# COMPUTING

**Paper 9691/11**
**Written Paper**

## General comments

A significant number of candidates appear to be rote learning certain topics. Questions which required an application of knowledge were overall less well answered. Candidates will need to change how they approach computing topics. This is a challenge which many Centres and candidates, of course, will relish.

Candidates and Centres are reminded that written papers are now scanned in and marked on computer screens by Examiners. Consequently, if a candidate writes the answer to a question on an additional page they must indicate VERY CLEARLY to the Examiner where their revised answer is to be found. If answers are crossed out, the new answers must be very clear so that Examiners can easily read the text and award candidates the appropriate mark.

## Comments on specific questions

### Question 1

**(a) (i)** Most candidates correctly wrote that ROM is non-volatile and RAM is volatile. The better candidates also wrote that data held in ROM can not be altered, whereas in RAM data can be altered.

**(ii)** Most candidates knew that the bootstrap program must be held in ROM so that it is present when the computer is switched on.

**(iii)** Most candidates knew that the bootstrap program runs start-up sequences and loads the operating system.

**(b) (i)** Weaker candidates need to be aware not to just rewrite the question. A correct definition of an input device is a peripheral that allows data to be entered to a computer.

**(ii)** Similarly an output device is a peripheral that allows information to be reported by a computer after processing.

**(c)** Many candidates ignored the stem of the question and gave vague answers such as keyboard. A significant number suggested a monitor as a suitable output device for blind people. A common error was to call voice recognition an example of a hardware input device. For storage, a significant number think USB is a storage device. These are two good examples of candidates not thinking clearly before putting pen to paper. The better candidates suggested a Braille keyboard so the secretary can feel the characters on each key, a Braille printer to produce documents the secretary can read and a USB flash memory stick to take back-ups of files.

### Question 2

**(a)** In part **(i)**, many candidates just re-wrote the question by saying: “it controls the operation of the system”. There needed to be some reference to the computer here to gain the definition mark. Part **(ii)** was not really answered much better; there were some very general answers which could have been descriptions of ANY type of software.

**(b)** For part **(i)** a large number of candidates gained credit for referring to *off-the-shelf software* as being readily available. However, in part **(ii)** many candidates wrongly thought that you do not need a programmer to write the software for *off-the-shelf* packages. The best candidates correctly explained that off-the-shelf software is cheaper to buy because the development costs are shared.

**(c) (i)** Most candidates correctly identified the student could use a word processor to write an essay.

**(ii)** Most candidates correctly identified the student could use a spreadsheet to store rainfall readings and produce graphs.

**(iii)** Most candidates correctly identified the student could use desk-top publishing to produce the final project report.

**(d)** There were some good attempts at answering this question. However, some candidates confused this with time sharing systems. The best answers stated that only one user is allowed access at any one time, that users are allocated disk space to store their files, accessed by passwords, that individual users can have different access rights to files and software and that the system will appear to be running more than one piece of software at a time.

**Question 3**

**(i)** Both types of real time processing (transaction and process control/monitoring) were chosen here by candidates. The weaker candidates struggled to define what it meant. The better candidates defined transaction processing as immediate updating of files. Good answers stated that process control is where the current output affects the next input.

**(ii)** Computer games were often chosen as the example. Unfortunately, there needed to be some indication as to why the game required real time processing before credit could be given. Examples given of transaction processing included airline booking systems.

**(iii)** Most candidates either chose “to avoid double booking” (transaction processing) or an immediate response is needed (process control/monitoring).

**Question 4**

**(i)** Most candidates could give the response that the reader reads the position of a mark on the paper document, for example a multiple choice exam question paper. Only the best candidates could explain that the position the reader collects is then translated into information.

**(ii)** As with part **(i)**, many candidates mentioned reading the shape of the character and for giving a suitable example where OCR would be used. Only the best candidates had the required understanding of how the character shape was recognised (e.g. by having a library of shapes stored).

**Question 5**

This question was very fairly well answered. The majority of candidates mentioned economically possible and technically feasible. However, many candidates just expanded on these two issues; only the best candidates could explain further.

**Question 6**

Many candidates just regurgitated previous mark schemes and referred to colour blind users, use of large fonts and the natural reading pattern of left to right. Such responses miss the point of this question. Very few tried to fit their answer to the scenario which was the control/monitoring of a section of railway track where the need for safety, a fast response and easy navigation would be of paramount importance. Many candidates really need to spend a few minutes to consider the scenario presented to them before they begin to write their answer.

**Question 7**

**(a)** Part **(i)** asked for a 10-bit binary representation. Many candidates gave answers such as 100010110 which is only 9 bits. However, part **(ii)** gave few problems; the main error here was the answer 344 instead of 172 (presumably this was due to candidates numbering the binary bit positions as: 2, 4, 8, … missing out the 1 value in the right-most bit position).

**(b) (i)** The course level was stated as a single character, but several candidates still gave answers such as A Level, AS Level, etc. The start date clearly needed to be in date format. However, candidates still gave answers such as 15 May 2012 or put the date between quotes.

**(ii)** This was generally well answered; although for the *NumberOfPrizesWon*, whilst data type integer was correctly chosen, the most common reason given: "since it needs to be an integer value" is not enough – it is necessary to define an integer as, for example, "the number needs to be a whole number" or "cannot include decimals". Likewise, the data type for A*verageExaminationMark* was often correctly given as floating point or real, but the reason given was often incorrect or too vague. All that was required was some statement to the effect that fractional values need to be stored.

**Question 8**

**(a)** Most candidates could explain benefits of connecting computers in a local area network and define parallel data transmission.

**(b)** There was quite a bit of evidence of rote learning but the concept of circuit switching and packet switching seemed to be reasonably well understood.

**Question 9**

**(a)** Most candidates correctly completed the truth table for a NOR gate.

**(b)** Most candidates correctly completed the truth table for the given logic circuit.

# COMPUTING

**Paper 9691/12**
**Written Paper**

## General comments

A significant number of candidates appear to be rote learning certain topics. Questions which required an application of knowledge were overall less well answered. Candidates will need to change how they approach computing topics. This is a challenge which many Centres and candidates, of course, will relish.

Candidates and Centres are reminded that written papers are now scanned in and marked on computer screens by Examiners. Consequently, if a candidate writes the answer to a question on an additional page they must indicate VERY CLEARLY to the Examiner where their revised answer is to be found. If answers are crossed out, the new answers must be very clear so that Examiners can easily read the text and award candidates the appropriate mark.

## Comments on specific questions

### Question 1

**(a) (i)** Most candidates correctly wrote that ROM is non-volatile and RAM is volatile. The better candidates also wrote that data held in ROM can not be altered, whereas in RAM data can be altered.

**(ii)** Most candidates knew that the bootstrap program must be held in ROM so that it is present when the computer is switched on.

**(iii)** Most candidates knew that the bootstrap program runs start-up sequences and loads the operating system.

**(b) (i)** Weaker candidates need to be aware not to just rewrite the question. A correct definition of an input device is a peripheral that allows data to be entered to a computer.

**(ii)** Similarly an output device is a peripheral that allows information to be reported by a computer after processing.

**(c)** Many candidates ignored the stem of the question and gave vague answers such as keyboard. A significant number suggested a monitor as a suitable output device for blind people. A common error was to call voice recognition an example of a hardware input device. For storage, a significant number think USB is a storage device. These are two good examples of candidates not thinking clearly before putting pen to paper. The better candidates suggested a Braille keyboard so the secretary can feel the characters on each key, a Braille printer to produce documents the secretary can read and a USB flash memory stick to take back-ups of files.

### Question 2

**(a)** In part **(i)**, many candidates just re-wrote the question by saying: “it controls the operation of the system”. There needed to be some reference to the computer here to gain the definition mark. Part **(ii)** was not really answered much better; there were some very general answers which could have been descriptions of ANY type of software.

**(b)** For part **(i)** a large number of candidates gained credit for referring to *off-the-shelf software* as being readily available. However, in part **(ii)** many candidates wrongly thought that you do not need a programmer to write the software for *off-the-shelf* packages. The best candidates correctly explained that off-the-shelf software is cheaper to buy because the development costs are shared.

**(c) (i)** Most candidates correctly identified the student could use a word processor to write an essay.

**(ii)** Most candidates correctly identified the student could use a spreadsheet to store rainfall readings and produce graphs.

**(iii)** Most candidates correctly identified the student could use desk-top publishing to produce the final project report.

**(d)** There were some good attempts at answering this question. However, some candidates confused this with time sharing systems. The best answers stated that only one user is allowed access at any one time, that users are allocated disk space to store their files, accessed by passwords, that individual users can have different access rights to files and software and that the system will appear to be running more than one piece of software at a time.

**Question 3**

**(i)** Both types of real time processing (transaction and process control/monitoring) were chosen here by candidates. The weaker candidates struggled to define what it meant. The better candidates defined transaction processing as immediate updating of files. Good answers stated that process control is where the current output affects the next input.

**(ii)** Computer games were often chosen as the example. Unfortunately, there needed to be some indication as to why the game required real time processing before credit could be given. Examples given of transaction processing included airline booking systems.

**(iii)** Most candidates either chose “to avoid double booking” (transaction processing) or an immediate response is needed (process control/monitoring).

**Question 4**

**(i)** Most candidates could give the response that the reader reads the position of a mark on the paper document, for example a multiple choice exam question paper. Only the best candidates could explain that the position the reader collects is then translated into information.

**(ii)** As with part **(i)**, many candidates mentioned reading the shape of the character and for giving a suitable example where OCR would be used. Only the best candidates had the required understanding of how the character shape was recognised (e.g. by having a library of shapes stored).

**Question 5**

This question was very fairly well answered. The majority of candidates mentioned economically possible and technically feasible. However, many candidates just expanded on these two issues; only the best candidates could explain further.

**Question 6**

Many candidates just regurgitated previous mark schemes and referred to colour blind users, use of large fonts and the natural reading pattern of left to right. Such responses miss the point of this question. Very few tried to fit their answer to the scenario which was the control/monitoring of a section of railway track where the need for safety, a fast response and easy navigation would be of paramount importance. Many candidates really need to spend a few minutes to consider the scenario presented to them before they begin to write their answer.

**Question 7**

**(a)** Part **(i)** asked for a 10-bit binary representation. Many candidates gave answers such as 100010110 which is only 9 bits. However, part **(ii)** gave few problems; the main error here was the answer 344 instead of 172 (presumably this was due to candidates numbering the binary bit positions as: 2, 4, 8, … missing out the 1 value in the right-most bit position).

**(b) (i)** The course level was stated as a single character, but several candidates still gave answers such as A Level, AS Level, etc. The start date clearly needed to be in date format. However, candidates still gave answers such as 15 May 2012 or put the date between quotes.

**(ii)** This was generally well answered; although for the *NumberOfPrizesWon*, whilst data type integer was correctly chosen, the most common reason given: "since it needs to be an integer value" is not enough – it is necessary to define an integer as, for example, "the number needs to be a whole number" or "cannot include decimals". Likewise, the data type for A*verageExaminationMark* was often correctly given as floating point or real, but the reason given was often incorrect or too vague. All that was required was some statement to the effect that fractional values need to be stored.

**Question 8**

**(a)** Most candidates could explain benefits of connecting computers in a local area network and define parallel data transmission.

**(b)** There was quite a bit of evidence of rote learning but the concept of circuit switching and packet switching seemed to be reasonably well understood.

**Question 9**

**(a)** Most candidates correctly completed the truth table for a NOR gate.

**(b)** Most candidates correctly completed the truth table for the given logic circuit.

# COMPUTING

**Paper 9691/13**
**Written Paper**

## General comments

A significant number of candidates appear to be rote learning certain topics. Questions which required an application of knowledge were overall less well answered. Candidates will need to change how they approach computing topics. This is a challenge which many Centres and candidates, of course, will relish.

Candidates and Centres are reminded that written papers are now scanned in and marked on computer screens by Examiners. Consequently, if a candidate writes the answer to a question on an additional page they must indicate VERY CLEARLY to the Examiner where their revised answer is to be found. If answers are crossed out, the new answers must be very clear so that Examiners can easily read the text and award candidates the appropriate mark.

## Comments on specific questions

### Question 1

**(a) (i)** Most candidates correctly wrote that ROM is non-volatile and RAM is volatile. The better candidates also wrote that data held in ROM can not be altered, whereas in RAM data can be altered.

**(ii)** A large number of candidates realised that RAM was used to allow files to be altered. The majority made a reasonable attempt at an answer.

**(iii)** Only the best candidates gave the key part to the answer in this question " …. in current use" . Just stating "for parts of the operating system" was not sufficient.

**(b)** A large number of candidates gained credit for claiming that peripheral storage devices were used so that files could be downloaded at a later date when required. Only the best candidates made reference to backing up or that RAM is generally too small to store all the files.

**(c)** This question was poorly answered with devices such mouse, microphone, electric motors and keyboards all being offered as input and output devices for the robot. Such answers are clearly incorrect in an application of this type. Very few candidates referred to the need for actuators to control the wheel motors or the need for a solid state memory to contain the control program or user's instructions. Most candidates attempted to describe the sensors that would be needed but very few came up with the right type such as touch or pressure sensors.

### Question 2

**(a)** In part **(i)**, many candidates just re-wrote the question by saying: "it controls the operation of the system". There needed to be some reference to the computer here to gain the definition mark. Part **(ii)** was not really answered much better; there were some very general answers which could have been descriptions of ANY type of software.

**(b)** For part **(i)** a large number of candidates gained credit for referring to *off-the-shelf software* as being readily available. However, many candidates wrongly thought that you do not need a programmer to write the software for *off-the-shelf* packages. The best candidates correctly explained that custom-written software is more expensive to buy because the customer has to pay for the whole development cost.

**(c) (i)** Many candidates suggested that drawing packages were used to design new kitchen appliances and cupboards; this ignored the fact that the question was referring to the sales team and not the design team.

**(ii)** This was fairly well answered with many realising that the database would be used to store data about potential new customers.

**(iii)** Most candidates' correct response was that presentation software could be used to produce an audio-visual presentation to attract customers to the mall.

**Question 3**

**(i)** Most candidates responded correctly here.

**(ii)** Reference to payroll systems was the most common response here. Equally as correct would be utility/mobile phone billing systems or the clearing of cheques at a bank.

**(iii)** The majority of candidates were given credit here for indicating that all the calculations were similar or involved large amounts of data and batch processing was therefore very suitable. Very few realised that pay cannot be calculated until all the data about each worker (e.g. number of hours worked in the week/month, number of hours overtime, days off sick, etc.) was known; this meant that real time processing, for example, would not be a suitable method.

**Question 4**

**(i)** The majority of candidates gained credit for reference to barcodes being a series of dark and light lines of varying thickness and also for supplying a good example of their use. Only the best candidates realised that the width of the barcode lines was measured by the laser.

**(ii)** The majority of candidates gained credit for reference to swiping the card through a card reader and for giving an example of the use of a card reader (e.g. paying for goods using a credit/debit card). Only the best candidates knew that such cards have three strips of information available on which a small amount of data can be stored.

**Question 5**

This question was very well answered with some really good descriptions of corrective, adaptive and perfective maintenance.

**Question 6**

Many candidates just regurgitated previous mark schemes and referred to colour blind candidates and the natural reading pattern of left to right. Such responses did not really answer the question. The answer really needed references to the use of bright colours, use of animation and sound and use of large fonts plus the need to encourage the children by completing some drawing or by playing a fanfare every time they got the answer right.

**Question 7**

**(a)** Part **(i)** asked for a 10-bit binary representation. Many gave answers such as 1101 1110 which is only 8 bits. However, part **(ii)** gave few problems; the main error here was the answer 216 instead of 108 (presumably this was due to candidates numbering the binary bit positions as: 2, 4, 8, … missing out the 1 value in the right most bit position).

**(b) (i)** The item size was stated as a single character, but several candidates still gave answers such as XL, XXS, etc. The sell by date clearly needed to be in date format. However, candidates still gave answers such as 15 May 2012 or put the date between quotes.

**(ii)** This was generally well answered; although for the *MinimumStockLevel*, whilst data type integer was correctly chosen, the most common reason given: "since it needs to be an integer value" is not enough – it is necessary to define an integer as, for example, "the number needs to be a whole number or cannot include decimals". Likewise, the data type for *Price* was often correctly given as currency or real, but the reason given was often not very clear e.g. "it needs to be a decimal"; a more detailed answer is required, such as: "currency requires the number to be 2 places of decimal".

**Question 8**

**(a)** Most candidates could explain what is meant by a WAN and define serial data transmission.

**(b)** Part **(i)** was answered reasonably well. In part **(ii)**, many candidates realised that byte 5 was incorrect due to its even parity. Only the very best could explain that because column 4 has an even number of 1s, the 0 in row 5, column 4 needs to be changed to 1.

**Question 9**

**(a)** Most candidates correctly completed the truth table for a NAND gate.

**(b)** Most candidates correctly completed the truth table for the given logic circuit.

# COMPUTING

**Paper 9691/21**
**Written Paper**

**Key message**

Candidates need practical programming experience to do well in this component.

**General Comments**

The candidates who did well in this examination session were those who had clearly learnt how to program in a procedural language. Not only learnt, but had gone through the necessary processes of designing their programs, writing them in a programming language, compiling them on the computer and testing them using various testing techniques.

The paper aims to allow candidates to show their abilities in writing pseudocode, writing programming code and having a good knowledge of the details of their chosen language. They need to be able to use trace tables, flowcharts and all the methods available to test a program.

The best way to learn all this is to have a problem to solve, and to go through all the processes as if it were coursework. The problem will need to involve the simple manipulation of data files, and be complex enough to have to break some of it into procedures. The only way to understand the use of a programming language is by using it in as real a situation as possible.

The best candidates were the ones who had clearly done a reasonable amount of programming, and who had dealt with the stages and problems in writing meaningful code. Those who had not written much code at all struggled with this paper as there was a considerable number of code related questions. This will continue to be the case.

**The questions and responses in detail.**

**Question 1**

**(a)** Most candidates gained most of the marks in this question. The better candidates made appropriate use of a drop-down list for type of books and radio buttons for whether a students reads novels or not. Candidates need to understand that it is important to indicate to a user how to write the date into an input box (for example YYYYMMDD). Another way is to provide drop down lists of a calendar object.

**(b)** Many candidates did not seem to realise what 'justify' meant. They only described what they had done, rather than giving reasons. The better candidates could justify the use of drop-down lists and radio buttons to reduce errors and that these were appropriate for the purpose as well as easy to use.

**(c)** This type of question is now generally well answered by most candidates. Candidates do need to understand that it is important to give a single value for field size and not a range.

**(d)** Most candidates showed that they had some idea of what a nested IF statement was. Common faults were not initialising the totals and ignoring the three steps that had been put in to help. All candidates showed that they could write pseudocode.

**(e)** Most candidates were able to design a printed report showing a title, totals boxes and percentage boxes for each book type.

**(f)** Some candidates gave thorough, detailed answers to this question. They did well, and must have used the file handling statements when programming to be so familiar with them. Some candidates guessed, whilst others did not appear to know what the question meant. They clearly had not handled files in their practical programming.

**Question 2**

**(a)** There was a considerable improvement in the ability of candidates to use array elements correctly when tracing some code.

**(b)** Most candidates correctly stated that the type of error was a logical error. Not so many candidates could correct the line of pseudocode. The majority changed the right line, but used a wrong condition. Candidates need to perform a quick mental check to see if what they had written worked. This should have show that it required the statement

```
WHILE Element <= ArraySize DO
```

**(c)** Many candidates started the array at position 0 in spite of the trace table showing that it started at position 1. Most also got the ending condition wrong, so again they had not checked their work here. A correct solution would be

```
Element ← 1
REPEAT
   INPUT Number[Element]
   Element ← Element + 1
UNTIL Element > ArraySize
```

**(d)** This question was designed to test if the candidate had had to check the logic of his/her pseudocode before coding. It was a situation where the normal ideas of test data did not apply. Stating that the way to do this was to run the program were not appropriate as the question did not refer to a written program.
One method would be to

check starting condition
check state at iteration 499
check state at iteration 500
check state at iteration 501

**Question 3**

**(a)** This question required a good understanding of the string handling facilities in a chosen language. The weaker candidates tried to make the problem simplistic and not general. The better candidates showed that they had practised string handling in a real programming language and were familiar with using the built-in functions listed in the syllabus.

**(b) (i)(ii)** The better candidates showed that the input string names needed to be the parameters for the function and were able to write the function header showing the type of result the function was returning.

**(b) (iii)** Most candidates stated that the function gave a single output, but they did not gain the mark if they then stated that a procedure cannot return a value.
This is completely false in all languages. There are many websites that state that a procedure cannot return a value, often next to an example of one that does. These websites are wrong. Procedures can return values via parameters passed by reference.

**Question 4**

**(a)** The majority of candidates got most parts of this question correct, showing that they understood the difference between the different types of division and that MOD returns the remainder of integer division.

**(b)** The better candidates could then apply this understanding into a practical application and state that Y DIV X will give the number of full boxes, whereas Y MOD X will give the number of melons left over.

**Question 5**

**(a)** The better candidates answered this part correctly. Others worked their way down through the procedure levels but did not realise that they had to work back up through those levels to get the answer.

There are many different ways to represent the working. One example is

Happening (4) becomes Happening (3) + 4
Happening (3) becomes Happening (2) + 3
Happening (2) becomes Happening (1) + 2
Ends at 1
Diagram works back through function calls and finally gives
Happening (4) = 10

**(b) (i)** Most candidates were able to give the line numbers 4 and 6 where the pseudocode shows that the subroutine is a function because the function name is assigned a value.

**(b) (ii)** Most candidates were able to give the line number 6 where the pseudocode shows that the function is recursive because the function is defined in terms of itself.

**(c)** The better candidates correctly stated that the function call `Happening (-1)` would cause an infinite loop and eventually the system would run out of stack space.

**(d)** Most candidates found this part of the question difficult to answer. Candidates need to understand that a recursive solution may be a more elegant solution. However, a recursive solution makes a large number of function calls when the initial value of Num is very large, which could cause stack overflow. This is not a problem for small values of Num. An iterative solution is less likely to cause an error and programmers may find a FOR loop simpler to write and understand.

# COMPUTING

**Paper 9691/22**
**Written Paper**

**Key message**

Candidates need practical programming experience to do well in this component.

**General Comments**

The candidates who did well in this examination session were those who had clearly learnt how to program in a procedural language. Not only learnt, but had gone through the necessary processes of designing their programs, writing them in a programming language, compiling them on the computer and testing them using various testing techniques.

The paper aims to allow candidates to show their abilities in writing pseudocode, writing programming code and having a good knowledge of the details of their chosen language. They need to be able to use trace tables, flowcharts and all the methods available to test a program.

The best way to learn all this is to have a problem to solve, and to go through all the processes as if it were coursework. The problem will need to involve the simple manipulation of data files, and be complex enough to have to break some of it into procedures. The only way to understand the use of a programming language is by using it in as real a situation as possible.

The best candidates were the ones who had clearly done a reasonable amount of programming, and who had dealt with the stages and problems in writing meaningful code. Those who had not written much code at all struggled with this paper as there was a considerable number of code related questions. This will continue to be the case.

**The questions and responses in detail.**

**Question 1**

**(a)** Most candidates gained most of the marks in this question. The better candidates made appropriate use of a drop-down list for type of books and radio buttons for whether a students reads novels or not. Candidates need to understand that it is important to indicate to a user how to write the date into an input box (for example YYYYMMDD). Another way is to provide drop down lists of a calendar object.

**(b)** Many candidates did not seem to realise what 'justify' meant. They only described what they had done, rather than giving reasons. The better candidates could justify the use of drop-down lists and radio buttons to reduce errors and that these were appropriate for the purpose as well as easy to use.

**(c)** This type of question is now generally well answered by most candidates. Candidates do need to understand that it is important to give a single value for field size and not a range.

**(d)** Most candidates showed that they had some idea of what a nested IF statement was. Common faults were not initialising the totals and ignoring the three steps that had been put in to help. All candidates showed that they could write pseudocode.

**(e)** Most candidates were able to design a printed report showing a title, totals boxes and percentage boxes for each book type.

**(f)** Some candidates gave thorough, detailed answers to this question. They did well, and must have used the file handling statements when programming to be so familiar with them. Some candidates

guessed, whilst others did not appear to know what the question meant. They clearly had not handled files in their practical programming.

**Question 2**

**(a)** There was a considerable improvement in the ability of candidates to use array elements correctly when tracing some code.

**(b)** Most candidates correctly stated that the type of error was a logical error. Not so many candidates could correct the line of pseudocode. The majority changed the right line, but used a wrong condition. Candidates need to perform a quick mental check to see if what they had written worked. This should have show that it required the statement

```
WHILE Element <= ArraySize DO
```

**(c)** Many candidates started the array at position 0 in spite of the trace table showing that it started at position 1. Most also got the ending condition wrong, so again they had not checked their work here. A correct solution would be

```
Element ← 1
REPEAT
   INPUT Number[Element]
   Element ← Element + 1
UNTIL Element > ArraySize
```

**(d)** This question was designed to test if the candidate had had to check the logic of his/her pseudocode before coding. It was a situation where the normal ideas of test data did not apply. Stating that the way to do this was to run the program were not appropriate as the question did not refer to a written program.
One method would be to

check starting condition
check state at iteration 499
check state at iteration 500
check state at iteration 501

**Question 3**

**(a)** This question required a good understanding of the string handling facilities in a chosen language. The weaker candidates tried to make the problem simplistic and not general. The better candidates showed that they had practised string handling in a real programming language and were familiar with using the built-in functions listed in the syllabus.

**(b) (i)(ii)** The better candidates showed that the input string names needed to be the parameters for the function and were able to write the function header showing the type of result the function was returning.

**(b) (iii)** Most candidates stated that the function gave a single output, but they did not gain the mark if they then stated that a procedure cannot return a value.
This is completely false in all languages. There are many websites that state that a procedure cannot return a value, often next to an example of one that does. These websites are wrong. Procedures can return values via parameters passed by reference.

**Question 4**

**(a)** The majority of candidates got most parts of this question correct, showing that they understood the difference between the different types of division and that MOD returns the remainder of integer division.

**(b)** The better candidates could then apply this understanding into a practical application and state that Y DIV X will give the number of full boxes, whereas Y MOD X will give the number of melons left over.

**Question 5**

**(a)** The better candidates answered this part correctly. Others worked their way down through the procedure levels but did not realise that they had to work back up through those levels to get the answer.
There are many different ways to represent the working. One example is

Happening (4) becomes Happening (3) + 4
Happening (3) becomes Happening (2) + 3
Happening (2) becomes Happening (1) + 2
Ends at 1
Diagram works back through function calls and finally gives
Happening (4) = 10

**(b) (i)** Most candidates were able to give the line numbers 4 and 6 where the pseudocode shows that the subroutine is a function because the function name is assigned a value.

**(b) (ii)** Most candidates were able to give the line number 6 where the pseudocode shows that the function is recursive because the function is defined in terms of itself.

**(c)** The better candidates correctly stated that the function call `Happening (-1)` would cause an infinite loop and eventually the system would run out of stack space.

**(d)** Most candidates found this part of the question difficult to answer. Candidates need to understand that a recursive solution may be a more elegant solution. However, a recursive solution makes a large number of function calls when the initial value of Num is very large, which could cause stack overflow. This is not a problem for small values of Num. An iterative solution is less likely to cause an error and programmers may find a FOR loop simpler to write and understand.

# COMPUTING

**Paper 9691/23**
**Written Paper**

**Key message**

Candidates need practical programming experience to do well in this component.

**General Comments**

The candidates who did well in this examination session were those who had clearly learnt how to program in a procedural language. Not only learnt, but had gone through the necessary processes of designing their programs, writing them in a programming language, compiling them on the computer and testing them using various testing techniques.

The paper aims to allow candidates to show their abilities in writing pseudocode, writing programming code and having a good knowledge of the details of their chosen language. They need to be able to use trace tables, flowcharts and all the methods available to test a program.

The best way to learn all this is to have a problem to solve, and to go through all the processes as if it were coursework. The problem will need to involve the simple manipulation of data files, and be complex enough to have to break some of it into procedures. The only way to understand the use of a programming language is by using it in as real a situation as possible.

The best candidates were the ones who had clearly done a reasonable amount of programming, and who had dealt with the stages and problems in writing meaningful code. Those who had not written much code at all struggled with this paper as there was a considerable number of code related questions. This will continue to be the case.

**The questions and responses in detail.**

**Question 1**

**(a)** Most candidates gained most of the marks in this question. The better candidates made appropriate use of a drop-down list for age and radio buttons for whether a students is a member of a sports club or not. Candidates need to understand that it is important to indicate to a user what to write into text boxes and therefore these text boxes need to be clearly labelled.

**(b)** Many candidates did not seem to realise what 'justify' meant. They only described what they had done, rather than giving reasons. The better candidates could justify the use of drop-down lists and radio buttons to reduce errors and that these were appropriate for the purpose as well as easy to use.

**(c)** Most candidates had a good try at this question. Male colour blindness is not considered a visual impairment.

**(d)** This type of question is now generally well answered by most candidates. Candidates do need to understand that it is important to give a single value for field size and not a range.

**(e)** Most candidates showed that they had some idea of what a CASE statement was. Common faults were not initialising the totals, ignoring the three steps that had been put in to help and using IF statements within the CASE statement. All showed that they could write pseudocode.

**(f)** Some candidates gave thorough, detailed answers to this question. They did well, and must have used the file handling statements when programming to be so familiar with them. Some candidates guessed, whist others did not appear to know what the question meant. They clearly had not handled files in their practical programming.

**Question 2**

**(a)** This question required a good understanding of the string handling facilities in a chosen language. The weaker candidates tried to make the problem simplistic and not general. The better candidates showed that they had practised string handling in a real programming language and were familiar with using the built-in functions listed in the syllabus.

**(b) (i)(ii)** The better candidates showed that the input string name needed to be the parameter for the function and were able to write the function header showing the type of result the function was returning.

**(b) (iii)** Most candidates stated that the function gave a single output, but they did not gain the mark if they then stated that a procedure cannot return a value.
This is completely false in all languages. There are many websites that state that a procedure cannot return a value, often next to an example of one that does. These websites are wrong. Procedures can return values via parameters passed by reference.

**Question 3**

**(a)** A majority of candidates performed this trace correctly. Others did not appear to be able to follow the nested FOR statements.

**(b)** Most candidates correctly stated that the type of error was a logical error.

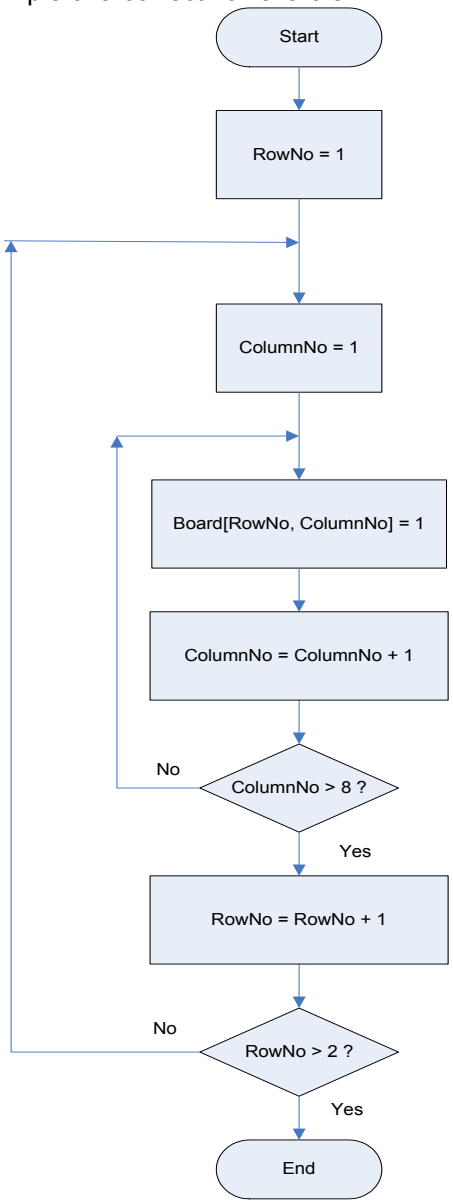
**(c)** The majority changed the right lines, but used a wrong condition. Candidates need to perform a quick mental check to see if what they had written worked. This should have show that it required the statements

**Line 2** `WHILE RowNo <= 4`
**Line 5** `WHILE ColumnNo <= 4`

**(d)** Most candidates correctly stated that a FOR loop would have been better.

**(e)** This was asked to see if candidates could relate array positions to the reality of the problem. A surprising number appeared not to understand that these squares are at opposite corners of the board.

**(f)** Responses to this question varied. Some were able to draw an immaculate flowchart, not an easy task in an examination; others got in a muddle with the layout, or did not notice that the row values were only 1 and 2. An example of a correct flowchart is:



**Question 4**

**(a)** The majority of candidates got most parts of this question correct, showing that they understood the difference between the different types of division and that MOD returns the remainder of integer division.

**(b)** The better candidates could then apply this understanding into a practical application and state that Y DIV X will give the number of full drums, whereas Y MOD X will give the number of litres of oil left over.

**(c)** Most candidates correctly stated that DIV gives the integer part of the result whereas / gives the full result (integer and fractional part).

**Question 5**

**(a)** The better candidates answered this part correctly. Others worked their way down through the procedure levels but did not realise that they had to work back up through those levels to get the answer.

There are many different ways to represent the working. One example is

Something(4) becomes Something(5) + 4
Something(5) becomes Something(6) + 5
Something(6) becomes Something(7) + 6
Ends at 7
Diagram works back through function calls and finally gives
Something(4) = 16

**(b)** The better candidates correctly stated that the function call `Something(8)` would cause an infinite loop and eventually the system would run out of stack space.

**(c)** The better candidates correctly stated that the recursive calls would never end as there would be no stopping condition.

**(d)** There was a little twist to this question in that the answer needed adjusting after using a FOR loop. About a fifth of the candidates picked this up and answered correctly, showing that they understood how both the recursion and the FOR loop worked. Well done.

# COMPUTING

**Paper 9691/31**
**Written Paper**

**General**

The paper presented a challenge to some candidates who did not perform well on questions which moved away from those which required little more than knowledge recall to questions which required the candidate to apply their knowledge to some given scenario i.e. assessing their understanding of how to apply this knowledge. Examples included **Question 5** where the candidate was asked to demonstrate their understanding of two of the modes of addressing in a practical scenario, and then carry this through to part **(c)** of the question to make use of the given subset of instructions. Candidates should be re-assured that they will always be given the instruction set to be used.

Similarly in **Question 3(a)** where candidates were asked to complete the given pseudocode. It was pleasing to see the high standard of responses to this question.

Candidates need to be very cautious when reproducing answers which are taken from previously published mark schemes. See the comments which follow for **Questions 2 (d)(ii)** and **Question 7 (c)** and **(d)**.

**Question 1**

Part **(a)** was poorly answered with very few candidates aware that the issue here was First Normal Form and then unable to recognise that the group of attributes Subject, Level and SubjectTeacher were being repeated for each student.

Part **(b)** was answered well but candidates were then unable to see how the actual data they had in the Student table would determine the primary key of table StudentSubjectChoices. By studying the entries candidates should have been able to deduce that the primary key was a composite key of StudentName + Subject which was the answer for **(c)(i)**.

For **(d)** candidates were unable to express why the table was not in 2NF. What was expected was that they recognised that the SubjectTeacher is only dependant on knowing part of the primary key (i.e. Subject). Better candidates used the correct terminology of partial dependency.

A similar picture for **(d)** with few candidates recognising that the issue was a dependency between non-key attributes i.e. that Tutor would be known from the TutorGroup (or vice versa).

**Question 2**

Generally most candidates scored well on **(a)**, **(b)** and **(c)**.

Part **(d)** proved to be a good discriminator question. Most candidates correctly computed the exponent to be +4 but there was often a completely wrong interpretation of the mantissa part with many candidates not appreciating that this is a fraction and giving a wrong answer of 104. The final mark was gained by the candidate either shifting the binary point the correct number of four places in their mantissa or by putting together the mantissa of +13/16 and the exponent +4.

Part **(i)** was well answered but some candidates stated how the normalised pattern would be recognised (i.e. first two bits of the mantissa would be different) which was not the question asked.

The better candidates correctly stated the mantissa and exponent pattern and computed this to 1/512. Worryingly, it was not uncommon to see answers using base 10.

Part **(e)** was no more than knowledge recall which candidates answered well.

**Question 3**

Part **(a)** - A different style of question on which candidates scored well.

Part **(b)** – The suspicion was that candidates thought this was a straightforward question but they were often let down by poorly expressed answers. *"the search will have to look at all the numbers"* was a frequent incorrect statement. Simple correct statements would have secured the marks such as "The search will have to make a large number of comparisons if the required item is near the end of the list" or "the search will have considered all the items if the item is not found".

Most candidates understood what a binary search involved and gained at least one mark by stating that the list must be in order.

**Question 4**

With the exception of the final part **(d)(iii)** the question was well answered by the majority of candidates.

The intention in providing the candidate with the table to complete for **(iii)** was to be helpful to the candidate in establishing the sequence of changes in the contents of the stack. A common error was to place the operators on the stack and candidates who did this generally did not score any marks. Another omission was not to evaluate the stack contents when an operator is met. E.g. when the stack contained 3 and 7 and the operator 'multiply' was the next term in the expression, the processor will evaluate this and place 21 as the only stack entry. Many candidates showed this as 3 * 7 and not the computed 21 as the stack contents.

**Question 5**

Parts **(a)** and **(b)** asked the candidate to annotate the diagram to illustrate each mode of addressing. All candidates were able for direct addressing to show the correct pattern (or binary number) copied to the Accumulator, but then did not show that it had been copied from address 105. The best answers circled the contents of 105 and then showed an arrow from this memory cell to the Accumulator.

Part **(b)** – Indexed addressing – was far less well understood. The annotation expected was to show 101 having 3 added to it (the contents of the Index Register). Then there is effectively direct addressing from the calculated address of 104. A significant number of candidates are clearly confusing indexed addressing with direct addressing (and wrongly) showed the contents of 101 being used as a forwarding address.

For part **(c)** this was the first time a trace of a simple assembly language program had been asked for, but a significant number of candidates found no difficulty with this and scored the full five marks. Some candidates had the mistaken understanding that following the copy of a value from the Accumulator to memory the Accumulator contents are set to zero.

**Question 6**

Parts **(a)** and **(b)** required little more than knowledge recall and candidates generally scored high marks by describing the ready, blocked and running state of any process.

Part **(b)(ii)** saw very few correct answers. The concept of each process needing a process control block was not seen. Some candidates gained the mark by describing a queue data structure for the ready and blocked processes; some candidates however wrongly stated that there would be a queue structure for all three states.

Part **(c)** was poorly answered despite the first and third mark requiring only a statement that the process would require considerable/very little use of the processor. Few candidates focused on the word 'application' in the stem of the question and as a consequence answers were vague such as *"performs lots of calculations"* for the second mark and did not score.

**Question 7**

Candidates were usually unable to gain the two marks. What was expected were the key concepts that the simulation provides a model of some real-world scenario and that the purpose of doing this is to make predictions about the way in which the simulated system will behave.

Candidates often made the point about this being a read-world problem with an example and this was acceptable.

Part **(b)** then asked for the reasons/benefits of producing a computer simulation and candidates were able to score here with a large number of varied points.

Parts **(c)** and **(d)** were generally well answered with candidates understanding the key issue for **(d)** that what was wanted was a 'cause and effect' which could be modelled.

There were a number of candidates who were quoting issues which were not relevant to the given scenario, such as the number of emergency exits and others, which were clearly lifted from the mark scheme for a previous paper. The message must be to make it clear to candidates that previous mark schemes are of limited use as a tool for revision.

# COMPUTING

**Paper 9691/32**
**Written Paper**

**General**

The paper presented a challenge to some candidates who did not perform well on questions which moved away from those which required little more than knowledge recall to questions which required the candidate to apply their knowledge to some given scenario i.e. assessing their understanding of how to apply this knowledge. Examples included **Question 5** where the candidate was asked to demonstrate their understanding of two of the modes of addressing in a practical scenario, and then carry this through to part **(c)** of the question to make use of the given subset of instructions. Candidates should be re-assured that they will always be given the instruction set to be used.

Similarly in **Question 3(a)** where candidates were asked to complete the given pseudocode. It was pleasing to see the high standard of responses to this question.

Candidates need to be very cautious when reproducing answers which are taken from previously published mark schemes. See the comments which follow for **Questions 2 (d)(ii)** and **Question 7 (c)** and **(d)**.

**Question 1**

Part **(a)** was poorly answered with very few candidates aware that the issue here was First Normal Form and then unable to recognise that the group of attributes Subject, Level and SubjectTeacher were being repeated for each student.

Part **(b)** was answered well but candidates were then unable to see how the actual data they had in the Student table would determine the primary key of table StudentSubjectChoices. By studying the entries candidates should have been able to deduce that the primary key was a composite key of StudentName + Subject which was the answer for **(c)(i)**.

For **(d)** candidates were unable to express why the table was not in 2NF. What was expected was that they recognised that the SubjectTeacher is only dependant on knowing part of the primary key (i.e. Subject). Better candidates used the correct terminology of partial dependency.

A similar picture for **(d)** with few candidates recognising that the issue was a dependency between non-key attributes i.e. that Tutor would be known from the TutorGroup (or vice versa).

**Question 2**

Generally most candidates scored well on **(a)**, **(b)** and **(c)**.

Part **(d)** proved to be a good discriminator question. Most candidates correctly computed the exponent to be +4 but there was often a completely wrong interpretation of the mantissa part with many candidates not appreciating that this is a fraction and giving a wrong answer of 104. The final mark was gained by the candidate either shifting the binary point the correct number of four places in their mantissa or by putting together the mantissa of +13/16 and the exponent +4.

Part **(i)** was well answered but some candidates stated how the normalised pattern would be recognised (i.e. first two bits of the mantissa would be different) which was not the question asked.

The better candidates correctly stated the mantissa and exponent pattern and computed this to 1/512. Worryingly, it was not uncommon to see answers using base 10.

Part **(e)** was no more than knowledge recall which candidates answered well.

**Question 3**

Part **(a)** - A different style of question on which candidates scored well.

Part **(b)** – The suspicion was that candidates thought this was a straightforward question but they were often let down by poorly expressed answers. *"the search will have to look at all the numbers"* was a frequent incorrect statement. Simple correct statements would have secured the marks such as "The search will have to make a large number of comparisons if the required item is near the end of the list" or "the search will have considered all the items if the item is not found".

Most candidates understood what a binary search involved and gained at least one mark by stating that the list must be in order.

**Question 4**

With the exception of the final part **(d)(iii)** the question was well answered by the majority of candidates.

The intention in providing the candidate with the table to complete for **(iii)** was to be helpful to the candidate in establishing the sequence of changes in the contents of the stack. A common error was to place the operators on the stack and candidates who did this generally did not score any marks. Another omission was not to evaluate the stack contents when an operator is met. E.g. when the stack contained 3 and 7 and the operator 'multiply' was the next term in the expression, the processor will evaluate this and place 21 as the only stack entry. Many candidates showed this as 3 * 7 and not the computed 21 as the stack contents.

**Question 5**

Parts **(a)** and **(b)** asked the candidate to annotate the diagram to illustrate each mode of addressing. All candidates were able for direct addressing to show the correct pattern (or binary number) copied to the Accumulator, but then did not show that it had been copied from address 105. The best answers circled the contents of 105 and then showed an arrow from this memory cell to the Accumulator.

Part **(b)** – Indexed addressing – was far less well understood. The annotation expected was to show 101 having 3 added to it (the contents of the Index Register). Then there is effectively direct addressing from the calculated address of 104. A significant number of candidates are clearly confusing indexed addressing with direct addressing (and wrongly) showed the contents of 101 being used as a forwarding address.

For part **(c)** this was the first time a trace of a simple assembly language program had been asked for, but a significant number of candidates found no difficulty with this and scored the full five marks. Some candidates had the mistaken understanding that following the copy of a value from the Accumulator to memory the Accumulator contents are set to zero.

**Question 6**

Parts **(a)** and **(b)** required little more than knowledge recall and candidates generally scored high marks by describing the ready, blocked and running state of any process.

Part **(b)(ii)** saw very few correct answers. The concept of each process needing a process control block was not seen. Some candidates gained the mark by describing a queue data structure for the ready and blocked processes; some candidates however wrongly stated that there would be a queue structure for all three states.

Part **(c)** was poorly answered despite the first and third mark requiring only a statement that the process would require considerable/very little use of the processor. Few candidates focused on the word 'application' in the stem of the question and as a consequence answers were vague such as *"performs lots of calculations"* for the second mark and did not score.

**Question 7**

Candidates were usually unable to gain the two marks. What was expected were the key concepts that the simulation provides a model of some real-world scenario and that the purpose of doing this is to make predictions about the way in which the simulated system will behave.

Candidates often made the point about this being a read-world problem with an example and this was acceptable.

Part **(b)** then asked for the reasons/benefits of producing a computer simulation and candidates were able to score here with a large number of varied points.

Parts **(c)** and **(d)** were generally well answered with candidates understanding the key issue for **(d)** that what was wanted was a 'cause and effect' which could be modelled.

There were a number of candidates who were quoting issues which were not relevant to the given scenario, such as the number of emergency exits and others, which were clearly lifted from the mark scheme for a previous paper. The message must be to make it clear to candidates that previous mark schemes are of limited use as a tool for revision.

# COMPUTING

**Paper 9691/33**
**Written Paper**

**General**

The paper presented a challenge to some candidates who did not perform well on questions which moved away from those which required little more than knowledge recall to questions which required the candidate to apply their knowledge to some given scenario i.e. assessing their understanding of how to apply this knowledge. Examples include **Question 3** where candidates could not complete the given algorithm or design a simple algorithm and write it using pseudocode. Similarly in **Question 6**, candidates were no longer simply asked to *"Describe direct addressing"* and *"Describe indirect addressing"* but were asked to apply this knowledge using the assembly language instructions given to the two diagrams. Candidates should be reassured that they will always be given the instruction set which has to be applied to some practical problem such as the diagrams in 6 **(a)** and **(b)** and the trace of a simple assembly language program for part **(c)**.

**Question 1**

Few candidates showed understanding why the given dataset was not in First Normal form and this carried through into part **(e)** where Third Normal Form appeared not to be understood. A simple statement was required explaining that there is no dependency between the Country and NoOfRunways attributes, or more generally that the NoOfRunways would not be known from the Country (or vice versa).

Answers to part **(f)** were poor, but better answers described data which was updated in one table but not similarly updated in a second table.

**Question 2**

Most candidates scored well on part **(a)** but answers for **(b)** were weak. There were few candidates who were able to explain that all bits have a place value, or that all addition and subtraction calculations give the correct result (as long as the answer is within the possible range).

The candidates' ability to interpret a pattern of bits representing a floating-point number remains weak. Many candidates successfully computed the exponent to be +7 but then wrongly expressed the mantissa as an integer. More worrying were answers which showed a base of 10 used when the mantissa and exponent were combined**.**

**Question 3**

Many candidates were able to secure the four marks for the completion of the diagram but answers to part **(b)** were especially weak. Very few candidates realised that the test for an empty list was HeadPointer = 0 or that the pointer value for any node was referenced with Pointer[Current].

Candidates need to practice writing algorithms like this as there are similar ones which would be used for the various operations for other data structures in the syllabus i.e. a stack, queue and binary tree.

Somewhat illogically, weak answers for part **(b)** were often followed by an answer for **(c)** which demonstrated the key points of the algorithm needed for this requirement and so scored three or four of the available marks.

For the final part **(d)** few candidates were able to describe that the simple change was that the pointer for BERLIN needs to be changed to the current pointer for CANBERRA or better that

Pointer[Previous] ← Pointer[Current].

Better candidates then went on to describe that the space released by CANBERRA would need to be returned to the pool of 'free space' array locations.

**Question 4**

With the exception of the final part **(d)(iii)** the question was well answered by the majority of candidates. Candidates were comfortable converting infix expressions to reverse Polish and the reverse. Most candidates gave as their benefit of the use of reverse Polish that there was no longer the need for brackets to indicate the order of the computation.

The intention in providing the candidate with the table to complete for **(iii)** was to be helpful to the candidate in establishing the sequence of changes in the contents of the stack. A common error was to place the operators on the stack and candidates who did this generally did not score any marks. Another omission was not to evaluate the stack contents when an operator is met. E.g. when the stack contained 7 and 4 and the operator 'multiply' was the next term in the expression, the processor will evaluate this and place 28 as the only stack entry. Many candidates showed this as 4* 7 and not the computed 28 as the stack contents.

**Question 5**

Two key points were looked for in part **(a)**. The concepts that, what is being investigated is a real-world problem (and some candidates did this with an example which was acceptable) and that this is done by building a model of the problem using software. The simulation will then be able to make predictions about the behaviour of the system.

Candidates scored well for part **(b)** but some gave their reasons for **(b)** as the definition asked for in **(a)**.

Most candidates appreciated that the parameters which could be changed were factors such as the number of cars using each road, their speed, the timing of the light change sequence. More resourceful candidates suggested issues such as the time of the day (e.g. busier early morning?) and weather conditions.

For part **(d)** what was asked for was a simple 'cause and effect' such as an increase in the cars on the original road (maybe at certain times of the day?) would require the lights to stay for a longer time on green. Candidates gained no credit for **(d)** by interpreting the concept of 'cause and effect' without reference to the parameters of the simulation e.g. *"the lights changing to green would cause the traffic to start flowing in this direction"***.**

**Question 6**

Most candidates showed the correct pattern/number in the Accumulator for **(a)** in either binary or denary. The better answers then circled the contents of address 66 and showed this being copied to the Accumulator with an arrow. An explanation written on the diagram such as *"this is using direct addressing"* was considered insufficient for the second mark, as this is simply a repeat of the wording in the second column of the given table of instructions.

Part **(c)** used a new question style and many candidates scored the maximum four marks**.**

**Question 7**

Candidates were able to give a definition of an interrupt but then the sequence of steps given for **(b)** lacked many of the key points about interrupt handling. Very few answers were seen which described any of the points which follow which would be a requirement of any interrupt system; the need to save the contents of the Program Counter and other registers, that a stack would be used to store these values, that they would need to be retrieved at some point to resume the interrupted process or, the most obvious point, that for every different interrupt received the operating system must identify the source of the interrupt and then run the Interrupt Service Routine (ISR) program to service the interrupt.

Part **(c)** required little more than knowledge recall and candidates often scored five or the full six marks. The more popular answers described paging and some form of segmentation.

Part **(d)** was poorly answered because candidates did not realise that the high level scheduler is making this decision before the program is loaded into memory, and candidate answers often described scheduling algorithms such as "shortest job next" which were not relevant here. Answers expected included a user priority, an indication of the anticipated run time or the peripheral devices the program will require.

# COMPUTING

**Paper 9691/04**
**Project**

## General comments

This report provides general feedback on the overall quality of project work for A Level Computing candidates. In addition, all Centres receive specific feedback from their Moderator in the form of a short report that is returned after moderation. This reporting provides an ongoing dialogue with Centres giving valuable pointers to the perceived strengths and weaknesses of the projects moderated.

The projects submitted covered a wide variety of topics with better candidates again showing evidence of researching a problem beyond their School or college life.

In order to have the full range of marks available to the candidate, the computing project must involve a third party client whose requirements are considered and clearly documented at all stages of the system development. Centres are reminded that the project work is designed to test the candidates' understanding of the systems life cycle not just the implementation of a solution. 'The Guidance on selecting the Computing Project' ***section 7.1*** provides an overview of what is required. The requirements are clearly set out in syllabus ***section 4***, 'The Guidance on Marking the Computing Project' ***section 7.2*** acts as a useful checklist, for teachers and candidates, setting out the expected contents of each section.

Centres are reminded that this guidance and the marking scheme have changed in 2011. Please use the up-to-date A Level Computing Syllabus for guidance on project choice, content required and how to assess candidates' project work.

Centres are also reminded that candidates should use this guidance for the expected contents of their reports rather than some of the popular A Level textbooks available for project work, which do not cover the full requirements of the CIE syllabus. Candidates who prepare their work only using these text books and not the syllabus for guidance may miss out vital sections of their reports; or complete unnecessary work e.g. feasibility studies and costings. Also some candidates are including unnecessary work that relates to the old syllabus; the sub-sections on **Appropriateness of structure and exploitation of available facilities** and **Desirable extensions** are no longer required and do not attract any marks.

## Project Reports and Presentation

As usual, the presentation of most of the reports was to a very high standard, with reports word-processed and properly bound. However, candidates should ensure that only material essential to the report is included so that there is only one volume of work submitted per candidate. Candidates are reminded that only authentic letters from clients and/or users must be used to provide evidence for the Evaluation, Implementation, Investigation and Analysis sections, these letters must not be typed out by the candidates.

It is strongly recommended that the structure of the candidate's report follows that of the mark scheme set out in the syllabus. Essential evidence should not be relegated to appendices. This allows both teachers at the Centres and Moderators to easily check that work for all sections has been included. Also it is essential that the pages of the report are clearly numbered by the candidate.

## Project assessment and marking

It was pleasing to see that nearly all Centres used the marking grid on pages 48-51 of the syllabus to provide a breakdown of marks showing the marks given for each sub-section of the report. However in order to aid the process of moderation, the completed grid should include references to the appropriate pages in the candidates' reports where evidence for each section can be found. Also teachers should comment as to why they awarded the marks for each section. Moderators have noticed that where there is a good commentary provided by a teacher the marking is usually very close to the agreed standard.

***Section 3***

**Comments on Individual Sections**

The comments set out below identify areas where candidates' work is to be praised or areas of concern and are not a guide to the required contents of each section.

**(a) Quality of report.**

Most candidates set out their reports in the appropriate sections and made good use of illustrations including diagrams and screenshots. Weaker candidates sometimes did not include page numbers in their reports, this meant that teachers could not clearly identify to the Moderator where evidence was to be found and those candidates were unable to cross reference items within their report.

**(b) Definition Investigation and Analysis**

**(i) Definition - nature of the problem**

Most candidates described the organisation and many identified the methods used in the area under investigation but only the better candidates identified the origins and form of the data. This is a brief introduction for anyone who is unfamiliar with the organisation and the area under investigation.

**(ii) Investigation and Analysis**

In order to gain good marks candidates must clearly document client and user involvement and state specific agreed outcomes. Candidates need to consider carefully the evidence obtained from interviews, observation of the existing system and user documents, and then ask follow up questions to fill in any gaps in the knowledge obtained about the current system or requirements for the new system. Also alternative approaches, that the candidate could undertake, for provision of the proposed system need to be discussed in depth. A detailed requirements specification should be produced based on the information collected, this must include the specific requirements of this new system and not just concentrate on hardware and software.

This sub-section of the report remains the same as previous years. However Centres are reminded that a distinction has been made between the 'client', who requires the new system and the day to day 'users' of the system. In many cases the client may also be a user of the system.

**(c) Design**

**(i) Nature of the solution**

The requirements specification set out in the analysis needs to be discussed with the client and a set of measurable objectives agreed. These objectives will then form the basis for the project evaluation. Most candidates provided designs that included proposed data structures, layouts for input screens and reports required, better candidates used pseudocode and/or flowcharts to provide a detailed description of the processes to be implemented. Candidates need to obtain evidence that their client has seen and commented on the design work, and then show what has changed as a result of these comments. Evidence from the solution is not required here.

**(ii) Intended benefits**

Candidates need to describe the benefits of their intended system, not just provide a list of general statements that could apply to any system.

**(iii) Limits of the scope of solution**

Candidates need to describe the limitations of their intended system including an estimate of the size of any files required, not just provide a list of general statements that could apply to any system.

Full marks for the design section cannot be awarded without candidates clearly supplying evidence for **(i)**, **(ii)** and **(iii)**.

**(d) Software Development, Programming Testing and Installation**

**(i) Development**

**Evidence of development should include program listings of code written by the candidate only, data structures used and evidence of tailoring of software packages. This should match the design specification in (c)(i). There is no requirement to include code that has been generated by software.**

**(ii) Programming**

It is important that the programming assessed in this sub-section is written by the candidate and not produced as a result of tailoring part of a software package. Candidates need to show that they can apply the programming skills developed at AS level in Paper 2 to a real situation. This includes technical competence in coding and ensuring that their program could be maintained by writing self-documented code. Marks can only be awarded for code written by the candidate not for any code generated by software.

**(iii) Testing**

Evidence of testing needs to be supported by a well-designed test plan with appropriate test data, including valid, invalid and extreme cases, and expected results for all tests. For top marks to be awarded the test plan should clearly identify that all parts of the system have been tested. Many candidates only tested the validation and navigation aspects of their system, and omitted to test that their system did what it is supposed to do, thus being unable to gain marks in the highest band for this section.

**(iv) Installation**

For good marks, candidates need to provide a detailed implementation plan that contains details of user testing, user training and system changeover together with evidence to show that the system has been tested and used by the client and/or users and these plans have been agreed with the client.

**(iii) Documentation**

**(i) Systems Maintenance Documentation**

This sub-section of the report is now a systems maintenance document. Please see page 43 of the 2012 syllabus for details. For top marks to be awarded the candidate must explain any adaptive maintenance that could be undertaken.

**(ii) User Documentation**

For marks in the top band to be awarded the candidate must include an index and a glossary, the guide needs to be complete including details of how to install the new system, backup routines and a guide to common errors. Also good on-screen help should exist where this is a sensible option.

**(iv) Evaluation**

Centres are again reminded in order to gain high marks candidates need to provide a detailed evaluation as set out in the guidance for marking projects section of the syllabus. Many candidates provided scant evidence for this section, if this is the case then there are few marks that can be awarded.

Centres are again reminded that possible extensions and the good and bad points of their final system are not now required for the report.

**(i) Discussion of the degree of success in meeting the original objectives**

Candidates need to consider each objective set in **(c) (i)** and explain how their project work met the objective or explain why the objective was not met. Candidates should also indicate where the evidence, probably from testing or feedback from the users of the system, could be found in their report to support these conclusions.

**(ii) Evaluate the client's and users' response to the system**

Again Centres are reminded that this response needs to be clearly provided from the client and user(s) showing that they have used the system, not just reported by the candidate. The candidate should then evaluate their client's and users' responses.

For evidence in this section to be creditworthy, the candidate must include original letters, preferably on headed notepaper, signed by the client and not typed and/or composed by the candidate.