A Level Computer Science Exam Tips

General Questions

- Know your basic definitions - e.g. the parts of the CPU, object orientation, laws relating to computing - as these come up every time. I have a Quizlet set available for purchase within my revision bundle with all of these definitions, so you can easily study wherever you are.

- Check the number of marks for the question - a 9 mark essay-style question will expect **knowledge, application and evaluation.** Make sure you relate everything you write to the scenario in the question for the maximum marks. Unless you're answering a 1 mark question, you'll usually need an explanation for your answer.

- Read the examiner's reports! These are available alongside past papers, and they basically explain what candidates did well and what they didn't do so well at. If you struggle to understand what they're expecting of you in a question, this tends to give you some indication of what they wanted to see. For example, this point tells you that it's worth drawing diagrams if it gets your point across more easily!

2c	Few candidates could give a clear answer in part (i) using the correct technical vocabulary that the array index/subscript could be used as the node number. Many candidates could work through the logic required in the trace table in part (ii), but fewer could actually explain what it was doing in (iii) within the context of the scenario. Part (iv) was often best answered by those candidates who used the diagram to give the solution. Candidates should be encouraged to use diagrams where they can be used to good effect rather than lengthy or vague prose descriptions.

  Also, if a lot of candidates struggle on a question, they tend to repeat questions on that topic to see if the teaching for that topic is improving, and to make it easier to see who the top performing students are. This isn't guaranteed though, so make sure you revise as many topics as possible!

- When you're revising, refer back to the specification if you need so that you have a checklist of everything you need to cover.

Pseudocode Questions

- Understand the different sorting and searching algorithms - you don't have to memorise these but you may be asked to fill in the blanks with pseudocode. For example, see this question from Paper 2 June 2017:

1 A programmer needs to sort an array of numeric data using an insertion sort.

(a) (i) The following, incomplete, algorithm performs an insertion sort.

Complete the algorithm.

```
procedure sortit(dataArray, lastIndex)
  for x = 1 to lastIndex
    currentData = dataArray[.....X.....]
    position = x
    while (position > 0 AND dataArray[position-1] > currentData)
      dataArray[position]  = dataArray[position - 1]
      position = position - 1
    endwhile

    dataArray[position] = currentData
  next x
endprocedure
```

[3]

The first blank must be X, because the counter in the for loop is X and you want to iterate over the array. The second blank is position - 1, because insertion sort works by moving items from the unsorted part of the list (everything between x and lastIndex) into the correct part of the sorted list. The condition within the while loop checks if the element directly before currentData is greater than currentData - if it is, then it must be in the wrong place - and then moves these elements around. Therefore the third blank must be currentData, because the element that was too big has moved into currentData's spot, so we need to re-insert currentData into the list.

- On a related note, make sure all of the variables that were defined have been used! If the last gap wasn't currentData then there would be no point giving currentData a value. If you haven't used all of the variables and the rest of the code hasn't either, then re-read the question. Consider highlighting/ crossing out in pencil all of the variables (in the above answer you want to highlight x, currentData and position) and also mark the points in the code they were used. This can help you keep track of things.

- If you get confused by reading pseudocode, try writing out in plain English what each line is supposed to do. Using the same example:

1 A programmer needs to sort an array of numeric data using an insertion sort.

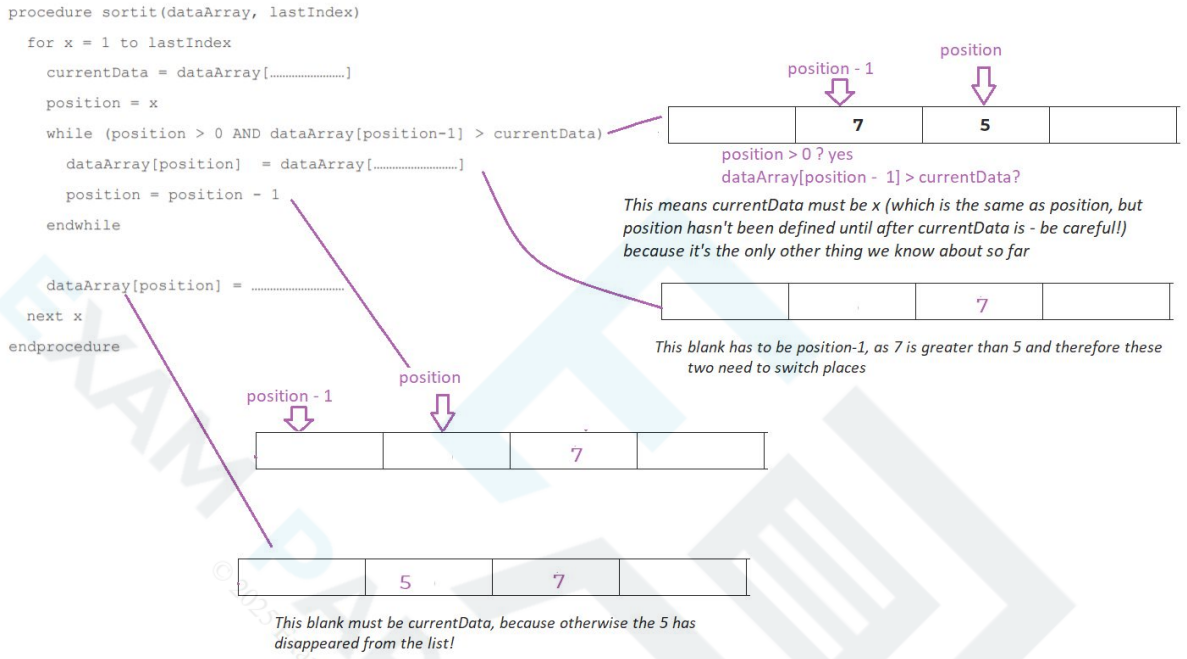
(a) (i) The following, incomplete, algorithm performs an insertion sort.

Complete the algorithm.

```
procedure sortit(dataArray, lastIndex)
  for x = 1 to lastIndex      for loop: going over the whole array. 'For each item in the list...'
    currentData = dataArray[.....]
    position = x
    while (position > 0 AND dataArray[position-1] > currentData)      while: check if a condition is met and repeat if so.
      dataArray[position]  = dataArray[.....]                         'if position isn't the first item in the list and it's bigger than currentData...'
      position = position - 1          move the items in the list around and repeat
    endwhile                           these steps with the item to the left
```

This should lead you to notice what the blanks are more easily.

- Alternatively, draw a quick diagram for any long or confusing steps (like this while loop)



- If you have to write your own pseudocode, make sure your variable names are sensible! SIngle letters like i and x are fine for loop counters but try and make the names descriptive if possible. In the example above, they used position = x to make it more readable - do something similar or you may lose marks. (This is general good programming practice - do it in your coursework as well!)

- Know what's required of you in terms of pseudocode, SQL, HTML/CSS or Little Man Computer. This is all in the specification (page 36 according to the numbers at the bottom is a good starting point) so don't learn more than you need to unless you already know it. An idea is to highlight or cross off any that you confidently know and research the rest, coming back to the list when you've had some practise.

Tips for Sitting the Exam

- Bring a pencil, ruler, eraser and maybe a highlighter to the exam - this will be helpful for diagrams and also making quick notes to yourself if you're trying to format a long-answer question.

- Watch the time - obviously, some questions will take longer for you than others based on their length and how well you understand the topic. If you can't think of the answer then you can always come back to it - it's a long enough exam.

- You don't have to do the questions in order! If you open the first page and you want to cry, try skipping forward and knock out a few of the simpler questions or lower marks.

- If you have a tendency to run out of time, then try aiming for the longer questions first, or your best topics, so you know that you can get the most marks in the allowed time. The five or so minutes you spend scanning the paper for your best questions and what order you should try them in will be much more beneficial than spending 5 minutes on a question not knowing what to write.

- Time yourself doing a past paper so you know if you need to speed up / slow down. This can be hard on this course when there's not very many past papers, but it's incredibly useful. You may also do a past paper in exam conditions in class.

- If you're running out of past papers but still want to revise, try the old specification past papers, and skip any questions you have no idea about or aren't on the new specification. Start with the newest papers first and work your way back as the newest papers are most likely to be similar to your paper.

- If you get really nervous during exams (like me!), I'd also recommend scanning over the whole paper first. This means you don't get any surprises coming your way and you can ease yourself into it with your favourite questions first.

- Another tip for exam nerves is to bring water with you and drink it slowly whilst you think about questions, or between questions to clear your head. Not only does this give you a moment when you're not staring at the paper but it takes your brain out of panic mode by stopping you from rushing. (This also works for interviews - if they offer you a drink then water is a good choice as it takes your mind off it!)

- Don't overload yourself with information - ultimately there will be questions you can do and questions you can't, and no amount of late night cramming will change this. Be prepared, but no matter how the exam goes remember that you've done your best and your best is the best.