GCST

# 2.1 ALGORITHMS

## TOPIC WISE EXAM QUESTIONS

## ANSWERS

GCSE OCR

GCSE

# 2023

| | | | | | |
|---|---|---|---|---|---|
| 1 | (b) | | `total = num1 + num2` | 1 (AO3 2b) | Allow other logically valid responses that result in `total` storing the correct value. Accept other suitable assignment operators (e.g. ← )<br><br>e.g.<br>`total = sum(num1, num2)`<br><br>`total = num2 + num1`<br><br>`x = num1 + num2`<br>`total = x`<br><br>Ignore any values given to the variable. Ignore capitalisation and minor misspelling. Ignore superfluous code that does not affect outcome. |
| 1 | (c) | (i) | `print(12 ^ 2)` | 1 (AO2 1a) | Accept ** or other sensible operator that indicates raising to a power.<br><br>If pseudocode operator given, must be a single word/symbol (e.g. `pow`), not containing spaces. |
| 1 | (c) | (ii) | `if number MOD 2 == 0 then` | 1 (AO2 1a) | Accept % or other sensible operator that indicates modulus<br><br>If pseudocode operator given, must be a single word/symbol (e.g. `modulo`), not containing spaces |
| 1 | (c) | (iii) | `difference = measurement1 - measurement2` | 1 (AO2 1a) | Accept other sensible operator that indicates subtraction.<br><br>If a pseudocode operator given, must be a single word/symbol (e.g. `minus`), not containing spaces. |

GCSE

| | | | | | |
|---|---|---|---|---|---|
| 1 | (d) | | 1 mark each:<br>• Start is set to **3** on line **01** and **3** is output on line **03**.<br>• **2**, **1** and **0** are output on next 3 iterations with start updated **to 2, 1, 0, -1** on correct line numbers.<br>• **Finished** is output on line **06** | 3 (AO3 2c) | Ignore lines 02 and 05 in answer unless these change or output any values.<br><br>Candidate may repeat start value when unchanged, this is acceptable.<br><br>Penalise incorrect or missing line numbers or additional output once only then FT. This includes where variable change and output appear on the same line.<br><br>-1 must **not** be output for BP2<br><br>Penalise missing or incorrect output once only for BP1 and FT for missing or incorrect output for BP2<br><br>Finished may be with or without quotes. Ignore case or minor spelling error.<br><br>Max 2 marks if any incorrect output or changes to `start`.<br><br>Do not accept calculated values of start (e.g. `3-1`) |

| Line | start | Output | |
|---|---|---|---|
| 01 | 3 | | BP1 |
| 03 | | 3 | |
| 04 | 2 | | BP2 |
| 03 | | 2 | |
| 04 | 1 | | |
| 03 | | 1 | |
| 04 | 0 | | |
| 03 | | 0 | |
| 04 | -1 | | |
| 06 | | Finished | BP3 |

| | | | | | |
|---|---|---|---|---|---|
| 2 | (a) | | 1 mark each:<br>Syntax error<br>• Error in the rules/grammar (of the program language).<br>• Program does **not** (fully) **run / translate / execute / start** (BOD)<br><br>Logic error<br>• Produces incorrect / unexpected **result/output**<br>• Program **runs/does not crash** | 2 (AO1 1a) | Question asks for a definition. Examples may strengthen the response but are not acceptable by themselves.<br><br>Do not allow "error/problem in the code, does not work / does not do what designed/intended to do" for either, this applies to both.<br><br>"Error in the syntax" or "error in the logic" are NE even with examples |

| | | | | | |
|---|---|---|---|---|---|
| 2 | (b) | | Line number<br>• 02<br>Correction<br>• `for scoreCount = 0 to scores.length - 1`<br><br>Line number<br>• 03<br>Correction<br>• `total = scores[scoreCount] + total`<br>• `total = total + scores[scoreCount]`<br>• `total += scores[scoreCount]` | 4 (AO3 2c) | 1 mark for each line number correctly identified.<br>1 mark for each correction. Correction must match line number.<br><br>If wrong line number, do not mark correction. If no line number, mark correction only.<br><br>Do not penalise if response removes `-1` from `scores.length` as long as it starts at 0.<br><br>Do not penalise potential off by 1 errors for looping (Python).<br><br>Do not penalise case or minor spelling errors as long as intention is clear.<br><br>Allow description of change that would be made (e.g. "change 1 to 0")<br><br>First correction is fixing indexing error so element 0 is included. This could be done on line 03 e.g. `scores[scoreCount-1]`. Second correction is fixing addition of total.<br><br>If both errors fixed on line 03, full marks should be given. e.g. |

| | | | | | |
|---|---|---|---|---|---|
| 3 | (a) | | 1 mark each<br>• stores/holds **data/value/name/names [pos]**<br>• …so (value) can be changed / swapped / moved / overwritten / inserted<br>• …without being lost.<br>• will be assigned to `names[pos-1]` | **2**<br>(AO2 1b) | Do not allow answers that clearly refer storing the position / index (or any other out of context data) for BP1; it is the name itself that is being stored, not the position. If unclear, allow BOD.<br><br>e.g. do not allow "holds the values of the index / holds value for position of the name".<br><br>Allow FT for subsequent points. |
| 3 | (c) | (i) | 1 mark each for insertion and bubble sort, max 2<br><br>Insertion sort:<br>• inserts/moves values into **correct position**<br>• inserts value **once** (then in correct position)<br>• stops when end of array reached // completes in one **pass** through the array<br>• moves items down the array / left<br>• start of array becomes sorted first<br>• creates a sorted array **within** an array // has a sorted/unsorted **partition / section / list**<br>• starts on 2nd value<br>• more efficient/faster than bubble sort<br>• … because fewer iterations / comparisons (on average)<br>• … when data more scrambled<br><br>Bubble sort :<br>• compares/swaps **pairs** of values<br>• value is **repeatedly** moved/swapped (until in correct position)<br>• repeats if a swap has been made // needs multiple passes | **2**<br>(AO1 1b) | Answer must reference both bubble sort and insertion sort for 2 marks except if efficiency mark plus expansion given.<br><br>Allow reference to big O for efficiency discussion.<br><br>Only award efficiency once. Only award fewer iterations once<br><br>Do not accept "completes in one iteration" for insertion sort.<br><br>Accept list / data / values / etc for array.<br><br>"when data more scrambled" only makes sense when discussing efficiency/speed, do not give marks for saying that either can handle data that is more scrambled (they both can sort data however it is arranged).<br><br>Do not accept "bubble/insertion sort does not" for 2nd mark. |
| | | | • will complete a final iteration once sorted (to check for no swaps needed)<br>• moves items up the array<br>• end of array becomes sorted first<br>• moves/bubbles the highest value to the top<br>• less efficient/slower than insertion sort (on large sets of values)<br>• … more iterations / comparisons (on average)<br>• … when data more scrambled | | |
| 3 | (c) | (ii) | 1 mark each to max 2<br>e.g.<br>• Both produce a sorted list / array<br>• Both work in place / without duplicating data / without using divide and conquer<br>• Both need a temporary variable<br>• Both swap values<br>• Both use loops / iteration / repeats<br>• Both loops are nested / inside each other<br>• Both (may) need multiple passes<br>• Both use selection<br>• Both work with an array / list data structure<br>• Both work from left to right<br>• Both build up **sorted list** one item at a time (after every pass)<br>• Both compare (pairs of) **values**<br>• Both (typically) **less efficient / slower** than merge sort (or other sorting algorithms) | **2**<br>(AO1 1b) | Allow reference to both sorting / putting items into order for BP1.<br><br>"Allows sorting of numbers and strings" meets BP1<br><br>Allow answers relating to not needing additional memory as BP2.<br><br>Allow "breaking into smaller lists" as divide and conquer for BP2.<br><br>If answer is a statement (e.g. "uses loops"), assume candidate is talking about both algorithms doing this. |

| | | | | | |
|---|---|---|---|---|---|
| 5 | (c) | | 1 mark each to max 6<br><br>• Initialise / declare `score` (to zero) before use, outside of any loop<br>• Generates 2 random numbers between 1 and 10<br>• Inputs answer from user displaying suitable numbers<br>• Checks if input is correct answer…<br>• … if correct adds 1 to `score`<br>• Repeats BP2 to 5 three times (for bullet points attempted)<br>• Outputs `score` after reasonable attempt at counting | 6<br>(AO3 2b) | No need to cast data to string/integer.<br><br>If random numbers chosen, BP3 must use these. If no random numbers chosen, allow manually setting values<br><br>BP6 can be awarded for either a loop repeating 3 times or the same code written out 3 times<br><br>BP5 can be given FT if sensible attempt at BP4<br><br>Do not award BP6 if same numbers used for every question. Must pick new values each time.<br><br>Do not penalise potential off by 1 errors for looping (Python) or random number generation<br><br>Example answer<br><br>`score = 0`<br>`for count = 1 to 3`<br>`  num1 = random(1, 10)`<br>`  num2 = random(1, 10)`<br>`  ans = input("What is" +num1 +  " + " + num2 + "?")`<br>`  if ans = num1 + num2 then`<br>`  score = score + 1`<br>`  end if`<br>`next count`<br>`print("You scored " + score)` |

GCSE

2022

Decomposition
Abstraction

- Start and end/stop with all boxes connected, no boxes that do not lead to another box (no arrows needed)
- Input **three** variables using **parallelogram shape**
- Checks all three criteria (day, student, discount card) using **diamond shape(s)** with two lines from each

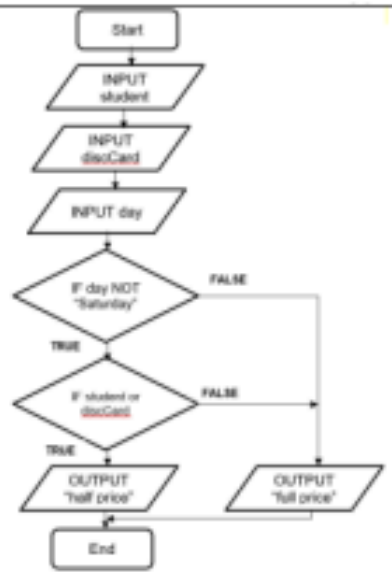
- ...**Outputs** "full price" with correct conditions using **parallelogram shape**
- ...**Outputs** "half price" with correct conditions using **parallelogram shape**

Guidance for correct outputs

| Conditions | Outcome |
|---|---|
| Not Saturday **and** (either a student or has a discount card). | Half price |
| Saturday **or** (not a student and doesn't have a discount card). | Full price |

| Saturday | Student | Discount Card | Outcome |
|---|---|---|---|
| N | N | N | Full price |
| N | N | Y | **Half price** |
| N | Y | N | **Half price** |
| N | Y | Y | **Half price** |
| Y | N | N | Full price |
| Y | N | Y | Full price |
| Y | Y | N | Full price |
| Y | Y | Y | Full price |

5
(AO3 2a)




Question asks for a flowchart. Answers as pseudocode, high level language or other forms are not acceptable 9 (NAQ).

BP 4 and 5 only to be awarded if all decisions ensure correct output and clear what the decisions are. FT for incorrect shapes used or no inputs as long as decisions are logically correct. Must attempt all three decisions.

Allow calculation of half price / full price instead of message but this must still be output.

Inputs / decisions may be presented as individual or combined boxes but must still store as three variables.

Penalise lack of parallelogram for input/output once only then FT

BOD parallelogram shapes if not sure whether input or output as long as context is clear (e.g inputs at start, outputs at end)

GCSECOMPUTERSCIENCETUTOR.COM

| | | | | | |
|---|---|---|---|---|---|
| 2 | (c) | | • Number of people (at the table) // whether there are **more** than 5 people or not<br>• Choice between percentage **and** value // actual value of both **percentage**, **value** | 2 (AO3 2a) | Ignore additional inputs that would be sensible, such as cost of the meal.<br><br>Accept inputs in form of pseudocode / high-level language.<br><br>Max 1 if other irrelevant inputs given.<br><br>"Whether to leave a tip or not" or "Amount of tip" NE for BP2. Must address both the percentage and value of tip if asked for. BOD "type of tip" for BP2 |
| 2 | (d) | (i) | • Convert/change one data **type** to another<br>• Line 03 // 3 // three | 2 (AO1 1b, AO2 2b) | **Do not** accept "change to string" – this is the use in this example but not a definition. |
| 2 | (d) | (ii) | • **Kofi2021** as staffID on line 03<br>• **Kofi2021x** as staffID on line 05<br>• **Kofi2021xx** as staffID on line 05<br>• **ID Kofi2021xx** output on line 07 as first and only output | 4 (AO3 2c) | Max 2 if incorrect order. Ignore misspelling of Kofi<br><br>Penalise lack of / errors with line numbers once then FT. Ignore capitalisation. Ignore additional lines unless outcome impacted.<br><br>`staffID` does not have space in. Output does have a space in. Penalise spaces once then FT. Do not penalise unless obvious.<br><br>Quotes around answer is OK, but do not allow quotes around partial answers, e.g. `"ID" Kofi2021xx` is incorrect. |

| Line number | `surname` | `year` | `staffID` | Output |
|---|---|---|---|---|
| 01 | Kofi | | | |
| 02 | | 2021 | | |
| 03 | | | **Kofi2021** | |
| 05 | | | **Kofi2021x** | |
| 05 | | | **Kofi2021xx** | |
| 07 | | | | **ID Kofi2021xx** |

| | | | | | |
|---|---|---|---|---|---|
| 3 | (a) | | • Merge into correct **sorted** lists of **size 2** (12 45 / -99 100 / -13 0 / -27 17)<br>• Merge into correct **sorted** lists of **size 4** (-99 12 45 100 / -27 -13 0 17) …<br>• …Merge into correct **sorted** list of **size 8** (-99 -27 -13 0 12 17 45 100) | 3 (AO2 1b) | Do not credit BP3 simply for a sorted list.<br><br>Groups of numbers must clearly be the correct size.<br><br>Do not all allow answers that show lists being merged and then sorting in place, this is incorrect. |
| 3 | (b) | | Any four bullet points for 1 mark each<br>• Select / choose / pick **middle** number (or left/right of middle as even number) and …<br>• …check if selected number is **equal to / matches** target number *(not just compare)*<br>• …if searched number is **larger**, discard left half // if searched number is **smaller**, discard right half<br>• Repeat until **number found**<br>• … or remaining list is of size 1 / 0 (number not found) | 4 (AO1 1b) | Do not allow "split the list in half" on its own as first step, this is incorrect.<br><br>Can get BP1 and 2 in one step (e.g. "check if the middle number is the one we're looking for")<br><br>For BP3, accept focussing on correct half<br><br>Repeat (BP4) must be in the context of an attempt at a binary search. Allow correct reference to recursion.<br><br>"until number is not in the list" is NE for final BP. Need to explain how this is known. |
| 3 | (c) | | 1 mark each<br>• Starting with the **first value**<br>• Checking **all values in order** | 2 (AO1 1b) | 2nd bullet point must cover both ideas of checking all of the values AND being done in order.<br><br>"Checks each value" / "one by one" / "step by step" by itself is NE, does not say in order.<br><br>Do not accept "repeat until value found" for BP2 (question says number is not in the list)<br><br>"Checks each value from beginning to end" implies order so gets both BP1 and BP2. |

| | | | | | |
|---|---|---|---|---|---|
| 4 | (c) | | • input and stores/uses value with message<br>• attempt at repeating…<br>• …correctly repeats number of times given as input<br>• …correctly take number as input within loop and calculates total of these numbers<br>• …correctly calculate an average (total/num)<br>• Output both total and average | 6<br>(AO3 2b, AO3 2c) | e.g.<br>`num = input("Enter how many numbers")`<br>`for x = 1 to num`<br>`  temp = input("Enter a number")`<br>`  total = total + temp`<br>`next x`<br>`print(total)`<br>`print(total / num)`<br><br>If flow chart used, correct shapes needed.<br><br>Allow tolerance of 1 with number of loops for BP3 with `for` loops<br><br>BP1 requires input with a message (can be two statements, e.g. print and then input or combined. Input must be stored or used.<br><br>BP3, 4, 5 must be logically correct to be credited Ignore non-initialisation of total<br><br>BP 5 can be given as FT as long as an attempt has been made at working out a total within the loop.<br><br>BP6 can be given as FT long as attempt made at total and average (not necessarily in a loop) |
| 5 | (e) | | • Inputs hours AND electric (two separate inputs), storing or using these.<br>• Checks if car is electric (IF/Select statement)…<br>• …correctly calculates and outputs price (hours * 2 // price / 2) for electric<br>• …correctly calculates and outputs price (hours * 4 // electric price * 2) for non-electric<br>• Attempt at repetition of BP1 to 4…<br>• …until 0 hours entered | 6<br>(AO3 2c) | Initialisation of price and hours not necessary, but if present hours must be non-zero for BP6 to be given.<br><br>BP5 must include all points attempted. Can still be credited if any of BP1 to 4 not attempted / incorrect.<br><br>BP6 can be given as FT even if BP5 (loop) is in the wrong place / does not include all required code<br><br>BP6 could be achieved as repeated function calls / recursion<br><br>Initial input outside of loop that is then also included within loop is fine. For example, input of hours outside of loop but input is then repeated again at end of loop.<br><br>Do not accept `while hours > 0` (could be -1)<br><br>Do not penalise answers where 0 is output when loop exits<br><br>e.g.<br>`while hours != 0`<br>`  hours = input("Enter hours")`<br>`  electric = input("enter Y for electric or N")`<br>`  if electric == "Y" then`<br>`  price = hours * 2`<br>`  elseif electric == "N" then`<br>`  price = hours * 4`<br>`  endif`<br>`  print(price)`<br>`endwhile` |

GCST

SAMPLE

| | | | | | |
|---|---|---|---|---|---|
| 2 | a | | • `input("enter first number")`<br>• `if`<br>• **`num2`**<br>• `print (num1)`<br>• `print (num2)` | 5<br>(AO3 2b) | Allow equivalent pseudocode expressions<br>Variables must not have speech marks around them |
| | b | | • use of condition controlled loop (while or do/until)…<br>• …checking condition of number larger than or equal to 0<br>• Input number from user within loop (FT if no loop)<br>• multiply number input by 2…<br>• ….output value in number | 5<br>(AO3 2b) | e.g. 1<br>store 10 in number<br>while number is greater than or equal to 0 do the following:<br>Take input from the user, store in number<br>Multiply number by 2<br>Output number<br><br>e.g. 2<br>while number >= 0<br>number = input()<br>output(number * 2)<br>Ignore non-initialisation of value used in condition for loop. |
| 4 | a | | • RebEl | 1<br>(AO2 1b) | Correct Answer Only (allow any case) |
| | b | i | • uitFr | 1<br>(AO2 1b) | Correct Answer Only (allow any case) |
| | | ii | • Taking firstname, surname and teacher or student as input<br>• Checking IF role is teacher/student (using appropriate selection)<br>• For teacher ...Generating last 3 letters of surname using appropriate string manipulation<br>• ...Generating first 2 of letters of firstname and adding to previous<br>• For student.... correctly calculating as before<br>• Correct concatenation **and** output<br><br>e.g.<br>Ask the user to input the data, store in variables firstname, surname and role.<br>Check whether the role entered is teacher. If it is, join the right 3 most letters in surname with the left 2 letters in firstname. Store this in username.<br>If it is not teacher, join the left 3 letters from firstname with the left 2 letters from surname. Store this in username.<br>Output the value in username. | 6<br>(AO3 2b) | 1 mark for each correct bullet to a maximum of 6.<br><br>If used, a flowchart should represent the bulleted steps in the answer column. |
| 6 | a | | crime bait fright victory nibble loose<br>bait crime fright victory nibble loose<br>bait crime fright nibble victory loose<br>bait crime fright nibble loose victory<br>bait crime fright loose nibble victory | 4<br>(AO2 1b) | 1 mark for each row from rows 2–5. Allow multiple swaps in one stage, where it is clear that a bubble sort has been applied. |

| | | | | | |
|---|---|---|---|---|---|
| 6 | b | | • Comparing zebra to orange<br>• Greater, so split and take right side<br>• Further comparison (1 or 2 depending on choices made)<br>• Correct identification of zebra using methodology above<br><br>e.g.<br><br>compare zebra to orange<br><br>greater, split right<br><br>compare to wind<br><br>greater, split right<br><br>compare to zebra | 4<br>(AO2 1b) | 1 mark per bullet (multiple ways through, marks awarded for appropriate comparison and creation of sub groups). |

| | | | | | |
|---|---|---|---|---|---|
| 8 | e | | <table><tr><th></th><th>x</th><th>y</th><th>output</th></tr><tr><td>MP1</td><td>15</td><td>0</td><td></td></tr><tr><td rowspan="2">MP2</td><td>14</td><td>1</td><td></td></tr><tr><td>12</td><td>2</td><td></td></tr><tr><td rowspan="3">MP3</td><td>9</td><td>3</td><td></td></tr><tr><td>5</td><td>4</td><td></td></tr><tr><td>0</td><td>5</td><td></td></tr><tr><td>MP4</td><td></td><td></td><td>5</td></tr></table> | 4<br>(AO3 2c) | one mark for first row<br><br>one mark for row 2 and 3<br><br>one mark for rows 4, 5, and 6<br><br>one mark for the correct output (the only value in the output column, in any position) |

| | | | | | |
|---|---|---|---|---|---|
| 8 | g | i | **Input**<br>• Number of hours <u>and</u> minutes<br><br>**Output**<br>• Number of minutes | 2<br>(AO3 2a) | |
| 8 | g | iii | • Takes input from the user<br>• Compares if input is larger than 120…<br>• …if true, outputs `"You played games for too long!"`<br>• …if false, outputs `"You are under your time limit!"` | 4<br>(AO3 2b) | **<u>High-level programming language / OCR Exam Reference Language response required</u>**<br><br>Do not accept pseudocode / natural English.<br><br>Example algorithm given below<br><br>`minutes = input("Enter minutes played")`<br>`if minutes > 120`<br>`  print "You played games for too long!"`<br>`else`<br>`  print "You are under your time limit!"`<br>`endif`<br><br>Accept alternative (but suitable) output messages.<br><br>Accept logical comparison of input less than or equal to 120 and appropriate True/False statements. |

GCSE

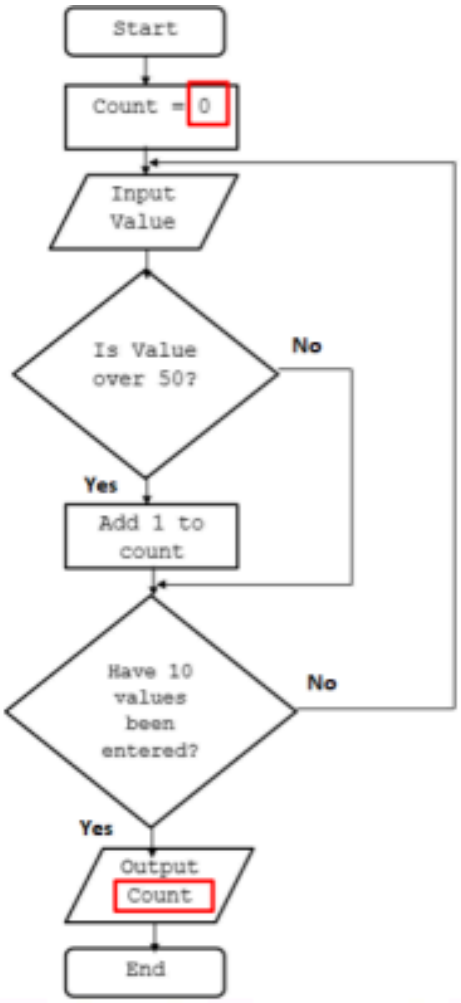
# 2021

| | | | | | |
|---|---|---|---|---|---|
| 2 | (a) | | | 1 | 1 mark per row |

| Statement | True (✓) | False (✓) |
|---|---|---|
| The list of words is initially split into a sorted set and an unsorted set | ✓ | |
| The insertion sort uses a divide stage and then a conquer stage. | | ✓ |
| The list of words must be in order before the insertion sort can start | | ✓ |
| Each word is inserted into the correct place in the array, one by one | ✓ | |
| The insertion sort will not work because the word "wall" appears twice. | | ✓ |

| | | | | | |
|---|---|---|---|---|---|
| | (b) | | • Pick middle value / pumpkin // find midpoint<br>• Compare this to house, **no match**<br>• pumpkin>house…<br>• …so discard top half of list // focus on bottom half<br>• Pick **middle value** again, either house or flour…<br>• …finds value // repeat to find value | 4 | Do not award generic responses except for BP1 Must clearly show the steps taken for this list to achieve more than 1 mark.<br><br>Do not award "splits the list in half" for BP1 or 4 – incorrect<br><br>Allow diagrams to demonstrate the process<br><br>Allow reasonable attempt at BP3 to allow access to BP4 |

| | | | | | |
|---|---|---|---|---|---|
| 3 | (a) | | • Initialises (total) as 0 (outside loop if present)<br>• Inputs a number and stores the value<br>• Adds the input **to the total** (initialised in BP1 if present)<br>• Prints the total<br>• Iterates over BP2-4 (if present)…<br>• …until total is **over** 100 | 6 | Example answer<br>`total = 0`<br>`while total <=100`<br>`  x = input("Enter a number")`<br>`  total = total + x`<br>`  print(total)`<br>`endwhile` |

| | | | | |
|---|---|---|---|---|
| (c) | (i) | • Count = **0**<br>• Output **Count**<br><br>• All non-decision boxes and YES from decision boxes linked in a sequential fashion from Start to End.<br>• NO from first decision box linked to skip over increment of count<br>• NO from second decision box linked back to INPUT<br><br>Ignore superfluous instructions as long as they do not affect the outcome of the algorithm.<br><br>BOD misspelling of `Count` as long as it is recognisable Ignore capitalisation. | 5 |  |
| | (ii) | 1 mark per bullet point, max 5<br>• Initialises a count variable to 0<br>• asks user for an input<br>• Check if input is over 50…<br>• … increment count variable if True<br>• Repeats BP 2 and 3 (if present) until 10 numbers have been entered<br>• Outputs count <u>once 10 numbers have been entered</u> | 5 | <u>Example answer</u><br><br>`count = 0`<br>`for x = 1 to 10`<br>`  value = input("enter a value")`<br>`  if value > 50 then`<br>`  count = count + 1`<br>`  endif`<br>`next x`<br>`print(count)`<br><br>Response must be in pseudocode as per question, flowcharts or structured English are NAQ. |
| | (d) | e.g.<br>• Abstraction<br>• … focussing on the important elements // ignoring elements that do not contribute to the solution // simplifying the problem<br><br>• Decomposition<br>• …breaking a problem down (into its constituent parts)<br><br>• Algorithmic thinking<br>• …set out the steps needed to solve the problem // represented in a flow chart / as pseudocode | 4 | Mark in pairs. 1 mark for name, 1 mark for description. Description must match technique (if given). |

GCST

# 2020

**2** **(a)**

| Line | Program code | Output |
|---|---|---|
| 08 | `print score` | **18** |
| 09 | `print "name"` | **name** |
| 10 | `print newscore(score,2)` | **37** |
| 11 | `print score` | **18** |

4

AO2
1b(4)

---

**3** **(c)**

1 mark per bullet

- Checking if `money>=price`…
- …decision (diamond shape) used
- …`venditem()` and `giveChange(money-price)` if True/Yes
- …output an error if False / No
- Terminator used to start and end the program and all paths terminated

5

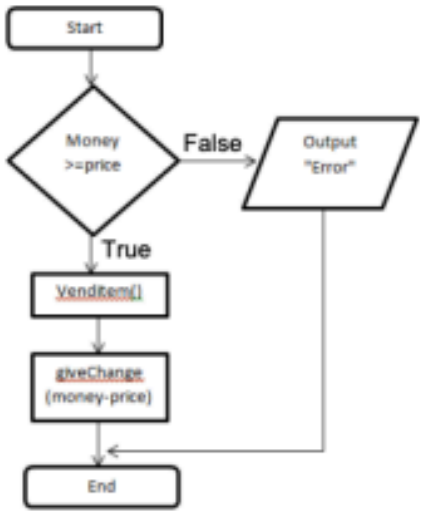
AO3
2b(5)

Reasonable attempt at BP1 needed for credit BP2, 3 and 4

Ignore other additional code.

BP3 and BP4 must follow on from True/False // Yes/No decision to be credited.

Subroutines names and parameters must be correct. Ignore missing brackets on `venditem`.




---

**6** **(a)**

- Access "Rob" / studentnames[0]…
- …does **not** equal "Anna" // not desired item // move on
- Access "Anna" / studentnames[1]
- …does equal "Anna" // stop // item found

4

AO2
1b(4)

Answer must refer to this array, not a generic description of linear search. "Access first item" is NE for BP1 or BP3. Must refer to this scenario.

Max 1 for "Compare '**Anna**' to **each item** in list" if nothing else credited.

---

**6** **(b)**

- Anna inserted before Rob as first two elements…
- …Huw correctly inserted into sorted list…
- …Emma correctly inserted into sorted list …
- …Patrice correctly inserted into sorted list …
- …Iqbal correctly inserted into sorted list and **no further changes** made.

5

AO2
1b(5)

| | | | | | |
|---|---|---|---|---|---|
| **Rob** | Anna | Huw | Emma | Patrice | Iqbal |
| **Anna** | Rob | Huw | Emma | Patrice | Iqbal |
| Anna | **Huw** | Rob | Emma | Patrice | Iqbal |
| Anna | **Emma** | Huw | Rob | Patrice | Iqbal |
| Anna | Emma | Huw | **Patrice** | Rob | Iqbal |
| Anna | Emma | Huw | **Iqbal** | Patrice | Rob |

Sorted list highlighted

GCSECOMPUTERSCIENCETUTOR.COM

GCST

# 2019

| | | | | | |
|---|---|---|---|---|---|
| 2 | (a) | | 1 mark per bullet to max 4, 1 mark per row<br><br>• 10<br>• 6<br>• 6<br>• 2 | 4<br>AO2 1b (4) | Correct Answer Only<br><br>Do not accept "X", "Y", etc. |
| 2 | (b) | | 1 mark per bullet to max 6.<br><br>• Inputs two value (as X and Y)<br>• Compares if X is larger than Y…<br>• …Outputs Y*X only when False<br>• Compares if X is less than 12…<br>• …Outputs X only when True and X > Y<br>• …Outputs Y only when False and X > Y | 6<br>AO3 2b (6) | Question specifically asks for pseudocode.<br><br>Outputs should only be given if they occur with the right condition(s).<br><br>Example algorithm<br><br>`input x`<br>`input y`<br>`if x > y then`<br>`  if x < 12 then`<br>`  print x`<br>`  else`<br>`  print y`<br>`  end if`<br>`else`<br>`  print y*x`<br>`end if`<br><br>Variables do not have to be called x and y.<br><br>Accept equivalent comparisons (e.g. `if X <= Y`)<br><br>Allow FT for outputs from incorrect comparisons where a sensible attempt has been made. |
| 3 | (a) | (i) | 1 mark per bullet to max 1<br><br>• An error that results in **incorrect output / unexpected result**<br>• Contains an error but **still runs / doesn't crash** | 1<br>AO1 1b (1) | Do not accept examples of logic errors. |
| 3 | (a) | (ii) | `if num MOD 2 == 0 then`<br><br>`if num MOD 2 = 0 then` | 1<br>AO3 2b (1) | Important point is that >= is changed to == or =.<br>Accept alternatives that produce the same result (e.g. <=0, <1, !=1, etc.)<br><br>Ignore any casting (e.g. using int() to convert to a number)<br><br>Accept other minor changes to the line as long it logically works.<br><br>Accept versions of MOD from high level languages (e.g. Python : `if num % 2 == 0`) |
| 3 | (b) | (i) | 1 mark per bullet to max 1<br><br>• An error in the grammar of the program // error that breaks the rules of the programming language<br>• Contains an error but will not run / translate / execute | 1<br>AO1 1b (2) | **Do not** accept examples of syntax error (e.g. misspelling) |
| 3 | (b) | (ii) | `print("odd")` | 1<br>AO2 1b (1) | Must include quotes (single or double). Do not penalise spelling mistakes in message.<br>Accept sensible alternatives to "odd"<br>Accept alternatives for print / output as long as spelling is accurate |

| | | | | | |
|---|---|---|---|---|---|
| 4 | (a) | (i) | 1 mark per bullet to max 2<br><br>• Removing / hiding / obscuring **unnecessary** detail<br>• Focusing on the important detail<br>• Simplifies **the problem** // reduces complexity // Easy to solve / understand | 2<br>AO1 1a (1)<br>AO1 1b (1) | Accept answers relating to using fewer computational resources<br><br>Must be the programmer making the decision. |
| 4 | (a) | (ii) | 1 mark per bullet to max 1<br><br>• Suitable example of what can be focused on (e.g. player name, match results, goals scored)<br>• Suitable example of what to remove/hide (anything relevant that is not results/goals scored)<br>• Suitable example of a simplification made | 1<br>AO2 1a (1) | Mark first answer only<br><br>Allow any suitable example of abstraction as long it is relevant to the system.<br><br>Allow either first name or surname to be removed as an example, but do not allow both to be removed. |

| | | | | | |
|---|---|---|---|---|---|
| 6 | (c) | (i) | • Not in order / sorted | 1<br>AO2 1b (1) | Mark first answer only |
| 6 | (c) | (ii) | • Linear (search) | 1<br>AO1 1b (1) | Mark first answer only<br>Allow other valid searching algorithms as long as they work on an unsorted list (e.g front and back search) |
| 6 | (d) | (i) | 1 mark per bullet to max 2<br><br>• Flag / record whether a swap has taken place or not<br>• checked as condition to decide **whether to repeat** | 2<br>AO2 1b (2) | The variable records whether a swap has taken place; it does **not** perform the swap. |
| 6 | (d) | (ii) | 1 mark per bullet to max 2<br><br>• Swaps..<br>• ...values of `queuesize[p]` and `queuesize[p+1]`<br>• ...when `queuesize[p]` is larger than `queuesize[p+1]`<br>• using a temporary variable //doesn't overwrite numbers //explanation of process | 2<br>AO2 1b (2) | Do not accept "sorts numbers"<br><br>"swaps numbers" meets BP1. Explanation of which values in the array are swapped meets BP1 and BP2.<br><br>Do not accept direct word for word repetition from the program (e.g. `temp = queuesize[p]` ), question asks for an explanation.<br><br>Explanation of temporary variable must be logically correct. |

| | | | | | |
|---|---|---|---|---|---|
| 6 | (d) | (iv) | 1 mark per bullet to max 2.<br><br>• Insertion (sort)<br>• Merge (sort) | 2<br>AO1 1a (2) | Accept "insert". Do not penalise spelling.<br><br>Do not accept bubble sort (given in previous questions)<br><br>Do not award searching algorithms<br><br>Allow other valid sorting algorithms.<br>(e.g. quick sort, heap sort, shell sort, selection sort, radix sort, bucket sort, tim sort, comb sort, pigeonhole sort, etc.) |

GCSECOMPUTERSCIENCETUTOR.COM

| | | | | | |
|---|---|---|---|---|---|
| 6 | (e) | | 1 mark per bullet to max 8.<br><br>• Input height<br>• Accepts riders > / >= 140 with suitable message<br>• Rejects riders < / <= 120 with suitable message<br>• Checks if height between 120 and 140…<br>• … If True, input whether accompanied<br>• … Suitable output message for True AND False<br>• Correctly counts number of riders in all cases of being allowed to ride *(do not penalise candidates for counting or not counting accompanying adults)*<br>• Attempt to loop based on 8 riders allowed<br><br>Some checks for rider height may be implicit (e.g. using ELSE after checking other heights). If the answer logically works to produce the correct output, it should be marked as correct.<br><br>Loop will almost certainly be condition controlled (WHILE/DO UNTIL) to gain BP8; count controlled (FOR) loop requires significant manipulation to work successfully. | 8<br>AO3 2b (8) | Answers can be in any suitable format (including pseudocode, flowchart, etc). If flowchart used, accept any sensible shapes.<br><br>Do not penalise for lack of initialisation of variables.<br><br>Loop must repeat until 8 riders allowed, not just loop 8 times.<br><br>Do not credit asking whether accompanied if in the wrong place.<br><br>Condition for BP4 may be `120 < h < 140`<br><br>Example algorithm<br>`riders=0`<br>`while riders <8`<br>`  input height`<br>`  if height >= 140 then`<br>`  output "allowed"`<br>`  riders = riders + 1`<br>`  elif height >=120 then`<br>`  input withadult`<br>`  if withadult == "yes"`<br>`  output "allowed"`<br>`  riders = riders + 1`<br>`  else`<br>`  output "not allowed"`<br>`  end if`<br>`  else`<br>`  output "not allowed"`<br>`  end if`<br>`endwhile` |
| 6 | (c) | (i) | • Not in order / sorted | 1<br>AO2 1b (1) | Mark first answer only |
| 6 | (c) | (ii) | • Linear (search) | 1<br>AO1 1b (1) | Mark first answer only<br>Allow other valid searching algorithms as long as they work on an unsorted list (e.g front and back search) |
| 6 | (d) | (i) | 1 mark per bullet to max 2<br><br>• Flag / record whether a swap has taken place or not<br>• checked as condition to decide **whether to repeat** | 2<br>AO2 1b (2) | The variable records whether a swap has taken place; it does **not** perform the swap. |
| 6 | (d) | (ii) | 1 mark per bullet to max 2<br><br>• Swaps..<br>• …values of `queuesize[p]` and `queuesize[p+1]`<br>• …when `queuesize[p]` is larger than `queuesize[p+1]`<br>• using a temporary variable //doesn't overwrite numbers //explanation of process | 2<br>AO2 1b (2) | Do not accept "sorts numbers"<br><br>"swaps numbers" meets BP1. Explanation of which values in the array are swapped meets BP1 and BP2.<br><br>Do not accept direct word for word repetition from the program (e.g. `temp = queuesize[p]` ) , question asks for an explanation.<br><br>Explanation of temporary variable must be logically correct. |
| 6 | (d) | (iv) | 1 mark per bullet to max 2.<br><br>• Insertion (sort)<br>• Merge (sort) | 2<br>AO1 1a (2) | Accept "insert". Do not penalise spelling.<br><br>Do not accept bubble sort (given in previous questions)<br><br>Do not award searching algorithms<br><br>Allow other valid sorting algorithms.<br>(e.g. quick sort, heap sort, shell sort, selection sort, radix sort, bucket sort, tim sort, comb sort, pigeonhole sort, etc.) |

| | | | | | |
|---|---|---|---|---|---|
| 6 | (e) | | 1 mark per bullet to max 8.<br><br>• Input height<br>• Accepts riders > / >= 140 with suitable message<br>• Rejects riders < / <= 120 with suitable message<br>• Checks if height between 120 and 140…<br>• … If True, input whether accompanied<br>• … Suitable output message for True AND False<br>• Correctly counts number of riders in all cases of being allowed to ride *(do not penalise candidates for counting or not counting accompanying adults)*<br>• Attempt to loop based on 8 riders allowed<br><br>Some checks for rider height may be implicit (e.g. using ELSE after checking other heights). If the answer logically works to produce the correct output, it should be marked as correct.<br><br>Loop will almost certainly be condition controlled (WHILE/DO UNTIL) to gain BP8; count controlled (FOR) loop requires significant manipulation to work successfully. | 8<br>AO3 2b (8) | Answers can be in any suitable format (including pseudocode, flowchart, etc). If flowchart used, accept any sensible shapes.<br><br>Do not penalise for lack of initialisation of variables.<br><br>Loop must repeat until 8 riders allowed, not just loop 8 times.<br><br>Do not credit asking whether accompanied if in the wrong place.<br><br>Condition for BP4 may be `120 < h < 140`<br><br>Example algorithm<br>`riders=0`<br>`while riders <8`<br>`  input height`<br>`  if height >= 140 then`<br>`  output "allowed"`<br>`  riders = riders + 1`<br>`  elif height >=120 then`<br>`  input withadult`<br>`  if withadult == "yes"`<br>`  output "allowed"`<br>`  riders = riders + 1`<br>`  else`<br>`  output "not allowed"`<br>`  end if`<br>`  else`<br>`  output "not allowed"`<br>`  end if`<br>`endwhile` |

GCST

# 2018

| | | | | | |
|---|---|---|---|---|---|
| 2 | (a) | (i) | • 2, 3, 4 | 1 | All three numbers needed in the correct order (with no other numbers) for mark. |
| 2 | (a) | (ii) | • 15 | 1 | Accept 3 x 5 |

| | | | | | |
|---|---|---|---|---|---|
| 4 | (c) | (i) | 1 mark per bullet, max 4.<br><br>• List split into individual elements (may be done over several steps or just as a starting point)<br>• **Merge** individual elements into **sorted** lists of size 2<br>• **Merge** lists of size 2 into **sorted** lists of size 4<br>• **Merge** lists of size 4 into final **sorted** list. | 4 | Candidates **can** describe how the merge sort would work rather than showing output values at each stage.<br><br>Ignore intermediate steps.<br><br>**Do not** give final mark for simply showing the list sorted. Must have the (correct) idea of where it being merged from previous lists.<br><br>Candidates' answers describing / showing other sorting algorithms (e.g. bubble sort, insertion sort) are worth 0 marks.<br><br>  |
| 4 | (c) | (ii) | 1 mark per bullet, max 2.<br><br>• Faster/quicker (to sort)…<br>• …for large lists // for lists that are **more unordered**<br><br>• Has a consistent running time (for a lists of same length)…<br>• …doesn't depend on how ordered original list is | 2 | Accept (correct) reference to big O notation for 2nd mark on either mark point although this is beyond scope of GCSE specification.<br><br>Allow "more efficient" for BOD on first bullet point. |
| 8 | | | 1 mark per bullet, max 6.<br><br>• Initialisation of A, B and C as zero.<br>• Allows input (of anything) from the user<br>• Incrementing A, B and C depending on input<br>• Repeats bullet points 2 and 3<br>• …stopping **only** when "END" is entered<br>• Prints out all 3 individual counts and prints **calculated** total count | 6 | Example algorithm<br><br>```<br>acount = 0<br>bcount= 0<br>ccount= 0<br>vote = ""<br>while vote != "END"<br>    vote = input("enter A, B or C")<br>    if vote == "A" then<br>        acount = acount + 1<br>    elseif vote == "B" then<br>        bcount = bcount + 1<br>    elseif vote == "C" then<br>        ccount = ccount + 1<br>    end if<br>endwhile<br>print acount<br>print bcount<br>print ccount<br>print acount+bcount+ccount<br>``` |

GCSE

# 2017

| | | | | | |
|---|---|---|---|---|---|
| 7 | | | 1 mark per bullet:<br>• Storing a number for the user to guess<br>• Loops 10 times correctly<br>• Inputs the user's guess<br>• If correct, outputs congratulations **and** stops the loop / ends the game (any appropriate method of breaking out of loop)<br>• If the guess is greater than stored number, outputs lower (or similar)<br>• If the guess is lower than stored number, outputs higher (or similar)<br><br>e.g. using while loop<br>`num = 50 //(could be a random number)`<br>`x = 0`<br>`while x < 10`<br>`  input guess`<br>`  if guess == num then`<br>`  output "Congratulations"`<br>`  x = 10`<br>`  elseif guess > num then`<br>`  output "lower"`<br>`  else`<br>`  output "higher"`<br>`  endif`<br>`x = x + 1`<br>`end while`<br><br>e.g. example using for loop<br>`num = 50 //(could be a random number)`<br>`for x = 1 to 10`<br>`  input guess`<br>`  if guess == num then`<br>`  output "Congratulations"`<br>`  end  //(could be break / exit, or x = 10)`<br>`  elseif guess > num then`<br>`  output "lower"`<br>`  else` | 6 | Allow pseudocode, flowchart, or structured English as long as it is not just repeating the instructions and where it meets the bullet points.<br><br>If candidate uses FOR loop, accept 0 to 9 / 0 to 10 / 1 to 10 / 1 to 11 (or equivalent) as valid for 2nd bullet point. |

GCSE

# 2015

| | | | | | |
|---|---|---|---|---|---|
| | b | | • (Age < 20 is FALSE so) Dose = 2<br>• ( Gender = "Female" is FALSE) so Dose = Dose * 0.5<br>• ... therefore Dose = 1 | 3 | **Award mark for first bullet only if 2 clearly refers to the dose.**<br><br>Allow follow through error for second and third bullet. i.e. if candidate has the wrong dose they can still get a mark for Dose * 0.5 and for doing this calculation correctly. (Typically 3 * 0.5 = 1.5 which is therefore worth 2 marks) |
| | c | | • (Age is less than 20 = true) so Dose = 0.1 * Age 1.9<br>• [ isPregnant AND Dose > 1.5 ] is TRUE<br>• Dose = 1.5 | 4 | **Candidates do not need to refer to dose, provided it is clear that they are performing the correct operation.**<br><br>For 3rd bullet it is sufficient if the candidate has shown that both isPregnant and (Dose > 1.5) are TRUE (This may not be at the same point in the answer and they do not need to explicitly state the result of the AND).. |

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

THANK YOU!

GCST

GCSECOMPUTERSCIENCETUTOR.COM