

1(a) A programmer has been given this pseudocode algorithm.

```
01 do
02 data = input("Enter the code")
03 until data.length > 20
04 first = data.substring(0,1)
05 second = data.substring(1,1)
06 first = int(first)
07 second = int(second)
08 if first > second then
09 print(str(first) + "wins")
10 elseif first == second then
11 print("It's a draw")
12 else
13 print(second + "wins")
14 endif
```

The pseudocode algorithm calls different subroutines.

Name the programming constructs that start on line 01 and line 08 in the pseudocode algorithm.

01
08

[2]

(b) State the purpose of line 06 .

.....
.....

[1]

2 A programmer is designing a program that will store data.

The programmer is deciding whether to store the data in a stack or a queue.

The pseudocode function, enqueue, inserts an item into a queue.

```
01 function enqueue(item)
02     if tailPointer >= queue.length then
03         return false
04     else
05         queue[tailPointer] = item
06         tailPointer = tailPointer + 1
07         return true
08     endif
09 endfunction
```

i. Give the name of the parameter in the function enqueue.

[1]

ii. Give the name of **one** global variable that is used in the function enqueue.

[1]

iii. Describe **one** benefit and **one** drawback of using global variables instead of parameter passing in a subroutine.

Benefit -----

Drawback -----

[4]

iv. The function enqueue can be called by the main program.

Explain why the function enqueue returns true or false values, and how this can be used by the main program that calls the function.

-----[3]-----

- v. The pseudocode function, dequeue, removes and returns the first item in the queue. If the queue is empty, the function returns the string "EMPTY".

```
01 function dequeue(data)
02     if headPointer != tailPointer then
03         return "EMPTY"
04     elseif
05         value = queue[headPointer]
06         return value
07         headPointer = headPointer + 1
08     endif
09 endfunction
```

The function dequeue has **several** errors.

Identify the line number of any **three** errors **and** state the correction required.

Error 1 Line Number -----

Error 1 Correction -----

Error 2 Line Number -----

Error 2 Correction -----

Error 3 Line Number -----

Error 3 Correction -----[3]-----

- vi. The programmer has corrected all of the errors in the function dequeue.

The main program repeatedly calls the function dequeue until all of the elements in the queue have been output.

Write the main program using pseudocode or program code.

[3]

3(a) A student has written this pseudocode algorithm:

```
01 a = 12
02 do
03 b = input("Enter a number")
04 until b >= 0 and b <= 100
05 for c = 1 to a
06 print(c * a)
07 next c
```

Rewrite lines 05 to 07 to use a while loop instead of a for loop.

You should write your answer using either program code or pseudocode.

[4]

(b) The program uses variables.

i. Describe what is meant by a variable.

----- [2]

ii. Give the identifiers of all the variables used in this program.

----- [1]

(c) The student has used a do loop on line 02.

Describe the difference between a do loop and a while loop.

----- [2]

4 A text-based computer game allows a user to dig for treasure on an island. The island is designed as a grid with 10 rows and 20 columns to store the treasure. Each square is given an x and y coordinate. Some of the squares in the grid store the name of a treasure object. Each treasure object has a value, e.g. 100 and a level, e.g. "Bronze."

The main program initialises a new instance of Board. The programmer is considering declaring this as a global variable or as a local variable and then passing this into the subroutines that control the game.

Compare the use of variables and parameters in this game.

You should include the following in your answer:

- what is meant by a local variable and global variable
- how local and global variables can be used in this program
- the use of passing parameters by value and by reference.

Blank lined paper for writing.

- 5 A programmer has designed a program that includes a reusable program component.

The reusable program component is a function called `isInteger ( )`. This will take a string as an argument and then check that each digit is between 0 and 9. For example if 103 is input, it will check that the digits 1, 0 and 3 are each between 0 and 9.

The `asc ( )` function returns the ASCII value of each digit. For example `asc ("1")` returns 49.

The ASCII value for 0 is 48. The ASCII value for 9 is 57.

```
01  function isInteger(number)
02      result = true
03      for count = 0 to number.length-1
04          asciiValue = asc(number.substring(count, 1))
05          if not(asciiValue >= 48 and asciiValue <= 57) then

06              result = false
07          endif
08      next count
09      return result
10  endfunction
```

- i. Identify **one** identifier used in the function `isInteger ( )`.

-----[1]

- ii. Give the line number where the branching (selection) construct starts in the function `isInteger ( )`.

-----[1]

- iii. Give the line number where the iteration construct starts in the function `isInteger ( )`.

-----[1]

6 A recursive pseudocode function, recursiveAlgorithm(), is shown.

```
01  function recursiveAlgorithm(value)
02      if value <= 0 then
03          return 1
04      elseif value MOD 2 = 0 then
05          return value + recursiveAlgorithm(value - 3)
06      else
07          return value + recursiveAlgorithm(value - 1)
08      endif
09  endfunction
```

Describe the key features of a recursive algorithm.

You may refer to the function, recursiveAlgorithm() in your answer.

[3]

7 A programmer uses an Integrated Development Environment (IDE).

Complete the table by identifying **and** describing **three** IDE features that can help the programmer to develop, or debug a program.

IDE feature	Description

[6]

8 Layla writes a pseudocode algorithm to:

- input 20 positive numbers into a 0-indexed 1-dimensional array
- output the average (mean) number as a decimal
- output the smallest number
- output the largest number.

The pseudocode algorithm is shown. It contains various errors.

```
01 total = 1
02 smallest = 9999
03 largest = -1
04 for x = 0 to 21
05     dataArray[x] = input("Enter a number")
06     total = total + dataArray[x]
07     if dataArray[x] < largest then
08         largest = dataArray[x]
09     endif
10     if dataArray[x] < smallest then
11         smallest = dataArray[x]
12     endif
13 next x
14 print("Average = " + total * 20)
15 print("Smallest = " + smallest)
16 print("Largest = " + largest)
```

dataArray is defined as a local variable within the main program.

- i. State what is meant by a 'local variable'.

----- [1]

- ii. Give **one** benefit and **one** drawback of declaring dataArray as a local variable in the main program.

Benefit -----

Drawback -----
----- [2]

9 Nina is writing a computer game using an Integrated Development Environment (IDE). Her friend, James, is writing a computer game using a text-editor which will allow James to create and edit text. James will use a separate compiler.

Discuss the differences between writing and debugging a program using an IDE and a text-editor.

You should include the following in your answer:

- features that are used when writing code
- features that are used when debugging code
- the benefits of using an IDE instead of a text-editor.

10 Layla writes a pseudocode algorithm to:

- input 20 positive numbers into a 0-indexed 1-dimensional array
- output the average (mean) number as a decimal
- output the smallest number
- output the largest number.

The pseudocode algorithm is shown. It contains various errors.

```
01 total = 1
02 smallest = 9999
03 largest = -1
04 for x = 0 to 21
05     dataArray[x] = input("Enter a number")
06     total = total + dataArray[x]
07     if dataArray[x] < largest then
08         largest = dataArray[x]
09     endif
10     if dataArray[x] < smallest then
11         smallest = dataArray[x]
12     endif
13 next x
14 print("Average = " + total * 20)
15 print("Smallest = " + smallest)
16 print("Largest = " + largest)
```

Identify **two** variables used in this algorithm.

1

2

- 11 A card game uses a set of 52 standard playing cards. There are four suits; hearts, diamonds, clubs and spades. Each suit has a card with a number from; 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13.

The card game randomly gives 2 players 7 cards each. The unallocated cards become known as the deck.

The players then take it in turns to turn over a card. A valid move is a card of the same suit or the same number as the last card played.

The winner is the first player to play all of their cards.

A function, `checkValid()`, takes the card the player has selected, and the last card played as parameters.

It returns `true` if the player's move is valid and returns `false` if the player's move is not valid.

- i. State the reason why `checkValid()` is a function and not a procedure.

----- [1]

- ii. The programmer will use a branching (selection) construct to make decisions.

Describe the decisions that will be made in the `checkValid()` function and how these change the return values.

----- [3]

12 A program uses the recursive function calculate (). The function is written in pseudocode.

```
1.function calculate(number : byVal)
2.  if number == 1 then
3.    return number
4.  else
5.    return number + calculate (number - 1)
6.  endif
7.endfunction
```

i. Give the line number in the algorithm calculate () where a recursive call is made.

[1]

ii. State **two** features of any recursive algorithm.

Feature 1

Feature 2

[2]

- 13 Describe **one** feature of an Integrated Development Environment (IDE) that can be used to help write a program **and one** feature that can be used to help test a program.

Write

.....

.....

.....

.....

Test

.....

.....

.....

.....

[4]

- 14 Given the following procedure:

```
01 procedure generate(number)
02     a = 0
03     while number > 0
04         if number MOD 2 == 0 then
05             a = a + 2
06             print(a)
07             number = number - 2
08         else
09             a = a + 1
10             print(a)
11             number = number - 1
12         endif
13     endwhile
14 endprocedure
```

State the values printed by the procedure generate when number = 8.

.....

.....

[1]

15(a) Hugh has written a recursive function called `thisFunction()` using pseudocode.

```
01 function thisFunction(theArray, num1, num2, num3)
02   result = num1 + ((num2 - num1) DIV 2)
03   if num2 < num1 then
04     return -1
05   else
06     if theArray[result] < num3 then
07       return thisFunction(theArray, result + 1, num2, num3)
08     elseif theArray[result] > num3 then
09       return thisFunction(theArray, num1, result - 1, num3)
10     else
11       return result
12   endif
13 endif
14 endfunction
```

The function DIV calculates integer division, e.g. $5 \text{ DIV } 3 = 1$

theArray has the following data:

Index:	0	1	2	3	4	5	6	7
Data:	5	10	15	20	25	30	35	40

Trace the algorithm, and give the final return value, when it is called with the following statement:

```
thisFunction(theArray, 0, 7, 35)
```

You may choose to use the table below to give your answer.

Function call	num1	num2	num3	result
thisFunction(theArray, 0, 7, 35)				

Final return value [5]

(b) State the name of the standard algorithm thisFunction() performs.

..... [1]

(c) Hugh could have written thisFunction() using iteration instead of recursion.

Compare **two** differences between recursion and iteration.

1

.....

.....

.....

.....

2

.....

.....

.....

[4]

(d) The recursive function `thisFunction()` is printed again here for your reference.

```
01 function thisFunction(theArray, num1, num2, num3)
02   result = num1 + ((num2 - num1) DIV 2)
03   if num2 < num1 then
04     return -1
05   else
06     if theArray[result] < num3 then
07       return thisFunction(theArray, result + 1, num2, num3)
08     elseif theArray[result] > num3 then
09       return thisFunction(theArray, num1, result - 1, num3)
10     else
11       return result
12     endif
13   endif
14 endfunction
```

Rewrite the function `thisFunction()` so that it uses iteration instead of recursion.

You should write your answer using pseudocode or program code.

[6]

16(a) A recursive function, GCD, is given in pseudocode.

```
function GCD (num1, num2)

    if num2 == 0 then

        return num1

    else

        return GCD(num2, num1 MOD num2)

    endif

endfunction
```

Trace the recursive function when it is called by the statement `GCD(250, 20)`. Give the final value returned.

Final return value: _____

[3]

(b) The function uses branching.

i. Identify the type of branching statement used in the function.

-----[1]

ii. Explain the difference between branching and iteration.

-----[2]

iii. Identify the **two** parameters in the function.

1 -----

2 -----

[1]

iv. State whether the parameters should be passed by value, or by reference. Justify your answer.

-----[2]

v. Describe the arithmetic operation of MOD. Use an example in your answer.

[2]

- 17 A recursive function, generate, is shown.

```
function generate(num1:byval)
    if num1 > 10 then
        return 10
    else
        return num1 + (generate(num1 + 1) DIV 2)
    endif
endfunction
```

Trace the algorithm to show the value returned when generate(7) is called. Show each step of your working.

[illegible]

[6]

END OF QUESTION PAPER

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
1	a		1 mark each to max 2: • 01 – Iteration // Looping • 08 - Branching // Selection	2	Do not accept if statement, for loop etc. Type of construct not example of construct required. <u>Examiner's Comments</u> Most candidates correctly identified iteration and selection, but some gave these structures in reverse order.
	b		1 mark for: • To change the data type of first to an integer • To cast the data type of first as an integer	1	<u>Examiner's Comments</u> Most candidates identified casting to an integer.
			Total	3	
2		i	item	1	Allow queue can be circular or linear, stack can only be linear <u>Examiner's Comments</u> The vast majority of candidates identified that item was the paramete
		ii	tailPointer	1	Allow queue <u>Examiner's Comments</u> Many candidates identified that tailPointer was an example of a global variable in the code.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
		iii	1 mark for benefit and 1 mark for expansion: e.g. • Simpler to program ... • ... because values do not need to be passed/renamed/moved between different subroutines • Do not need to worry about returning values // do not need to decide between byval/byref ... • ... all parts of the program can access the (same) value (in the same way) 1 mark for drawback and 1 mark for expansion: • Uses more memory ... • ... because the memory space is declared when the program starts and remains in use throughout • Makes testing / debugging more difficult ... • as it's difficult to test an individual block of code • Reduces data accuracy / integrity ... • changing a global variable may have an impact on another module	4	2 marks max for benefit 2 marks max for drawback <u>Examiner's Comments</u> The majority of candidates struggled to describe the implications of using global variables, and this highlighted a lack of appreciation of parameter passing and variable scope. At AS Level candidates should become more adept at using parameters and develop an understanding of why this is beneficial – both in theoretical terms and within their personal programming practice. There were few responses that were able to give complete descriptions.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
		iv	1 mark per bullet to max 3 - False is returned if the queue is full the main will not attempt to add another item - True is returned if the item is successfully added so the main program can try to add another item	3	Examiner's Comments Some candidates thought that if enqueue() returned True it meant that there would be space to add additional values. This was not the case; it meant that the item had been enqueued successfully, but it could have been the last item that then filled the queue to capacity. Sometimes candidates did explain the reasons why True and False would be returned but were then unable to go on to say how they could be used. Good responses were seen such as 'the main program can give an error message if the operation didn't work if the queue was full'. Exemplar 1 The enqueue() function returns "False" in the case that the queue is full (ie the tail pointer is greater than or equal to the predetermined length) and the return indicates the item has not been added. The function returns "True" if the queue is not full and indicates the item has been added. These return values can be used to check whether the item passed to enqueue() has been successfully added by assigning them to a variable: successful = enqueue("A") for example. [3] This response showed clear understanding of the need to store the result returned by dequeue() to be used in the conditional loop test.
		v	1 mark for error and correction to max 3 - Line 01 There is no need to pass in a parameter - Line 02 has != instead of == - Line 04 has elseif and no condition // replace with else - Line 06 should be headpointer = headpointer + 1 // Swap lines 06 and 07 - Line 07 should be return value	3	Do not award marks for the line number. The error must be related to the line number stated. Examiner's Comments Many candidates identified at least one logical error, but few were able to identify three errors. The most common error identified was the error in line 02.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
		vi	1 mark per bullet to max 3 • Use of loop to call dequeue().... • ...to output the return value... • ...until "EMPTY" is returned e.g. <pre> elementsLeft = true while elementsLeft == true value = dequeue() if not(value = "EMPTY") then print(value) else elementsLeft = false endif endwhile </pre>	3	<u>Examiner's Comments</u> Some candidates mistakenly gave a definition for the dequeue() function instead of answering the question set. Candidates need to be mindful to read the question carefully. Some candidates used a count-controlled loop whereas a conditional loop was required. Those who made a good attempt at a response often did not assign the result of dequeue () to a variable, so could not effectively use the return value from the function as well as print the value returned.
			Total	15	

Mark Scheme

Question		Answer/Indicative content	Marks	Guidance
3	a	1 mark each - Initialisation of counter variable before loop - Condition (e.g. while c <= a) - c incremented in loop - Completed loop will produce correct results 12, 24 ... 144 e.g. <pre> c = 1 c = 0 c = 0 while c <= a while c < a while c <> a print(c * a) c = c + 1 c = c + 1 c = c + 1 print(c * a) print(c * a) endwhile endwhile endwhile </pre>	4	BP2 – Allow any suitable logic for the while loop condition that iterates between 1 and a. Allow != for <> Allow += or equivalent for c = c + 1 Allow hard coded values for upper bound such as a = 12 or a = 13 depending on the relational operator used. No marks awarded if a conditional loop has not been used. Max 3 if solution does not completely work. Examiner's Comments This question was generally well attempted with marks relatively evenly distributed throughout the range of marks available. There were some off-by-one errors, with the loop counter or the position of the output line being incorrectly positioned before/after the counter increment. Incorrect loops such as count-controlled for loops were rejected. If responses did not produce a fully correct output, marks given were limited to 3 marks. Exemplar 1 <pre> ~~while c < a~~ c = 1 while c < a: print(c * a) c = c + 1 </pre> This response showed a typical off-by-one error where the candidate had not thought through the logic of the entire response. The condition operator should have been <= rather than < to iterate through all values of c from 1 to 12.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
	b	i	1 mark each • A memory location // named storage location • Stores/holds data / a value • That can be changed	2	Element/identifier on its own is not enough for BP1 <u>Examiner's Comments</u> Most candidates gained some credit for identifying that a variable holds a value, and this was the most popular response. Element/identifier on its own was insufficient for the first mark point as there had to be a clear indication that the identifier referred to a memory/storage location.
		ii	1 mark for all variables: a b c	1	Must have all three for the mark to be awarded. <u>Examiner's Comments</u> The majority of candidates had little difficulty correctly identifying the variables given in the code.
	c		1 mark for do loop and 1 mark for while loop up to a maximum of 2 marks. • While loop will check the condition at the start of the loop // pre-condition loop • Do loop will check the condition at the end of the loop // post-condition • The code in a while loop may never run (if the condition is already met) • The code in a do loop will always run at least one.	2	Answer must cover both do loop and while loop for 2 marks to be awarded BP1 and BP2 must be specific as to the location that the condition is placed <u>Examiner's Comments</u> Many responses were vague and did not accurately identify or distinguish that a while loop is a pre-condition loop while a do loop is a post-condition loop. Responses had to be clear as to the relative position of the condition in the loop. Accepted responses included a while loop is a pre-condition loop while a do loop is a post-condition loop. Fewer candidates identified that the body of a while loop may not be executed while the body of a do loop will always be executed at least once.
			Total	9	

Mark Scheme

Question	Answer/Indicative content	Marks	Guidance
4	Mark Band 3 – High level (7–9 marks) The candidate demonstrates a thorough knowledge and understanding of parameters and local/global variables; 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. <i>There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated.</i> Mark Band 2 – Mid level (4–6 marks) The candidate demonstrates reasonable knowledge and understanding of parameters and local/global variables; 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 provides a reasonable discussion, the majority of which is focused. Evaluative comments are, for the most part appropriate, although one or two opportunities for development are missed. <i>There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence.</i> Mark Band 1 – Low Level (1–3 marks) The candidate demonstrates a basic knowledge of parameters and local/global variables with limited understanding shown; the material is basic and contains some inaccuracies. The candidates makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgements if made are weak and unsubstantiated. <i>The information is basic and communicated</i>	9	AO1: Knowledge and Understanding Indicative content - Local variable can only be accessed within sub-program/main program it is declared within - Global variable can be accessed by all sub-programs - Parameters are items passed to a subproblem - Passing by reference sends a pointer to the original value, so this will be changed when control is returned - Passing by value sends the a copy of the value, so the original will not be changed when control is returned AO2: Application - If board is local it can only be accessed in the main program - This will need to be passed to any sub-programs that need to use it - If the board needs to be changed it will need passing by reference, so that the board is updated - If it only needs to be accessed and not changed it can be passed by value AO3: Evaluation - If global then this would be present in memory throughout hence using more memory ...however the board will be required throughout the program so may be as efficient as passing it through parameters ...if global then the programming may be more straight forward, and less likely to have errors with passing the board incorrectly to subprograms, i.e. it may not be updated when it needs to be - Using local means that the board can be manipulated by subprograms without affecting the actual board if needed, for example to simulate potential changes. <u>Examiner's Comments</u> A good range of Level 2 responses were observed with competent descriptive definitions of local versus global and

Mark Scheme

Question	Answer/Indicative content	Marks	Guidance
	<i>in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear.</i> 0 mark No attempt to answer the question or response is not worthy of credit.		byVal/byRef parameter passing. There were far fewer high level AO3 evaluations of relative memory use within the context of the scenario when passing byVal/byRef for a relatively large grid object. Good, contextualised responses took the elements of the scenario in section B of the paper and talked about the grid object and passing parameters like x and y coordinates. Responses such as this were relatively rare. Assessment for learning Candidates would benefit from producing solutions centred around the scenario presented in Section B of the paper. Having extensive OOP programming experience in a relevant high level language will help candidates to successfully tackle questions where algorithms have to be presented. Sample Python code based on Section B is presented below for consideration. Assessment for learning # 2023 Section B <pre>class Treasure: def __init__(self, pValue: int, pLevel: str): self.__value = pValue self.__level = pLevel def getValue(self): return self.__value def setValue(self,pValue: int): return self.__pValue def getLevel(self): return self.__level def setLevel(self,pLevel: str): return self.__pLevel class Board(): def __init__(self): self.__grid = [[Treasure(-1,"") for _</pre>

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
					<pre> in range(20)] for _ in range(10)] def getGridItem(self,x: int, y: int): return self.__grid[x][y] def setGridItem(self,x: int,y: int, pTreasure: Treasure): self.__grid[x][y] = pTreasure def guessGrid(b: Board): row = int(input("Row: ")) col = int(input("Col: ")) t = b.getGridItem(row, col) if t.getLevel() == "": print("No treasure") else: print(f"Treasure found at {row} {col}!") print(f"Level {t.getLevel()} Value {t.getValue()}") # Test code - Treasure @ 2,2 all other locations no treasure island = Board() prize = Treasure(5,"Prize!") island.setGridItem(2,2,prize) guessGrid(island) </pre>
			Total	9	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
5		i	1 mark for: • <code>isInteger</code> • <code>number</code> • <code>result</code> • <code>count</code> • <code>asciiValue</code>	1	Penalise excessive spaces in identifiers such as <i>ascii Value</i> instead of <i>asciiValue</i> <u>Examiner's Comments</u> This question was generally well done (although slightly less well done than parts (a) (ii) and (a) (iii)), but many candidates did very well. The most common erroneous responses were giving the names of predefined functions/properties or giving relational operators.
		ii	(0)5	1	<u>Examiner's Comments</u> This question required an exact answer only and was answered correctly by the majority of candidates.
		iii	(0)3	1	<u>Examiner's Comments</u> This question also required an exact answer only and was answered correctly by the majority of candidates.
			Total	3	
6			1 mark each to max 3 • The function calls itself.... • such as line 05 / 07 • Each recursive call will create a new copy of the values in the function.... • and add all of the values of the copy the call is being made from to a stack • There is a base case // condition that stops the recursive calls... • ...condition in line 02 • There may be more than one base case	3	Allow answers in context as long as they are clear what the features are. <u>Examiner's Comments</u> Weaker responses were limited to one or two points for 'calls itself' with example such as 'line 05'. Fewer candidates went on to identify the requirement for a base case. A variety of terminology was used for the base case such as terminating case and stopping case. If candidates were clearly referring to the case where the recursive calls stopped, and the recursion started to unwind, the mark was given.
			Total	3	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
7			1 mark for identification, 1 for description of feature e.g. • Error diagnostics • ... to locate and fix errors • Breakpoints • ...stop a program running at a point to check variables • Syntax highlighting • ... to identify key words, variables and help identify syntax errors • Stepping // step through • ... run the program line by line to check variable values at each stage • Variable watch window • ...view how variables change while the program executes • Auto-complete • ... start typing a command/identifier and it completes it	6	Consider awarding description without feature. Allow other suitable answers. <u>Examiner's Comments</u> Most candidates scored well on this question which lent itself to a wide variety of potential answers. Some candidates quoted 'autocorrect' rather than auto complete or suggested predictive code. IDEs can have integrated run-time environments and translation software built-in or attached, but some responses were rather vague or included repetition and did not make it clear that they were referring to an integrated system. Sometimes the descriptive expansion or the initial identification of the point was vague and required the identification plus the description boxes to be taken in combination to give the mark. This was particularly evident when candidates gave a very vague 'debugger/debugging' identification.
			Total	6	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
8		i	it can only be accessed within the subroutine//block in which it is declared	1	<u>Examiner's Comments</u> Few candidates were able to clearly define the term 'local variable'. The concept of scope of variable s appeared to be poorly understood, with few able to define that a local variable's scope was that of the function/procedure in which it was declared.
		ii	1 mark for benefit e.g. - Increases data integrity - More efficient memory usage - Stops other subroutines accidentally altering variable 1 mark for drawback e.g. - Cannot be accessed directly by other subroutines - It has to be passed into a subroutine as a parameter	2	<u>Examiner's Comments</u> Some candidates confused the terms local variable and global variable and gave definitions the wrong way round. A significant number of responses demonstrated conceptual misconceptions. The contents of dataArray could be used in other parts of the program if it was passed as a parameter to a function/procedure, but could not be referenced directly. Responses such as giving 'can't be used anywhere else in the program' as a disadvantage were, therefore, incorrect.
			Total	3	

Mark Scheme

Question	Answer/Indicative content	Marks	Guidance
9	Mark Band 3–High Level (7–9 marks) The candidate demonstrates thorough knowledge and understanding of IDEs; 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 IDEs; 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 provides a reasonable discussion, the majority of which is focused. Evaluative comments are for the most part appropriate, although one or two opportunities for development are missed. 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 IDEs, with limited understanding shown; 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 a limited discussion which is narrow in focus. Judgments if made are weak and unsubstantiated. The information is basic and	9	AO1: Knowledge and Understanding e.g. IDE: • pretty print / syntax highlighting • auto-complete • auto-correction • breakpoints • stepping Editor: • no helpful writing/debugging features • no excess features/interface AO2.1: Application e.g. IDE • identify syntax errors as writing • ...saves trying to find them • easier debugging as can step through a program • auto-indenting avoids errors from incorrect indentation • May have built in unit testing to automate testing and avoid new errors being introduced. Editor • does not offer suggestions on code corrections • Has a lower footprint on memory/CPU which may be suited to quick alterations or working on lowed spec'd systems. • May be better when learning to program as it forces the user to type everything in full / doesn't give suggestions, helping things stick in memory. AO3.3: Evaluation e.g. • IDE is helpful in reducing original errors • IDE is helpful in finding and correcting errors • Editor is a simpler system to use e.g. less memory needed to run it, does not try and auto-correct incorrectly or introduce errors that the programmer has not made.

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
			communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear. 0 mark No attempt to answer the question or response is not worthy of credit.		<u>Examiner's Comments</u> A pleasing number of candidates could identify and describe a range of features of an IDE, with most being able to demonstrate sufficient knowledge to achieve mark band Level 2. Few candidates displayed a level of evaluative reasoning of sufficient depth to achieve mark band Level 3. More successful responses clearly explained how relevant debugging tools in an IDE led to much greater productivity than a text editor because of the way that they could assist with finding logical errors in code through the use of stepping and tracing. An acknowledgement of the amount of system resources and the complexity of some IDE environments was less often evaluated. Candidates need to focus on making clear logical arguments with evidence of evaluation (AO3) to be able to achieve mark band 3.
			Total	9	
10			1 mark each to max 2 • total • smallest • largest • x • dataArray	2	<u>Examiner's Comments</u> Most candidates correctly identified two variables from the given code.
			Total	2	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
11		i	It returns a value	1	<u>Examiner's Comments</u> Most candidate correctly identified that <code>checkValid()</code> was a function because it returned a value. Some candidates erroneously stated 'because it has an output' which did not differentiate a function from a procedure since both can print an output as a side effect.
		ii	1 mark per bullet • If the players card is the same suit return true • if the players card is the same number return true • if neither is true, return false	3	Allow 1/0, 'T'/'F', "Yes"/"No" or any sensible alternative as return values. <u>Examiner's Comments</u> Many candidates were able to think computationally to identify the points in a solution where a decision had to be taken. Occasionally some candidates did not specify the specific result that would be returned from the function after determining if a card was valid or not. There were also occurrences of incorrect logic such as returning valid using an AND condition on the same number and same suit clauses, rather than executing them in sequence.
			Total	4	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
12		i	Line number 5	1	<u>Examiner's Comments</u> Nearly all candidates gave the correct answer line 5.
		ii	1 mark per feature • A function that calls itself // a function that is defined in terms of itself • ...has a base case (that terminates the recursion)	2	<u>Examiner's Comments</u> Most candidates knew that a recursive algorithm is self-referential and calls itself, but some candidates were too vague specifying that it 'calls a function'. Candidates often found it harder to gain the second mark and some gave answers not related to the question such as explanations of how recursion uses stack frames during execution. Technical vocabulary is important, and some candidates did not make it clear that recursion has a base case. Those that stated a stopping/terminating condition needed to qualify their response to say that these conditions stopped/halted the recursion. Where candidates just wrote 'stopping condition' it was too vague as it was unqualified since they could have been talking about any conditional loop.
			Total	3	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
13			1 mark per identification 1 mark for expansion, max 2 each. Write: e.g. • Auto-complete • Start typing an identifier/command and it fills in the rest • Auto-indent • Indents code automatically within structures to avoid errors • Coloured command words // pretty printing // syntax highlighting • Shows which commands are correct // help identify key elements Test e.g. • Breakpoints • Stop the program running at a set point to check variables • Variable watch window • Display the values of the variables while the program is run • Stepping • Run one line at a time and check variables Unit Testing • Automated tests to be run to check changes ensure changes haven't introduced errors.	4	<u>Examiner's Comments</u> Most candidates achieved at least 2 marks by identifying two suitable features, but often the expansions to describe each feature were less clear. Developments in modern IDEs meant that there were many valid features, but auto-completion and breakpoints were particularly common answers.
			Total	4	
14			• 2,4,6,8	1 (AO3.3) (1)	<u>Examiner's Comments</u> In many cases candidates gave the response 2 and did not iterate through the loop until the initial loop condition was false.
			Total	1	

Mark Scheme

Question		Answer/Indicative content	Marks	Guidance																									
15	a	1 mark per bullet • Calculation of result to 3 • Call with <code>thisFunction(theArray, num1=4, num2=7, num3=35)</code> • Result = 5 • call with <code>thisFunction(theArray, num1=6, num2=7, num3=35)</code> • (Result = 6) return of value 6 <table border="1"> <thead> <tr> <th>Function call</th><th>num1</th><th>num2</th><th>num3</th><th>result</th></tr> </thead> <tbody> <tr> <td><code>thisFunction(theArray, 0, 7, 35)</code></td><td>0</td><td>7</td><td>35</td><td>3</td></tr> <tr> <td><code>thisFunction(theArray, 4, 7, 35)</code></td><td>4</td><td>7</td><td>35</td><td>5</td></tr> <tr> <td><code>thisFunction(thisArray, 6, 7, 35)</code></td><td>6</td><td>7</td><td>35</td><td>6</td></tr> <tr> <td></td><td></td><td></td><td></td><td></td></tr> </tbody> </table>	Function call	num1	num2	num3	result	<code>thisFunction(theArray, 0, 7, 35)</code>	0	7	35	3	<code>thisFunction(theArray, 4, 7, 35)</code>	4	7	35	5	<code>thisFunction(thisArray, 6, 7, 35)</code>	6	7	35	6						5 AO2.1 (3) AO2.2 (2)	
Function call	num1	num2	num3	result																									
<code>thisFunction(theArray, 0, 7, 35)</code>	0	7	35	3																									
<code>thisFunction(theArray, 4, 7, 35)</code>	4	7	35	5																									
<code>thisFunction(thisArray, 6, 7, 35)</code>	6	7	35	6																									
	b	Binary search	1 AO2.1 (1)																										
	c	1 mark per bullet to max 4, e.g. • Recursion uses more memory... • ...iteration uses less memory • Recursion declares new variables //variables are put onto the stack each time... • ...iteration reuses the same variables • Recursive can run out of memory/stack space... • ...while iteration cannot run out of memory • Recursion can express a problem more elegantly // in fewer lines of code... • ...while iteration can take more lines of code // be harder to understand • Recursion will be self-referential // will call itself... • ... whereas iteration does not	4 AO1.1 (2) AO1.2 (2)																										

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
	d		1 mark per bullet to max 6 • Retains function call • Uses a loop • ...that will loop until all elements inspected or value found • Updates num1 appropriately • Updates num2 appropriately • Returns -1 in the correct place if the value has not been found • Returns the result in the correct place if the value has been found e.g. <pre>function thisFunction(theArray, num1, num2, num3) while (true) result = num1 + ((num2 - num1) DIV 2) if num2 < num1 then return -1 else if theArray[result] < num3 then num1 = result + 1 elseif theArray[result] > num3 then num2 = result - 1 else return result endif endif endwhile endfunction</pre>	6 AO2.2 (3) AO3.1 (3)	
			Total	16	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
16	a		1 mark per bullet to max 3 - Num2 != 0 therefore return GCD(20,10) - Num2 != 0 therefore return GCD(10,0) - Final return value = 10	3 AO2.1 (1) AO2.2 (2)	Allow FT for numerical errors <u>Examiner's Comments</u> Many candidates successfully traced the algorithm given.
	b	i	<u>if</u>	1 AO1.1 (1)	<u>Examiner's Comments</u> A number of candidates overthought the answer and did not identify the <i>if</i> branching statement used in the code.
		ii	1 mark per bullet - Branching decides which code is run / only runs code once - Iteration repeatedly runs the same code in the same sequence	2 AO1.2 (2)	<u>Examiner's Comments</u> Many candidates found difficulty in defining either branching or iteration clearly. Often the two terms were confused, with the clear differences between branching and iteration not expanded on.
		iii	num1, num2	1 AO2.1 (1)	Exact identifier names required <u>Examiner's Comments</u> Most candidates had little difficulty correctly identifying the parameters num1 and num2.
		iv	1 mark per bullet - By Value ... the original values do not need to be modified ... byRef would not work / would cause the routine to crash	2 AO2.2 (2)	<u>Examiner's Comments</u> Most candidates identified that the parameters should be passed by value, but fewer could successfully justify their choice, and hence show a deeper understanding of the reason for the choice they made.
		v	1 mark per bullet - Gives the remainder after division - E.g. 10 MOD 3 = 1	2 AO1.1 (1) AO1.2 (1)	<u>Examiner's Comments</u> Most candidates could successfully define the MOD function. A few confused it with the absolute() function or with integer division, and some did not illustrate their answer with the required example.
			Total	11	

Mark Scheme

Question			Answer/Indicative content	Marks	Guidance
17			1 mark per bullet for working to max 6 • generate(7) return 7 + (generate(8) DIV 2) • generate(8) return 8 + (generate(9) DIV 2) • generate(9) return 9 + (generate(10) DIV 2) • generate(10) return 10 + (generate(11) DIV 2) • generate(11) return 10 • Rewinding: return 10 + (10 DIV 2) = 10 + 5 = 15 • return 9 + (15 DIV 2) = 9 + 7 = 16 return 8 + (16 DIV 2) = 8 + 8 = 16 • return 7 + (16 DIV 2) = 7 + 8 = 15	6 AO1.2 (1) AO2.2 (5)	<u>Examiner's Comments</u> A significant number of candidates struggled to clearly present a program trace for a recursive algorithm. Those who used diagrammatic or clearly indented structures showing the recursive levels and the return values fared best. Candidates need to be encouraged to work on producing a logical layout for a recursive function trace.
			Total	6	