be) **A.** error message in `DuplicateDigit`; **Max 7** if any errors	**8**

18	**2**	**Mark is for AO3 (evaluate)** ****** SCREEN CAPTURE ****** Must match code from **13.1**, including prompts on screen capture matching those in code. Code for **13.1** must be sensible. Screen capture showing: `Enter your choice: S` `  1 2 3 4 5 6 7 8 9` `  |===.===.===|===.===.===|===.===.===|` `1| 8 . . 5	. .	. . 7 |` ` |...........|...........|...........|` `2| 9 . .	5 . . 4	. . |` ` |...........|...........|...........|` `3| 4 . 1 .	. 6 .	. . |` ` |===.===.===|===.===.===|===.===.===|` `4| . .	7 . .	1 . 6 . |` ` |...........|...........|...........|` `5| 1 . .	4 . . 6	. . 3 |` ` |...........|...........|...........|` `6| . 5 . 8	. . 1	. . |` ` |===.===.===|===.===.===|===.===.===|` `7| . .	. 1 .	. 4 . 9 |` ` |...........|...........|...........|` `8| . .	2 . . 7	. . 1 |` ` |...........|...........|...........|` `9| 2 . .	. .	5 . . 6 |` ` |===.===.===|===.===.===|===.===.===|`  `Enter row column digit:` `(Press Enter to stop)` `178` `Duplicate digit` `Enter row column digit:` `(Press Enter to stop)` `819` `Duplicate digit` `Enter row column digit:` `(Press Enter to stop)` `124` `Duplicate digit` `Enter row column digit:` `(Press Enter to stop)` `989` `Duplicate digit` `Enter row column digit:` `(Press Enter to stop)` `555`	**1**

1 2 3 4 5 6 7 8 9
1 8 5 7
2 9 5 4
3 4 1 6
4 7 1 6
5 1 4 5 6 3
6 5 8 1
7 1 4 9
8 2 7 1
9 2 5 6

19	**1**	**3 marks for AO3 (design) and 9 marks for AO3 (programming)**	**12**

Level	Description	Mark Range
3	A line of reasoning has been followed to arrive at a logically structured working or almost fully working programmed solution. All of the appropriate design decisions have been taken.	9–12
2	There is evidence that a line of reasoning has been partially followed. There is evidence of some appropriate design work. This is a partially working programmed solution.	5–8
1	An attempt has been made to write the subroutine `ClearEntries`. Some appropriate programming statements have been written. There is little evidence to suggest that a line of reasoning has been followed or that the solution has been designed. The statements written may or may not be syntactically correct and the subroutines will have very little or none of the extra required functionality. It is unlikely that any of the key design elements of the task have been recognised.	1–4

Marking guidance:

Evidence of AO3 design – 3 marks:

Evidence of design to look for in response:

1) Recognise a loop required that repeats depending on value entered by user.
2) Dealing with non-integer input.
3) Attempt to identify minus sign / negative number in input.

Note: AO3 (design) points are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not and regardless of whether the solution works.

Evidence of AO3 programming – 9 marks:

Evidence of programming to look for in response:

4) Create `ClearEntries` subroutine with correct parameters.
5) Extract number of cells to be cleared.
6) Extract row/column from entry in `Answer` within loop.
7) Extract row and column from entry in `Answer` within loop.
8) Replace entry in `PuzzleGrid` with space within loop **DPT.** incorrect row and/or column
9) Correct number of cells cleared if number to clear is less than or equal to number of entries **DPT.** incorrect string used to clear in MP8
10) Correct number of cells cleared in all circumstances.
11) Update answer count in `Answer[2]`.
12) Display grid after subroutine call.

		Max 11 if code does not function correctly	

19	**2**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **14.1**, including prompts on screen capture matching those in code.
		Code for **14.1** must be sensible.
		
		`Enter your choice: S`
		
		`   1   2   3   4   5   6   7   8   9`
		`  |===.===.===|===.===.===|===.===.===|`
		`1| 8 .   . 5 |   .   .   |   .   . 7 |`
		`  |...........|...........|...........|`
		`2| 9 .   .   | 5 . 7 . 4 |   .   .   |`
		`  |...........|...........|...........|`
		`3| 4 . 1 . 7 |   . 6 .   |   .   .   |`
		`  |===.===.===|===.===.===|===.===.===|`
		`4|   .   .   | 7 .   .   | 1 . 6 .   |`
		`  |...........|...........|...........|`
		`5| 1 . 7 .   | 4 .   . 6 |   .   . 3 |`
		`  |...........|...........|...........|`
		`6| 6 . 5 . 8 |   .   . 1 |   .   .   |`
		`  |===.===.===|===.===.===|===.===.===|`
		`7|   .   .   |   . 1 .   |   . 4 . 9 |`
		`  |...........|...........|...........|`
		`8|   .   .   | 2 .   . 7 |   .   . 1 |`
		`  |...........|...........|...........|`
		`9| 2 .   .   |   .   .   | 5 .   . 6 |`
		`  |===.===.===|===.===.===|===.===.===|`
		
		`Enter row column digit:`
		`(Press Enter to stop)`
		`-x`
		`not a valid integer`
		`Enter row column digit:`
		`(Press Enter to stop)`
		`-1`
			1

```
   1   2   3   4   5   6   7   8   9
 |===.===.===|===.===.===|===.===.===|
1| 8 .   . 5 |   .   .   |   .   . 7 |
 |...........|...........|...........|
2| 9 .   .   | 5 . 7 . 4 |   .   .   |
 |...........|...........|...........|
3| 4 . 1 . 7 |   . 6 .   |   .   .   |
 |===.===.===|===.===.===|===.===.===|
4|   .   .   | 7 .   .   | 1 . 6 .   |
 |...........|...........|...........|
5| 1 .   .   | 4 .   . 6 |   .   . 3 |
 |...........|...........|...........|
6| 6 . 5 . 8 |   .   . 1 |   .   .   |
 |===.===.===|===.===.===|===.===.===|
7|   .   .   |   . 1 .   |   . 4 . 9 |
 |...........|...........|...........|
8|   .   .   | 2 .   . 7 |   .   . 1 |
 |...........|...........|...........|
9| 2 .   .   |   .   .   | 5 .   . 6 |
 |===.===.===|===.===.===|===.===.===|

Enter row column digit:
(Press Enter to stop)
-5

   1   2   3   4   5   6   7   8   9
 |===.===.===|===.===.===|===.===.===|
1| 8 .   . 5 |   .   .   |   .   . 7 |
 |...........|...........|...........|
2| 9 .   .   | 5 .   . 4 |   .   .   |
 |...........|...........|...........|
3| 4 . 1 .   |   . 6 .   |   .   .   |
 |===.===.===|===.===.===|===.===.===|
4|   .   .   | 7 .   .   | 1 . 6 .   |
 |...........|...........|...........|
5| 1 .   .   | 4 .   . 6 |   .   . 3 |
 |...........|...........|...........|
6|   . 5 . 8 |   .   . 1 |   .   .   |
 |===.===.===|===.===.===|===.===.===|
7|   .   .   |   . 1 .   |   . 4 . 9 |
 |...........|...........|...........|
8|   .   .   | 2 .   . 7 |   .   . 1 |
 |...........|...........|...........|
9| 2 .   .   |   .   .   | 5 .   . 6 |
 |===.===.===|===.===.===|===.===.===|
```

20	**1**	**9 marks for AO3 (programming)** **Mark as follows:** 1) Correct variable declarations for `Number`, `X`, `Count`, `Multi`; **Note to examiners** If a language allows variables to be used without explicit declaration (eg Python) then this mark should be awarded if the correct variables exist in the program code and the first value they are assigned is of the correct data type. 2) Correct prompt `"Enter an integer greater than 1: "` and `Number` assigned integer value entered by user; 3) Correct initialisation of `X` and `Count` before outer `WHILE` loop; 4) Correct outer `WHILE` loop with syntax allowed by the programming language and correct condition for termination of the outer loop; 5) Correct assignment of `Multi` in outer loop; 6) Correct inner `WHILE` loop syntax allowed by the programming language and correct condition for termination of the loop; 7) `IF` statement with correct condition and output within inner loop; 8) Correct incrementation of `Count` and correct assignment to `Multi` and `Number` within inner `WHILE` loop; 9) Correct assignments of `X` in outer loop; **I.** minor differences in case and spelling **Max 8** if code does not function correctly	**9**

20	**2**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **05.1**, including prompts on screen capture matching those in code.
		Code for **05.1** must be sensible.
		
		Screen capture showing:
		'23' being entered and '23' displayed followed, by '1'
		'25' being entered and the message '5' displayed, followed by '2'
		'1260' being entered and '2 3 5 7' displayed, followed by '6'
		
		(Accept on same or separate lines)
		
		`Enter a number greater than 1: 23`
		`23`
		`1`
		`>>>`
		`Enter a number greater than 1: 25`
		`5`
		`2`
		`>>>`
		`Enter a number greater than 1: 1260`
		`2`
		`3`
		`5`
		`7`
		`6`
		`>>>`
		
		Alternative:
		
		`Enter a number greater than 1: 23`
		`23`
		`1`
		`>>>`
		`Enter a number greater than 1: 25`
		`5`
		`2`
		`>>>`
		`Enter a number greater than 1: 1260`
		`2  3  5  7`
		`6`
		`>>>`
			1

Qu		Marks	
21	1	**4 marks for AO3 (programming)** **Mark as follows:** 1) Add required parameter to subroutine call in `Execute`; 2) Add required parameter to `ExecuteSKP` definition; 3) Add `1` to `Registers[ACC]`; 4) Update status register (by calling `SetFlags` with correct parameters); **Max 3** if any errors	**4**

21	**2**	**Mark is for AO3 (evaluate)** ****** SCREEN CAPTURE ****** Must match code from **12.1**, including prompts on screen capture matching those in code. Code for **12.1** must be sensible. Screen capture showing (values changing from Frame 0 to Frame 5 shown highlighted): `Enter your choice: R` `****** Frame 0 **********` `*` `*  Memory  Location  Label  Op  Operand Comment` `*  Contents  Code` `* JMP 1	0 |` `* LDA# 3	1	LDA# 3 * test negative` `* SUB 10	2	SUB NUM1` `* SKP 0	3	SKP` `* STA 11	4	STA FINAL` `* HLT 0	5	HLT` `* 0	6 |` `* 0	7 |` `* 0	8 |` `* 0	9 |` `* 5	10	NUM1: 5` `* 0	11	FINAL: 0` `*` `* PC: 0 ACC: 0 TOS: 20` `* Status Register: ZNV` `* 100` `**********`  `****** Frame 5 **********` `* Current Instruction Register: STA 11` `*` `* Memory Location Label Op Operand Comment` `* Contents Code` `* JMP 1	0 |` `* LDA# 3	1	LDA# 3 * test negative` `* SUB 10	2	SUB NUM1` `* SKP 0	3	SKP` `* STA 11	4	STA FINAL` `* HLT 0	5	HLT` `* 0	6 |` `* 0	7 |` `* 0	8 |` `* 0	9 |` `* 5	10	NUM1: 5` `* -1	11	FINAL: 0` `*` `* PC: 5 ACC: -1 TOS: 20` `* Status Register: ZNV` `* 010` `**********` `Execution terminated`	**1**

Qu		Marks	
22	**1**	**5 marks for AO3 (programming)** **Mark as follows:** 1) Check for non-integer input; 2) Check within valid lower boundary; 3) Check within valid upper boundary; 4) At least 2 correct checks will be repeated until valid data is input at which point the loop exits; 5) Output suitable error message(s) under appropriate circumstances based upon at least 2 correct checks; **R.** if message is displayed when it should not be **Max 4** if any errors	**5**

22	**2**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **13.1**, including prompts on screen capture matching those in code.
		Code for **13.1** must be sensible.
		
		Screen capture showing:
		
		`Enter your choice: E`
		`Enter line number of code to edit: Q`
		`Not a valid number`
		`Enter line number of code to edit: 22`
		`Not a valid line number`
		`Enter line number of code to edit: 0`
		`Not a valid line number`
		`Enter line number of code to edit: 2`
		`  SUB  NUM1`
		`E - Edit this line`
		`C - Cancel edit`
		`Enter your choice:`
			1

23	**1**	**2 marks for AO3 (design) and 2 marks for AO3 (programming)** **Marking guidance:** **Evidence of AO3 design – 2 points:** Evidence of design to look for in response: 1) Check in `ExecuteJSR` that stack does not overwrite instruction / data; 2) Recognise that instructions for `JSR` should only be executed if no error; **Note:** AO3 (design) points are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not and regardless of whether the solution works. **Evidence of AO3 programming – 2 points:** Evidence of programming to look for in response: 3) Correct value for number of program lines passed into subroutine // check that memory location pointed to by `TOS` is empty; 4) `ReportRunTimeError` called with suitable message in appropriate place; **Max 3** if code does not function correctly	**4**

23	**2**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **14.1**, including prompts on screen capture matching those in code.
		Code for **14.1** must be sensible.
		
		`****** Frame 7 ************************************************`
		`*  Current Instruction Register:  JSR 7`
		`Run time error:  Memory Address Error`
		`Stack contents:`
		` ----`
		`|  3 |`
		`| 14 |`
		` ----`
		`Execution terminated`
			1

24	**1**	**3 marks for AO3 (design) and 9 marks for AO3 (programming)**	**12**

Level	Description	Mark Range
3	A line of reasoning has been followed to arrive at a logically structured working or almost fully working programmed solution. All of the appropriate design decisions have been taken. The last line of source code may not be displayed correctly (if last line not moved due to exclusive boundary).	9–12
2	There is evidence that a line of reasoning has been partially followed. There is evidence of some appropriate design work. The subroutine `EditSourceCode` has been amended and has some added functionality.	5–8
1	An attempt has been made to amend the subroutine `EditSourceCode`. Some appropriate programming statements have been written. There is little evidence to suggest that a line of reasoning has been followed or that the solution has been designed. The statements written may or may not be syntactically correct and the subroutines will have very little or none of the extra required functionality. It is unlikely that any of the key design elements of the task have been recognised.	1–4

Marking guidance:

Evidence of AO3 design – 3 marks:

Evidence of design to look for in response:

1) Adjust the number of lines stored in `SourceCode` (ie update `SourceCode[0]`)
2) Loop through program lines consecutively or equivalent
3) Move program lines after specified location in `SourceCode`

Note: AO3 (design) points are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not and regardless of whether the solution works.

A. design evidence in option `D` or `I` code.

Evidence of AO3 programming – 9 marks:

Evidence of programming to look for in response:

4) Insert `D` / `I` / both in addition to existing options in the menu and add `D` / `I` / both to conditions of `WHILE` loop
5) Add selection to test for option `D` / `I` / both after `WHILE` loop
6) Use correct range to loop through program lines in both options, `D` and `I`
7) Correctly adjust the number of lines stored in `SourceCode` in both options, `D` and `I`

		8) Within loop, move line referenced by loop counter one location in correct direction in option D 9) Within loop, move line referenced by loop counter one location in correct direction in option I 10) For option I get user input of new line 11) For option I insert new line if there is space, otherwise display error message 12) Insert line entered by user in correct row of SourceCode **Max 11** if code does not function correctly	

24	**2**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **15.1**, including prompts on screen capture matching those in code.
		Code for **15.1** must be sensible.
		
		Screen capture showing (for ease of reference inserted line highlighted):
		
		`Enter your choice: E`
		`Enter line number of code to edit: 10`
		`SKP`
		`E - Edit this line`
		`D - Delete this line`
		`I - Insert a new line above this line`
		`C - Cancel edit`
		`Enter your choice: D`
		
		`0 12`
		`1 NUM1: 2`
		`2 NUM2: 5`
		`3 NUM3: -1`
		`4 NUM4: 125`
		`5 START: LDA NUM1 * test while loop`
		`6 WHILE: CMP# 12`
		`7 BEQ WEND`
		`8 ADD NUM2`
		`9 JMP WHILE`
		`10 WEND: STA NUM3`
		`11 ADD NUM4`
		`12 HLT`
			1

24	**3**	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **15.1**, including prompts on screen capture matching those in code.
		Code for **15.1** must be sensible.
		
		Screen capture showing (for ease of reference inserted line highlighted):
		
		`Enter your choice: E`
		`Enter line number of code to edit: 4`
		`       STA  FINAL`
		`E - Edit this line`
		`D - Delete this line`
		`I - Insert a new line above this line`
		`C - Cancel edit`
		`Enter your choice: I`
		`Enter the new line:`
		`LABEL: SKP`
		
		`0 12`
		`1         LDA# 3       * test negative`
		`2         SUB  NUM1`
		`3         SKP`
		`4 LABEL: SKP`
		`5         STA  FINAL`
		`6         HLT`
		`7`
		`8`
		`9`
		`10`
		`11  NUM1:      5`
		`12 FINAL:     0`
			1

Qu		Marks	
25	1	**8 marks for AO3 (programming)** **Mark as follows:** 1. Correct variable declarations for `Number1`, `Number2`, `Number`, `Count` and initialisation; **Note to examiners:** If a language allows variables to be used without explicit declaration, (eg Python), then this mark should be awarded if the correct variables exist in the program code and the first value they are assigned is of the correct data type. 2. Correct prompts `"Enter an integer: "` and `Number1` assigned integer value entered by user and `"Enter another integer: "` and `Number2` assigned integer value entered by user; 3. Correct `IF THEN ELSE` statement syntax allowed by the programming language and correct condition; 4. Correct assignments to `Number` in `THEN` and `ELSE` part; 5. Loop iterates correct number of times; 6. Correct condition to output X; 7. Correct condition to output V; 8. Correct output within loop without line feed; **I.** case and minor typos **Max 7** if code does not function correctly	**8**

25	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **04.1**.
		Code for **04.1** must be sensible.
		
		Screen capture showing:
		
		`Enter an integer: 4`
		`Enter another integer: 99`
		`////V////X////V////X////`
			1

Qu		Marks	
26	1	**7 marks for AO3 (programming)** **Marking guidance:** **Evidence of AO3 programming – 7 marks:** Evidence of programming to look for in response: 1. Constant declared and used as index (any index between 6 and 9) for `Stats`; 2. If queue length not less than 5 … // if queue length equals 5 …; 3. … Increment count of shuns in `Stats` data structure; **R**. if not within a selection structure. 4. … Output buyer number and message; **R**. if not within a selection structure. 5. … If number of tills is less than `MAX_TILLS` …; **R**. if not within a selection structure. 6. …. Increment `NoOfTills`; **R**. if not within a nested selection structure. 7. In `OutputStats` output number of total shuns with suitable message; **Max 6** if any errors	7

26	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **16.1**, including prompts on screen capture matching those in code.
		Code for **16.1** must be sensible.
		
		Screen capture showing:
		
		`The simulation statistics are:`
		`==========`
		`The maximum queue length was: 5 buyers`
		`The maximum waiting time was: 12 time units`
		`33 buyers arrived during 50 time units`
		`The average waiting time was: 4.2 time units`
		`The average queue length was: 3.26 buyers`
		`4 buyers did not need to queue`
		`4 buyers turned away because the queue was too long`
			1

Qu		Marks	
27	1	**2 marks for AO3 (design) and 6 marks for AO3 (programming)** **Marking guidance:** **Evidence of AO3 design – 2 marks:** Evidence of design to look for in response: 1. Identify the need for a loop or equivalent to initialise the till speeds/re-use loop in `ResetDataStructures`; 2. Recognise need to use `Tills` data structure to calculate serving time in `CalculateServingTime`; **Note: AO3** (design) points are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not and regardless of whether the solution works. **Evidence of AO3 programming – 6 marks:** Evidence of programming to look for in response: 3. Correctly calculate and store the default till speeds; 4. Correctly change the size of each element in `Tills`; 5. Add `Tills` to parameter list of `ChangeSettings` definition and call // add `Tills` to return value (Python) and assign in call to `ChangeSettings` in `QueueSimulator`; 6. Correctly set up loop to set the speed for each till; 7. Output suitable message to user including till number and default till speed; 8. Store till speed entered by user in correct element of `Tills`; **Max 7** if any errors	**8**

Qu		Marks	
27	2	**Mark is for AO3 (evaluate)** ****** SCREEN CAPTURE ****** Must match code from **17.1**, including prompts on screen capture matching those in code. Code for **17.1** must be sensible. Screen capture showing: `---------- 5 B5(2) B5 3 3 1 0 6 4 2 1 5 5 3 5 1 2 ** Start of queue ** *** End of queue *** ----------`	**1**

Qu		Marks	
28	1	**3 marks for AO3 (design) and 9 marks for AO3 (programming)**	12

Level	Description	Mark Range
3	A line of reasoning has been followed to arrive at a logically structured working or almost fully working programmed solution. All of the appropriate design decisions have been taken.	9–12
2	There is evidence that a line of reasoning has been partially followed. There is evidence of some appropriate design work. This is a partially working programmed solution.	5–8
1	An attempt has been made to amend the subroutine `Serving` and/or `OutputTillAndQueueStates`. Some appropriate programming statements have been written. There is little evidence to suggest that a line of reasoning has been followed or that the solution has been designed. The statements written may or may not be syntactically correct and the subroutines will have very little or none of the extra required functionality. It is unlikely that any of the key design elements of the task have been recognised.	1–4

Marking guidance:

Evidence of AO3 design – 3 marks:

Evidence of design to look for in response:

1. Attempt to test for conditions to be served at express till (in `Serving`).
2. Recognise the need for a loop to find next buyer with < 10 items (in `ServeBuyerExpress`).
3. Recognise the need to move buyer records in `BuyerQ`.

Note: AO3 (design) points are for selecting appropriate techniques to use to solve the problem, so should be credited whether the syntax of programming language statements is correct or not and regardless of whether the solution works.

Evidence of AO3 programming – 9 marks:

Evidence of programming to look for in response:

4. Correct parameters and return values for `ServeBuyerExpress`.
5. Correct conditions for finding a buyer eligible for express till within loop. **R**. if multiple buyers would be found in a single method call.
6. Extract buyer data only if a buyer with less than 10 items has been found.
7. Correctly move buyer records in `BuyerQ`.
8. Output buyer ID.
9. Call `UpdateStats`.
10. Call `CalculateServingTime` with till 0 parameter.
11. Call `ServeBuyerExpress` under correct conditions (QLength > 0 and till 0 free).
12. Till 0 stats included in `OutputTillAndQueueStates`.

Max 11 if code does not function correctly

Qu		Marks	

28	2	**Mark is for AO3 (evaluate)**
		
		****** SCREEN CAPTURE ******
		Must match code from **18.1**, including prompts on screen capture matching those in code.
		Code for **18.1** must be sensible.
		
		If Question 17 has been implemented the output should be:
		
		`----------`
		`8 B7(7)`
		`B7 0 2`
		`0 5 4 1`
		`1 2 7 3`
		`2 6 3 0`
		`** Start of queue **`
		`*** End of queue ***`
		
		If Question 17 has **not** been implemented the output should be:
		
		`----------`
		`8 B7(7)`
		`B7 0 3`
		`0 4 5 2`
		`1 1 8 1`
		`2 4 5 0`
		`** Start of queue **`
		`B6 3 25`
		`*** End of queue ***`
		`----------`
			1