# End of Unit Quiz – Unit 2.1 Algorithms

* 1. How is **decomposition** used when thinking computationally?

|  |
| --- |
|  |

* 1. A theme park uses a 3D computer simulation of a rollercoaster. Riders must wear a virtual reality headset to experience the ride.

Using examples from this scenario, explain what is meant by **abstraction**.

|  |
| --- |
|  |

ai. The array *people* contains the values:

[“Imogen”, “Fletcher”, “Kirstie”, “Zoe”, “Gavin”]

What is the reason why this array could not be searched using a binary search?

|  |
| --- |
|  |

aii. Once the issue identified in part (i) has been resolved, describe the steps that would be taken to search the array for the value “Fletcher” using a binary search.

|  |
| --- |
|  |

bi. The algorithm below uses a different method to search through the array for a name.

Fill in the gaps to complete the algorithm.

|  |  |  |
| --- | --- | --- |
|  | array people[5]  people = [“Imogen”, “Fletcher”, “Kirstie”, “Zoe”, “Gavin”]  found = False  x = 0  searchfor = input(“enter a name to search for : “)  while found = False ………… x <5  if people[x] = searchfor then found = ……………………..  print “found at position” + …………………  x = x + 1  endif  ……………… |  |

bii. What is the name of this searching algorithm?

|  |
| --- |
|  |

c. A user has a database of 100,000 people and needs to search through to find one particular person.

Compare the efficiency of **both** searching algorithms covered in parts (a) and (b) for a data set of this size.

|  |
| --- |
|  |

ai. A programmer has a list of numbers in an array called *scores*, as shown below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **17** | **9** | **4** | **-12** | **3** | **39** |

When setting up a bubble sort algorithm for these numbers, the programmer uses a variable called *swaps* which can either be True or False.

What is the data type of the variable *swaps*?

|  |
| --- |
|  |

aii. How would the programmer use this variable when implementing the bubble sort?

|  |
| --- |
|  |

b. One section of the bubble sort algorithm used by the programmer is shown below:

|  |  |  |
| --- | --- | --- |
|  | if scores[x] > scores [x] + 1 //if scores in wrong order scores[x] = scores[x+1 scores[x+1] = scores[x] // swap numbers over |  |

What is the error that is contained in the code above? Give a corrected version.

|  |
| --- |
|  |

c. How would an insertion sort algorithm arrange the numbers in the *scores* array into order?

|  |
| --- |
|  |

d. What is the name of one **other** sorting algorithm?

|  |
| --- |
|  |

e. What is one advantage and one disadvantage of using a bubble sort?

|  |
| --- |
|  |

1. A school divides students into house groups based on the month that they were born in. Students born in January, February, March or April are put into Needwood house. Students born in May, June, July or August are put into Marchington House. All other students are put into Trent house.

Using pseudocode, write an algorithm that will:

* Ask the user to enter a number (1 to 12) relating to their birth month
* Decide which house they are in and print this out.
* Keep a running total of how many students are in each house.
* Repeat the above for 20 students.
* When 20 students have entered their details, print out how many students are in each house.

|  |
| --- |
|  |

|  |  |  |
| --- | --- | --- |
|  | num = 7  for x = 1 to num  print x \* num  next |  |

Draw a flowchart version of this algorithm.

|  |
| --- |
|  |

1. Complete the following table to describe the use of each of the following flow chart symbols.

|  |  |
| --- | --- |
| **Symbol** | **Explanation of use** |
|  |  |
|  |  |
|  |  |
|  |  |

1. A car dealer uses the following algorithm to determine the price to charge for cars.

|  |  |  |
| --- | --- | --- |
|  | 01 p = input(“purchase price of car” 02 i = input(“number of improvements made”)  03 a = input(“age of car in years”)  04 s = p + (i\*100)  05 if a <= 10 then  06 s = s + s  07 endif  08 print “sale price is “ + s |  |

Work out the output value with the following inputs:

ai. p = 1000, i = 2, a = 12

|  |
| --- |
|  |

aii. p = 5000, i = 3, a = 10

|  |
| --- |
|  |

aiii. p = 8000, i = 0, a = 5

|  |
| --- |
|  |

b. rewrite line 06 so that the + operator is **not** used.

|  |
| --- |
|  |

**Answers**

* 1. How is **decomposition** used when thinking computationally?

|  |
| --- |
| Used to break down a (complex) problem…  …into small parts / component parts  these parts are easier to solve / understand than the larger problem. |

* 1. A theme park uses a 3D computer simulation of a rollercoaster. Riders must wear a virtual reality headset to experience the ride.

Using examples from this scenario, explain what is meant by **abstraction**.

|  |
| --- |
| a representation of a concept / object / thing…  …in this case, the rollercoaster.  Picks out the important / relevant parts / components / ideas / details…  …in this scenario, the track / rider / car / physics / etc.  .  Ignores details which are not important / relevant…  …in this scenario, the queues / weather / smells / etc. |

ai. The array *people* contains the values:

[“Imogen”, “Fletcher”, “Kirstie”, “Zoe”, “Gavin”]

What is the reason why this array could not be searched using a binary search?

|  |
| --- |
| Values are not in (alphabetical) order. |

aii. Once the issue identified in part (i) has been resolved, describe the steps that would be taken to search the array for the value “Fletcher” using a binary search.

|  |
| --- |
| Array when sorted will be [“Fletcher”, “Gavin”, “Imogen”, “Kirstie”, “Zoe”].  Take the middle value and compare it to the item to be searched for…  ….in this case, compare “Fletcher” (search data) to “Imogen”.  Stop / return value / if data found.  Discard everything above this if the search data is smaller / discard everything below this if the search data is larger.  …in this case, “Fletcher” is smaller (alphabetically) than “Imogen” so discard top half of array.  Repeat / recursively call search routine again on remaining values in array. |

bi. The algorithm below uses a different method to search through the array for a name.

Fill in the gaps to complete the algorithm.

|  |  |  |
| --- | --- | --- |
|  | array people[5]  people = [“Imogen”, “Fletcher”, “Kirstie”, “Zoe”, “Gavin”]  found = False  x = 0  searchfor = input(“enter a name to search for : “)  while found = False AND x <5  if people[x] = searchfor then found = True  print “found at position” + x  x = x + 1  endif  Endwhile |  |

bii. What is the name of this searching algorithm?

|  |
| --- |
| Linear (search). |

c. A user has a database of 100,000 people and needs to search through to find one particular person.

Compare the efficiency of **both** searching algorithms covered in parts (a) and (b) for a data set of this size.

|  |
| --- |
| Linear search compares each value in turn.  Worst case scenario, 100,000 comparisons / all values checked.  Binary search splits size of list in half each time / uses divide and conquer.  Worst case scenario, many fewer comparisons (approx. 17 comparisons).  Binary search is more efficient / linear search is less efficient. |

ai. A programmer has a list of numbers in an array called *scores*, as shown below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **17** | **9** | **4** | **-12** | **3** | **39** |

When setting up a bubble sort algorithm for these numbers, the programmer uses a variable called *swaps* which can either be True or False.

What is the data type of the variable *swaps*?

|  |
| --- |
| Boolean. |

aii. How would the programmer use this variable when implementing the bubble sort?

|  |
| --- |
| Initialised to False at the start of the algorithm.  If pairs of numbers are swapped, set to True.  When all numbers have been compared, check again…  … if swaps is True, repeat algorithm / process again  …setting swaps back to False. |

b. One section of the bubble sort algorithm used by the programmer is shown below:

|  |  |  |
| --- | --- | --- |
|  | if scores[x] > scores [x] + 1 //if scores in wrong order scores[x] = scores[x+1 scores[x+1] = scores[x] // swap numbers over |  |

What is the error that is contained in the code above? Give a corrected version.

|  |
| --- |
| Error  numbers are not swapped correctly / scores[x] will be overwritten  Correction (eg..)  temp = scores[x]  scores[x] = scores[x+1]  scores[x+1] = temp |

c. How would an insertion sort algorithm arrange the numbers in the *scores* array into order?

|  |
| --- |
| Take first number (17) as a sorted list by itself.  Look at next number (9) and insert into the correct place…  …method for doing this (either repeatedly swapping until in right place or comparing against number in sorted list and moving).  Continue until last number is processed / repeat for each number in the list. |

d. What is the name of one **other** sorting algorithm?

|  |
| --- |
| Merge sort. |

e. What is one advantage and one disadvantage of using a bubble sort?

|  |
| --- |
| Advantage : easier / simpler / faster to implement (than other sorting algorithms).  Disadvantage : Less efficient / takes longer to run (than other sorting algorithms. |

1. A school divides students into house groups based on the month that they were born in. Students born in January, February, March or April are put into Needwood house. Students born in May, June, July or August are put into Marchington House. All other students are put into Trent house.

Using pseudocode, write an algorithm that will:

* Ask the user to enter a number (1 to 12) relating to their birth month
* Decide which house they are in and print this out.
* Keep a running total of how many students are in each house.
* Repeat the above for 20 students.
* When 20 students have entered their details, print out how many students are in each house.

|  |
| --- |
| Initialising variables at the start for count of students in three houses  Suitable loop that repeats 20 times  Inputting the birth month as a number  Printing out message AND adding 1 to counter if birth month is between 1 and 4  Printing out message AND adding 1 to counter if birth month is between 5 and 8  Printing out message AND adding 1 to counter if birth month is between 9 and 12  Printing out totals for all three counters at the end.  **Example:**  N = 0  M = 0  T = 0  FOR x = 1 to 20  INPUT birthnum  IF birthnum = 1 or birthnum = 2 or birthnum = 3 or birthnum = 4 THEN  print “Needwood house”  N = N + 1  ELIF birthnum = 5 or birthnum = 6 or birthnum = 7 or birthnum = 8 THEN  print “Marchington house”  M = M +1  ELIF birthnum = 9 or birthnum = 10 or birthnum = 11 or birthnum = 12 THEN  print “Trent house”  T = T + 1  ENDIF  print N, M, T  Must be pseucodode, not flowchart (asked in ).  Mark for loop only given if correct code is inside/outside the loop. Accept WHILE / DO loop as long as it repeats 20 times.  Only allow use of ELSE for between 9 and 12 if suitable validation is used to restrict answers to between 1 and 12.  Allow ECF for 4th and 5th bullet points if something is not correct with 3rd bullet point. |

|  |  |  |
| --- | --- | --- |
|  | num = 7  for x = 1 to num  print x \* num  next |  |

Draw a flowchart version of this algorithm.

|  |
| --- |
| Start and end symbols (both present and correct shape)  num initially given value 7  x initially given value 1  outputting the result of x\* num (even if previous steps missed)  Deciding whether to loop again correctly (ie if not yet looped 7 times)  …incrementing x by 1 if appropriate  … looping to the correct position in the algorithm if appropriate  …stopping if at end of loop / already repeated 7 times  correct shapes for flowchart symbols must be used.  Decision box must be labelled up with at least one of YES/NO (allow BOD if only 1 labelled).  Allow different variable names as long as the algorithm would logically work. |

1. Complete the following table to describe the use of each of the following flow chart symbols.

|  |  |
| --- | --- |
| **Symbol** | **Explanation of use** |
|  | Process  Used to work out calculations / assignment / instructions that have no input or output  Suitable example (eg x = x + 7) |
|  | Input / output  Used to get information from /give information to the user  Suitable example (eg print “hello”) |
|  | Decision  Used to make yes/no decisions/choices  Suitable example (eg IF x > 10) |
|  | Terminator / start / stop symbol  Used at the very beginning / very end of the program / algorithm. |

1. A car dealer uses the following algorithm to determine the price to charge for cars.

|  |  |  |
| --- | --- | --- |
|  | 01 p = input(“purchase price of car” 02 i = input(“number of improvements made”)  03 a = input(“age of car in years”)  04 s = p + (i\*100)  05 if a <= 10 then  06 s = s + s  07 endif  08 print “sale price is “ + s |  |

Work out the output value with the following inputs:

ai. p = 1000, i = 2, a = 12

|  |
| --- |
| 1200 |

aii. p = 5000, i = 3, a = 10

|  |
| --- |
| 10600 |

aiii. p = 8000, i = 0, a = 5

|  |
| --- |
| 16000 |

b. rewrite line 06 so that the + operator is **not** used.

|  |
| --- |
| S=s\*2 |

**OCR Resources**: *the small print*OCR’s resources are provided to support the delivery of OCR qualifications, but in no way constitute an endorsed teaching method that is required by the Board, and the decision to use them lies with the individual teacher. Whilst every effort is made to ensure the accuracy of the content, OCR cannot be held responsible for any errors or omissions within these resources.   
© OCR 2017 - This resource may be freely copied and distributed, as long as the OCR logo and this message remain intact and OCR is acknowledged as the originator of this work.

OCR acknowledges the use of the following content: n/a

Please get in touch if you want to discuss the accessibility of resources we offer to support delivery of our qualifications: [resources.feedback@ocr.org.uk](mailto:resources.feedback@ocr.org.uk)

We’d like to know your view on the resources we produce. By clicking on ‘[Like’](mailto:resources.feedback@ocr.org.uk?subject=I%20liked%20the%20GCSE%20Computer%20Science%20Quiz%202.1%20Algorithms) or [‘Dislike’](mailto:resources.feedback@ocr.org.uk?subject=I%20disliked%20the%20GCSE%20Computer%20Science%20Quiz%202.1%20Algorithms) you can help us to ensure that our resources work for you. When the email template pops up please add additional comments if you wish and then just click ‘Send’. Thank you.

Whether you already offer OCR qualifications, are new to OCR, or are considering switching from your current provider/awarding organisation, you can request more information by completing the Expression of Interest form which can be found here: [www.ocr.org.uk/expression-of-interest](http://www.ocr.org.uk/expression-of-interest)

Looking for a resource? There is now a quick and easy search tool to help find free resources for your qualification:   
[www.ocr.org.uk/i-want-to/find-resources/](http://www.ocr.org.uk/i-want-to/find-resources/)

This formative assessment resource has been produced as part of our free GCSE teaching and learning support package. All the GCSE teaching and learning resources, including delivery guides, topic exploration packs, lesson elements and more are available on the qualification webpages.

If you are looking for examination practice materials, you can find Sample Assessment Materials (SAMs) on the qualification webpage: [Computer Science (9-1)](http://www.ocr.org.uk/qualifications/gcse-computer-science-j276-from-2016/)