COMPUTING

Paper 9691/11
Written Paper

General comments

The format of the paper was different this June as indicated in the June and November 2012 Examiner reports. The emphasis in questions has now been moved from rote learning to more application-based questions. This leads candidates and Centres into the revised syllabus for 2015.

The standard of candidates' work this year needed to be further improved on when compared to last year. Candidates need to understand that simply learning sections from text books will score very few marks on future papers and there is a need to understand how to apply their knowledge in new scenarios (both in examinations and in their learning of the subject). This will hopefully give a better grounding of the subject but also enable candidates of all abilities to perform to their full potential in the examination system.

Comments on specific questions

Question 1

(a) A good start for most candidates; those that had learned the two definitions scored both marks. However, some candidates needed to further develop their understanding of the term interrupt as they thought that an interrupt was a *message* sent to the processor.

(b) (i) Similar questions to this had been asked in a number of previous exam papers. Candidates still need to develop their understanding of buffers and interrupts: for example, many thought that it was the buffer that sent the interrupt, and that the primary memory was the recipient of the interrupt. Many candidates also lost marks for just using the term buffer without indicating *which* buffer data was sent to. A number of candidates also made a brief reference to priorities, but gave no indication of how priorities were used by the processor.

(ii) Candidates needed to improve on their answers to this question. Incorrect answers such as "two buffers mean there is twice as much memory which speeds up printing" or "both buffers send the data to the printer at the same time" were seen. A small number of candidates realised that double buffering enabled the printer to take data from the second buffer while the first was being refilled.

Question 2

(a) Most of the marks here were given for examples of each type of media (e.g. CDs, memory sticks and hard disks). Those candidates that fully understood what the question required achieved all six marks for naming the media types, and for giving a suitable example. There were a number of candidates who thought that floppy disks had a reel of magnetic tape inside them, and even the better candidates struggled to get both marks when the example they gave was magnetic tape. A number of candidates also thought that CDs and DVDS were a form of magnetic media. However, some candidates lost marks for using the term *USB* to describe a type of solid state memory.

(b) (i) Apart from the candidates who got ROM and RAM the wrong way round when describing volatility, most candidates managed at least one mark, and many got both. Some candidates gave two points that were simply the reverse of each other and so limited themselves to just the one mark. When comparisons are made it not worth two marks to write: "the contents or ROM cannot be altered but the contents of RAM can be altered" – these are the same point with the second part of the statement just being the reverse of the first and consequently only one mark can be awarded.

(ii) A number of candidates were able to apply the concepts of RAM and ROM to the toy car. Many correctly said that ROM was needed for the start-up routines and that RAM stored the instructions

received from the remote whilst the car was in use. However, there were some quite different interpretations where the candidates did not seem to realise that it was a toy car, and they mentioned many features that would be found on a real car such as: automatic wipers, CD players and even satellite navigation systems.

Question 3

(a) Both question parts were reasonably well answered. In part **(i)**, the most common errors were using the term *cable* (which is usually a number of wires) instead of *line* (or equivalent) and omitting the word *only* from the definition of simplex, and replacing it with *at a time* which actually implies two way transmission is possible. The answers to part **(ii)** were better with the only notable error being "(in full duplex) data can be sent in two directions" – with the key word *simultaneously* missing from the description; without this word it implies transmission can occur both ways but not necessarily at the same time.

(b)(i)(ii) Both question parts were reasonably well answered. The most common error was to confuse odd parity with even parity.

(iii) Candidates needed to further develop their understanding for this question. None of the candidates actually showed intersecting loops on the diagram which would have helped their description enormously. Quite a few candidates realised that the character 'P' had been received with odd parity, but they did seem to struggle to explain how the erroneous bit would be identified and changed. In order to obtain marks, candidates needed to have written their description more clearly, for example when referring to row numbers or column numbers. Some candidates mixed up vertical and horizontal and got them the wrong way round.

Question 4

(a) The most common answers here were camera and scanner. Unfortunately, a large number of candidates omitted the word digital from the camera which lost them the mark. Other devices such as keyboards, mouse and monitor appeared indicating a lack of understanding of what the question was asking. Very few gained the reason mark with vague answers like: "a scanner is needed to scan the photograph in the passport" – with no indication of the production of an electronic or digital image to allow a comparison of the scanned passport image with the digital camera image to be made.

(b)(i) Candidates needed to improve their understanding of this question. Only a small number of candidates answered in terms of sensors, ADC conversion and digital data sent to a processor. Very few scripts made any reference to a calculation being done by the processor. Many of the answers were very generic, general knowledge type answers about what happens when someone checks their luggage in at an airport. It was very common to see imprecise answers such as the use of a *weight sensor* or "data from a weighing machine was sent to the system" – both of these were very unclear.

(ii) This part question was very well answered with many candidates realising that a barcode label could be used which would be read throughout the airport using a barcode reader allowing tracking of the luggage. A number of candidates suggested OCR which would be difficult to use in this scenario.

Question 5

(a) Data Flow Diagrams (DFDs) or Systems Flowcharts were the most common correct examples chosen here. Very few candidates were able to give a recognisable feature of a Systems Flowchart and described flowcharts in general. Some candidates correctly suggested Jackson Structure Diagrams and usually gained a mark for mention of modularisation.

(b) Many candidates confused Technical Documentation with User Documentation. A number of candidates included elements of the design documentation, such as the system specification and the requirements analysis or even feasibility study. However, candidates who did include items from the Technical Documentation often then failed to get the reason mark because they simply described what the item was and did not say what it would be used for or what its purpose was. Answers such as: "a list of variable names: this is a list of all the names used for the different variables in the program" were very common.

Question 6

(a) Candidates needed to develop their understanding for this question, as few candidates were aware of suitable output devices for each of the four company departments. Answers such as *monitor* or *printer* for use in the Design Office were not sufficient – it would have been acceptable to give, for example: ***large*** monitor (since very complex drawings may need to be edited) or ***3D*** printer (to allow actual prototypes to be made). Candidates needed to be more specific in all four parts of the question to gain marks.

(b) Candidates needed to develop their understanding for this part question. Some candidates appeared to have learned a previous mark scheme answer for this question and gave answers such as: *use of contrasting colours, do not use red green because of colour blindness, must take into account normal reading pattern of left to right*, and so on. There was very little real reference to an interface which would need to be used to monitor and control a manufacturing process. One or two candidates did try to answer the question and included: *use of touch screens and splitting the screen to show each process separately then being able to go to each one with a single touch*; all of which deserved credit.

Question 7

(a) (i) A number of candidates tried to explain the concept of each square in the game corresponding to a position in the array, but many of the explanations were imprecise and did not gain any marks. Those who tried to write code very rarely gave a good answer; but this would have been the best way to answer this question. Some candidates did obtain marks for correctly giving the array references of the symbols already in the grid. However, a significant number of candidates made no attempt at all to answer the question.

(ii) It was very common to see *Boolean* here as a data type. Presumably, these candidates considered only "X" or "O" (or "1" or "0") to be the only possible data – and clearly had not considered what was being asked in part **(iii)**

(iii) This part question was reasonably well answered with either *null* or <*space*> being the most common correct answers.

(b) Candidates needed to improve their understanding for this question. A lot of the answers were generic and gave no indication that the computer was doing any checking. Quite a few candidates thought that the checking would be done when all nine squares were filled in rather than after each player's turn. Some candidates wrote that the array did the checking. One or two candidates attempted to give an algorithm; most of these had the concept of looping for the row and column, but did not get as far as setting flags or checking adjacent cells.

(c) This question was very well answered with mouse or touch screen chosen by the majority of candidates.

Question 8

(a) Many candidates who had some idea about hashing algorithms unfortunately made reference to files instead of records. There seemed to be a general misconception that individual data items were being stored, not complete records. Very few candidates actually mentioned the fact that the hashing was being done to a key field.

(b) This part of the question was answered better than part **(a)**. Quite a few candidates realised that the record would be put in the next available free space. Some gave overflow or bucket as the answer, but did not seem to know about setting a tag from the original location. One or two correctly identified a linked list as a method, but very few went on to get the second mark. There were quite a number of answers along the lines of "if the hashing algorithm produces a collision, then use a different algorithm".

Question 9

(a) This question was very well answered; however, some of the drawings of the logic gates needed further improvement. Candidates need to realise that Examiners have to interpret the shapes of the logic gates that are drawn – for example, if the gate drawn was a mixture of an AND gate and an OR gate (which was unfortunately fairly common) then no mark could be awarded. Also marks were lost for careless mistakes such as not drawing the NOT gate properly. Some candidates used circles with the words NOT, AND and OR inside the circles; this is acceptable at the moment.

(b) This question was very well answered with marks from 0 to 4 seen.

COMPUTING

Paper 9691/12
Written Paper

General comments

The format of the paper was different this June as indicated in the June and November 2012 Examiner reports. The emphasis in questions has now been moved from rote learning to more application-based questions. This leads candidates and Centres into the revised syllabus for 2015.

The standard of candidates' work this year needed to be further improved on when compared to last year. Candidates need to understand that simply learning sections from text books will score very few marks on future papers and there is a need to understand how to apply their knowledge in new scenarios (both in examinations and in their learning of the subject). This will hopefully give a better grounding of the subject but also enable candidates of all abilities to perform to their full potential in the examination system.

Comments on specific questions

Question 1

(a) A good start for most candidates; those that had learned the two definitions scored both marks. However, some candidates needed to further develop their understanding of the term interrupt as they thought that an interrupt was a *message* sent to the processor.

(b) (i) Similar questions to this had been asked in a number of previous exam papers. Candidates still need to develop their understanding of buffers and interrupts: for example, many thought that it was the buffer that sent the interrupt, and that the primary memory was the recipient of the interrupt. Many candidates also lost marks for just using the term buffer without indicating *which* buffer data was sent to. A number of candidates also made a brief reference to priorities, but gave no indication of how priorities were used by the processor.

(ii) Candidates needed to improve on their answers to this question. Incorrect answers such as "two buffers mean there is twice as much memory which speeds up printing" or "both buffers send the data to the printer at the same time" were seen. A small number of candidates realised that double buffering enabled the printer to take data from the second buffer while the first was being refilled.

Question 2

(a) Most of the marks here were given for examples of each type of media (e.g. CDs, memory sticks and hard disks). Those candidates that fully understood what the question required achieved all six marks for naming the media types, and for giving a suitable example. There were a number of candidates who thought that floppy disks had a reel of magnetic tape inside them, and even the better candidates struggled to get both marks when the example they gave was magnetic tape. A number of candidates also thought that CDs and DVDS were a form of magnetic media. However, some candidates lost marks for using the term *USB* to describe a type of solid state memory.

(b) (i) Apart from the candidates who got ROM and RAM the wrong way round when describing volatility, most candidates managed at least one mark, and many got both. Some candidates gave two points that were simply the reverse of each other and so limited themselves to just the one mark. When comparisons are made it not worth two marks to write: "the contents or ROM cannot be altered but the contents of RAM can be altered" – these are the same point with the second part of the statement just being the reverse of the first and consequently only one mark can be awarded.

(ii) A number of candidates were able to apply the concepts of RAM and ROM to the toy car. Many correctly said that ROM was needed for the start-up routines and that RAM stored the instructions

received from the remote whilst the car was in use. However, there were some quite different interpretations where the candidates did not seem to realise that it was a toy car, and they mentioned many features that would be found on a real car such as: automatic wipers, CD players and even satellite navigation systems.

Question 3

(a) Both question parts were reasonably well answered. In part **(i)**, the most common errors were using the term *cable* (which is usually a number of wires) instead of *line* (or equivalent) and omitting the word *only* from the definition of simplex, and replacing it with *at a time* which actually implies two way transmission is possible. The answers to part **(ii)** were better with the only notable error being "(in full duplex) data can be sent in two directions" – with the key word *simultaneously* missing from the description; without this word it implies transmission can occur both ways but not necessarily at the same time.

(b)(i)(ii) Both question parts were reasonably well answered. The most common error was to confuse odd parity with even parity.

(iii) Candidates needed to further develop their understanding for this question. None of the candidates actually showed intersecting loops on the diagram which would have helped their description enormously. Quite a few candidates realised that the character 'P' had been received with odd parity, but they did seem to struggle to explain how the erroneous bit would be identified and changed. In order to obtain marks, candidates needed to have written their description more clearly, for example when referring to row numbers or column numbers. Some candidates mixed up vertical and horizontal and got them the wrong way round.

Question 4

(a) The most common answers here were camera and scanner. Unfortunately, a large number of candidates omitted the word digital from the camera which lost them the mark. Other devices such as keyboards, mouse and monitor appeared indicating a lack of understanding of what the question was asking. Very few gained the reason mark with vague answers like: "a scanner is needed to scan the photograph in the passport" – with no indication of the production of an electronic or digital image to allow a comparison of the scanned passport image with the digital camera image to be made.

(b)(i) Candidates needed to improve their understanding of this question. Only a small number of candidates answered in terms of sensors, ADC conversion and digital data sent to a processor. Very few scripts made any reference to a calculation being done by the processor. Many of the answers were very generic, general knowledge type answers about what happens when someone checks their luggage in at an airport. It was very common to see imprecise answers such as the use of a *weight sensor* or "data from a weighing machine was sent to the system" – both of these were very unclear.

(ii) This part question was very well answered with many candidates realising that a barcode label could be used which would be read throughout the airport using a barcode reader allowing tracking of the luggage. A number of candidates suggested OCR which would be difficult to use in this scenario.

Question 5

(a) Data Flow Diagrams (DFDs) or Systems Flowcharts were the most common correct examples chosen here. Very few candidates were able to give a recognisable feature of a Systems Flowchart and described flowcharts in general. Some candidates correctly suggested Jackson Structure Diagrams and usually gained a mark for mention of modularisation.

(b) Many candidates confused Technical Documentation with User Documentation. A number of candidates included elements of the design documentation, such as the system specification and the requirements analysis or even feasibility study. However, candidates who did include items from the Technical Documentation often then failed to get the reason mark because they simply described what the item was and did not say what it would be used for or what its purpose was. Answers such as: "a list of variable names: this is a list of all the names used for the different variables in the program" were very common.

Question 6

(a) Candidates needed to develop their understanding for this question, as few candidates were aware of suitable output devices for each of the four company departments. Answers such as *monitor* or *printer* for use in the Design Office were not sufficient – it would have been acceptable to give, for example: ***large*** monitor (since very complex drawings may need to be edited) or ***3D*** printer (to allow actual prototypes to be made). Candidates needed to be more specific in all four parts of the question to gain marks.

(b) Candidates needed to develop their understanding for this part question. Some candidates appeared to have learned a previous mark scheme answer for this question and gave answers such as: *use of contrasting colours, do not use red green because of colour blindness, must take into account normal reading pattern of left to right*, and so on. There was very little real reference to an interface which would need to be used to monitor and control a manufacturing process. One or two candidates did try to answer the question and included: *use of touch screens and splitting the screen to show each process separately then being able to go to each one with a single touch*; all of which deserved credit.

Question 7

(a) (i) A number of candidates tried to explain the concept of each square in the game corresponding to a position in the array, but many of the explanations were imprecise and did not gain any marks. Those who tried to write code very rarely gave a good answer; but this would have been the best way to answer this question. Some candidates did obtain marks for correctly giving the array references of the symbols already in the grid. However, a significant number of candidates made no attempt at all to answer the question.

(ii) It was very common to see *Boolean* here as a data type. Presumably, these candidates considered only "X" or "O" (or "1" or "0") to be the only possible data – and clearly had not considered what was being asked in part **(iii)**

(iii) This part question was reasonably well answered with either *null* or <*space*> being the most common correct answers.

(b) Candidates needed to improve their understanding for this question. A lot of the answers were generic and gave no indication that the computer was doing any checking. Quite a few candidates thought that the checking would be done when all nine squares were filled in rather than after each player's turn. Some candidates wrote that the array did the checking. One or two candidates attempted to give an algorithm; most of these had the concept of looping for the row and column, but did not get as far as setting flags or checking adjacent cells.

(c) This question was very well answered with mouse or touch screen chosen by the majority of candidates.

Question 8

(a) Many candidates who had some idea about hashing algorithms unfortunately made reference to files instead of records. There seemed to be a general misconception that individual data items were being stored, not complete records. Very few candidates actually mentioned the fact that the hashing was being done to a key field.

(b) This part of the question was answered better than part **(a)**. Quite a few candidates realised that the record would be put in the next available free space. Some gave overflow or bucket as the answer, but did not seem to know about setting a tag from the original location. One or two correctly identified a linked list as a method, but very few went on to get the second mark. There were quite a number of answers along the lines of "if the hashing algorithm produces a collision, then use a different algorithm".

Question 9

(a) This question was very well answered; however, some of the drawings of the logic gates needed further improvement. Candidates need to realise that Examiners have to interpret the shapes of the

logic gates that are drawn – for example, if the gate drawn was a mixture of an AND gate and an OR gate (which was unfortunately fairly common) then no mark could be awarded. Also marks were lost for careless mistakes such as not drawing the NOT gate properly. Some candidates used circles with the words NOT, AND and OR inside the circles; this is acceptable at the moment.

(b) This question was very well answered with marks from 0 to 4 seen.

COMPUTING

Paper 9691/13

Written Paper

General comments

The format of the paper was different this June as indicated in the June and November 2012 Examiner reports. The emphasis in questions has now been moved from rote learning to more application-based questions. This leads candidates and Centres into the revised syllabus for 2015.

The standard of candidates' work this year needed to be further improved on when compared to last year. Candidates need to understand that simply learning sections from text books will score very few marks on future papers and there is a need to understand how to apply their knowledge in new scenarios (both in examinations and in their learning of the subject). This will hopefully give a better grounding of the subject, but also enable candidates of all abilities to perform to their full potential in the examination system.

Comments on specific questions

Question 1

(a) This question was reasonably well answered; although it was fairly common to see: "it is a system that operates the computer" which was essentially just a simple re-wording of the question.

(b) Most candidates seemed to know about booking systems and gained marks here for reference to double booking if the system was not fast enough to update in real time. However, many candidates then gave drawing money from an ATM as a second application. Unfortunately, the question stated clearly that the applications had to come from two different types of real time systems; namely, real time transaction processing and real time process control. Consequently, all the marks for the second application were lost. The most common, correct, application for real time process control was the monitoring and control of a chemical process.

(c) This was fairly well answered with many candidates fully aware of the reasons why many household appliances did not need an operating system.

Question 2

(a) Similar questions to this had been asked in a number of previous exam papers. Candidates still need to develop their understanding of buffers and interrupts: for example, many thought that it was the buffer that sent the interrupt, and that the primary memory was the recipient of the interrupt. Many candidates also lost marks for just using the term buffer without indicating *which* buffer data was sent to. A number of candidates also made a brief reference to priorities, but gave no indication of how priorities were used by the processor.

(b) Candidates needed to improve on their answers to this question. There were some unclear and general answers here, such as: "manages computer's instructions". In the description of the functions of a *memory unit*, many candidates missed out the key phrase "currently in use". For example, to state that *the memory unit stores the programs* is not enough since the hard disk or a CD could do the same thing. It is the fact that it's the *program currently in use* that gains the mark.

Question 3

(a) Candidates needed to improve their understanding of this question. Only a small number of candidates answered in terms of sensors, ADC conversion and digital data sent to a processor. Incorrect responses included that sensors interpret data (and not the processor) and, in some cases, can even prevent the earthquake from happening. Such answers clearly indicate a lack of understanding of how these monitoring systems work in general. Unfortunately, there was strong evidence that mark schemes from previous papers had been learnt and the answers given were not adapted to this different scenario.

(b) (i) Candidates needed to improve on their responses to this part question. Several candidates appeared to have learned a previous mark scheme answer for this question and gave answers such as: *use of contrasting colours, do not use red green because of colour blindness, must take into account normal reading pattern of left to right*, and so on. There was very little real reference to an interface which would need to be used to monitor the data coming from the earthquake sensors (such as maps of the area with seismic activity superimposed onto the maps). One or two candidates did try to answer the question and included: *use of touch screens and splitting the screen to show each area being monitored separately then being able to go to each area with a single touch*; all of which deserved credit.

(ii) This was reasonably well answered with touch screens and keyboards being the most common input devices. However, the justifications for the chosen devices were a little unclear at times e.g. "a touch screen can be used as an input/output device" which did not really justify the choice of device. Acceptable answers could include: *makes it easy to select an area on a map shown on the screen* or *it is easy to make on screen selections such as upload a previous graph.*

Question 4

This was generally well answered, although it was common to see the words printer or screen. Such answers are not specialist devices as asked for in the question. Acceptable answers could include: **large** monitors (since very complex drawings may need to be edited), (graph) plotters to produce blueprints) or **3D** printers (to allow actual prototypes to be made).

Question 5

(a) Many candidates gained a mark for a declaration of the two-dimensional array; a few gained a second mark for giving an example of how the table values could be input. One or two candidates attempted an algorithm (which was a good way to answer this question) but very few gained any marks. A number of candidates also chose to represent the table as a one-dimensional array with 18 elements.

(b) This was generally well answered except for those candidates who had chosen to use a one-dimensional array in part **(a)**.

(c) Candidates needed to improve on their answers to this question. Better candidates attempted to answer the question using an algorithm (which is a good approach) but lost marks for incorrect matching criteria e.g. array value < 0 – the question indicated that valid numbers had to be greater than zero thus it was required to check if the value was > 0 OR < = 0.

Question 6

(a) This question was very well answered; however, some of the drawings of the logic gates needed further improvement. Candidates need to realise that Examiners have to interpret the shapes of the logic gates that are drawn – for example, if the gate drawn was a mixture of an AND gate and an OR gate (which was unfortunately fairly common) then no mark could be awarded. Also marks were lost for careless mistakes such as not drawing the NOT gate properly. Some candidates used circles with the words NOT, AND and OR inside the circles; this is acceptable at the moment. It was also common to see inputs to AND gates and OR gates being joined together so effectively these gates were one input gates.

(b) Very well answered with the marks from 0 to 4 seen.

Question 7

(a) Many candidates ignored the stem of the question and gave answers on file structures. There were some imprecise answers that made reference to stock control, but very few answered the question and gave examples of design such as the *user interface* or *the type of hardware or software needed.*

(b) Candidates needed to improve on their answers to this question. There were references to *normal data*, *abnormal data* and *extreme data* references. Very few candidates mentioned comparing actual results with expected results and how the results could be used as part of the testing strategy. It is very important that candidates gain practical experience in this course, as the practical element will help them develop better answers to this type of question.

Question 8

(a) Candidates made a good attempt to answer this question well. However, in part **(ii)** a number of candidates reversed the row number and column number. In part **(iii)** there seemed to be some confusion as to what is meant by *even* or *odd parity*. Many candidates claimed that if the left-most bit was a "0" then the number must be odd parity and if it was a "1" then the number must be even parity. This brought them the conclusion that letter "A" must be *even parity*; by the same logic, letter "G" should have *even parity* as well. This showed that candidates needed to further develop their understanding of how the parity of a number is determined and also the role of the parity bit.

(b) This was reasonably well answered, but some candidates confused checksum with parity.

Question 9

(a) Candidates needed to improve on their answers to this question. There were many general comments made about stock control which were too imprecise to gain any marks; for example:

- a barcode is read
- the number in stock is reduced
- new stock is ordered when the level gets to zero
- the manager is informed so he can order the new stock

Such answers are either imprecise or actually incorrect, but were all too common from a number of candidates. There was very little reference to how the barcode is used to locate the product in the stock file (used as the key field), how the stock levels are maintained (e.g. by writing the new stock value to the record) or that when stock levels reach the re-order level automatic reordering is done (either by contacting the warehouse directly or by printing out order forms).

(b) This part of the question was very well answered.

COMPUTING

Paper 9691/21

Written Paper

Key message: To do well in this component, candidates need to have practical programming experience in the programing facilities of a chosen high-level language (such as Basic, Visual Basic or Pascal). Candidates need to design and write their own programs, adapt programs written by others, and write programs from pseudocode prepared by others. Candidates who performed well in this paper were the ones who had done a reasonable amount of programming, had produced flowcharts, and had completed trace tables.

Question 1

(a) The better candidates gave answers which included points such as: modularisation allows debugging of a small section at a time; makes the program easier to maintain; produces reusable code. Weaker answers did not distinguish between easy to debug and easier to debug. Candidates need to understand the difference. A significant number of candidates thought that more than one programmer could be employed here. Candidates need to appreciate the scenario given in the question and not rely too much on answers seen in mark schemes of past examination papers.

(b) Many creditworthy choices of key field identifier were seen, such as `CourseworkID`. The scenario made choices such as `SubjectCode` or `CandidateID` inappropriate. Suitable data types were Integer or String.

Candidates who had practical experience of programming with records, stood out in this question and many correct answers were seen. Some candidates seem to be unsure which programming language they were using as some answers were an amalgamation of Visual Basic 6 and VB.net. The question clearly listed the identifiers and candidates were expected to use these in their answers. The weaker answers did not use data types recognised by the stated programming language (such as Text or Date/Time).

The majority of candidates correctly stated that 1 byte would be needed to store the value of `IsMarked`.

(c) Very few creditworthy answers were seen for this question. Candidates need practical experience of using the `EOF` function, preferably also stepping through their program to see the effect. It was not sufficient to state that `EOF` stood for End Of File. The question asked what this function does: it detects a marker written to the file immediately after the last record. Whenever the function is called it returns a Boolean value to report whether or not this marker has been reached.

(d) Most candidates made a good attempt at this question. The better candidates showed a systematic approach of first setting a Boolean variable to False and then setting it to True when the Physics assignment was found. The most appropriate loop structure is a `WHILE` loop as this will allow for the case of an empty file (`WHILE NOT EOF`). Many candidates forgot to read the next record from the file within the loop.

Question 2

(a) Many candidates correctly wrote (`IsMarked = 'Y'`) OR (`IsMarked = 'N'`). Candidates need to appreciate the need for repeating the variable name in the second part of the expression.

(b) The better candidates drew clear flowcharts with decision boxes (diamond shapes) with exits clearly labelled with Yes and No. Some candidates realised that the input would be a single string and that the separate parts of the date would have to be extracted using string handling functions. Very few answers included changing these substrings into integers so that the ranges could be

checked effectively. Some flowcharts also included setting a Boolean variable to True or False depending on whether the validation was successful or not.

Candidates need to further develop their understanding of flowcharts as there were a significant number of diagrams that were not recognisable as program flowcharts.

(c) The answers here showed that not all candidates read the question carefully enough. Candidates were asked to enter 'normal' or 'borderline' into the given table. Many answered 'valid' and 'invalid' instead.

Only the better candidates could clearly state the reason why the given test data was not a good example of invalid data: you cannot tell which of the three components is invalid. Many incorrect answers suggested that all parts of the date should be outside the valid range. This often resulted in not providing better examples of invalid test data in part **(iii)**.

(d) Many candidates correctly wrote (`HandInDate > DateSet`) AND (`HandInDate > CurrentDate`). Candidates need to appreciate the need for repeating the variable name `HandInDate` in the second part of the expression.

(e) Candidates need more practical experience of programming with nested `IF` statements. Although this question required an answer using pseudocode, programming experience will help candidates arrive at an answer such as:

```
Valid ← FALSE
IF DateReturned > HandInDate
  THEN
    IF DateReturned <= CurrentDate
      THEN
        IF (Mark >= 0) AND (Mark <= 100)
          THEN Valid ← TRUE
        ENDIF
    ENDIF
ENDIF
```

(f) (i)-(ii) The majority of candidates were able to complete the trace table correctly. Rather fewer candidates correctly explained that the output from this pseudocode is the number of assignments with a mark less than 40. Many incorrect answers only referred to the test data given in the question. Nowhere in the question was any reference to the mark of 40 being a boundary of pass/fail, but many answers referred to how many assignments Meena had failed. Candidates need to be careful not to read more into a question than is actually provided.

(iii)-(v) The majority of candidates correctly identified sensible variable names and indentation as the features that make the given pseudocode easier to understand. Most also gave comments or annotation as another desirable feature and provided a pseudocode statement with a suitable comment.

(vi)-(vii) Many candidates correctly replaced the `REPEAT … UNTIL` statements for `WHILE … ENDWHILE` statements. Fewer candidates provided the correct Boolean expression for this. A common mistake was not to include all the other statements within the loop that were given in the original pseudocode. The better candidates could give an acceptable reason why a `FOR –ENDFOR` loop would not be appropriate. Practical experience should help candidates appreciate that the number of records in the file must be known before a `FOR` loop is executed, which clearly is not the case in this scenario.

Question 3

(a) It was not sufficient to state that global variables are declared in the main program. Candidates need to be aware that this must be at the beginning of the main program code.

(b) Very few good answers were seen here. It was not sufficient to state that different modules could change the value in a global variable or that errors could occur. Creditworthy answers referred to the fact that it was difficult to find where variable values were changed when an error was found.

The better answers included: re-use of modules is more difficult; two threads running simultaneously could try to modify the value.

(c) Many imprecise answers were seen here. Candidates need to make it clear that the scope of a local variable is the module/block in which it is declared (not just used).

(d) Only the better candidates noticed that initialising the array Marks would need a dummy or rogue value: an integer other than those in the range 0 to 100, as these are valid marks for the assignments. A suitable initial value would be –1. Null is also a valid method of initialising this array, although it must not be confused with the value zero. Candidates need to appreciate that as this array was required to store integers, an empty string or a space was not a suitable initial value.

(e) Candidates who had practical experience or programming showed no difficulty of writing an array declaration and a loop to initialise the array.

(f) Most candidates made some attempt at a loop and a running total of marks. There were few attempts at ignoring the marks element of assignments that had not been marked yet. As the array would not necessarily contain marks in all elements, the average mark could not be arrived at by dividing the total mark by 30. Candidates needed to include a count of how many mark elements were other than the initial value.

(g) The better candidates could correctly distinguish between a function and a procedure: a function always returns one value. A procedure may return either none, one or several values (by using reference parameters). Many candidates correctly reasoned that the code written in part **(f)** could be written as a function because just one value, `AvMark`, needed to be returned to the calling program.

(h) Most candidates gave the correct answers to parts **(i)** and **(ii)**. Only the better candidates could combine the information given in the previous parts to write a user-defined function CalculateRounded. This showed again that those with practical experience could provide good answers, including how to write the function header with one real parameter and how to return the calculated integer value.

Question 4

(a) Sound output, voice recognition, and enlarged fonts were the most frequently credited answers. Hardware such as microphone, speakers and braille keyboards were not given credit as these are not software design features.

(b) There seems to be some confusion about how syntax errors are detected. Testing is not appropriate here as the program will not even compile when there are syntax errors. Only the better candidates mentioned integrated development environments (IDEs) which recognise syntax errors while code is typed into the editor. Other credit-worthy answers referred to compiler/interpreter/translator checking that the rules of the language are being followed.

Candidates mostly referred to the presence of logic errors when unexpected results occur and finding such errors using black-box or white-box testing.

COMPUTING

Paper 9691/22

Written Paper

Key message: To do well in this component, candidates need to have practical programming experience in the programing facilities of a chosen high-level language (such as Basic, Visual Basic or Pascal). Candidates need to design and write their own programs, adapt programs written by others, and write programs from pseudocode prepared by others. Candidates who performed well in this paper were the ones who had done a reasonable amount of programming, had produced flowcharts, and had completed trace tables.

Question 1

(a) The better candidates gave answers which included points such as: modularisation allows debugging of a small section at a time; makes the program easier to maintain; produces reusable code. Weaker answers did not distinguish between easy to debug and easier to debug. Candidates need to understand the difference. A significant number of candidates thought that more than one programmer could be employed here. Candidates need to appreciate the scenario given in the question and not rely too much on answers seen in mark schemes of past examination papers.

(b) Many creditworthy choices of key field identifier were seen, such as `CourseworkID`. The scenario made choices such as `SubjectCode` or `CandidateID` inappropriate. Suitable data types were Integer or String.

Candidates who had practical experience of programming with records, stood out in this question and many correct answers were seen. Some candidates seem to be unsure which programming language they were using as some answers were an amalgamation of Visual Basic 6 and VB.net. The question clearly listed the identifiers and candidates were expected to use these in their answers. The weaker answers did not use data types recognised by the stated programming language (such as Text or Date/Time).

The majority of candidates correctly stated that 1 byte would be needed to store the value of `IsMarked`.

(c) Very few creditworthy answers were seen for this question. Candidates need practical experience of using the `EOF` function, preferably also stepping through their program to see the effect. It was not sufficient to state that `EOF` stood for End Of File. The question asked what this function does: it detects a marker written to the file immediately after the last record. Whenever the function is called it returns a Boolean value to report whether or not this marker has been reached.

(d) Most candidates made a good attempt at this question. The better candidates showed a systematic approach of first setting a Boolean variable to False and then setting it to True when the Physics assignment was found. The most appropriate loop structure is a `WHILE` loop as this will allow for the case of an empty file (`WHILE NOT EOF`). Many candidates forgot to read the next record from the file within the loop.

Question 2

(a) Many candidates correctly wrote (`IsMarked = 'Y'`) OR (`IsMarked = 'N'`). Candidates need to appreciate the need for repeating the variable name in the second part of the expression.

(b) The better candidates drew clear flowcharts with decision boxes (diamond shapes) with exits clearly labelled with Yes and No. Some candidates realised that the input would be a single string and that the separate parts of the date would have to be extracted using string handling functions. Very few answers included changing these substrings into integers so that the ranges could be

checked effectively. Some flowcharts also included setting a Boolean variable to True or False depending on whether the validation was successful or not.

Candidates need to further develop their understanding of flowcharts as there were a significant number of diagrams that were not recognisable as program flowcharts.

(c) The answers here showed that not all candidates read the question carefully enough. Candidates were asked to enter 'normal' or 'borderline' into the given table. Many answered 'valid' and 'invalid' instead.

Only the better candidates could clearly state the reason why the given test data was not a good example of invalid data: you cannot tell which of the three components is invalid. Many incorrect answers suggested that all parts of the date should be outside the valid range. This often resulted in not providing better examples of invalid test data in part **(iii)**.

(d) Many candidates correctly wrote (`HandInDate > DateSet`) AND (`HandInDate > CurrentDate`). Candidates need to appreciate the need for repeating the variable name `HandInDate` in the second part of the expression.

(e) Candidates need more practical experience of programming with nested `IF` statements. Although this question required an answer using pseudocode, programming experience will help candidates arrive at an answer such as:

```
Valid ← FALSE
IF DateReturned > HandInDate
  THEN
    IF DateReturned <= CurrentDate
      THEN
        IF (Mark >= 0) AND (Mark <= 100)
          THEN Valid ← TRUE
        ENDIF
    ENDIF
ENDIF
```

(f) (i)-(ii) The majority of candidates were able to complete the trace table correctly. Rather fewer candidates correctly explained that the output from this pseudocode is the number of assignments with a mark less than 40. Many incorrect answers only referred to the test data given in the question. Nowhere in the question was any reference to the mark of 40 being a boundary of pass/fail, but many answers referred to how many assignments Meena had failed. Candidates need to be careful not to read more into a question than is actually provided.

(iii)-(v) The majority of candidates correctly identified sensible variable names and indentation as the features that make the given pseudocode easier to understand. Most also gave comments or annotation as another desirable feature and provided a pseudocode statement with a suitable comment.

(vi)-(vii) Many candidates correctly replaced the `REPEAT … UNTIL` statements for `WHILE … ENDWHILE` statements. Fewer candidates provided the correct Boolean expression for this. A common mistake was not to include all the other statements within the loop that were given in the original pseudocode. The better candidates could give an acceptable reason why a `FOR –ENDFOR` loop would not be appropriate. Practical experience should help candidates appreciate that the number of records in the file must be known before a `FOR` loop is executed, which clearly is not the case in this scenario.

Question 3

(a) It was not sufficient to state that global variables are declared in the main program. Candidates need to be aware that this must be at the beginning of the main program code.

(b) Very few good answers were seen here. It was not sufficient to state that different modules could change the value in a global variable or that errors could occur. Creditworthy answers referred to the fact that it was difficult to find where variable values were changed when an error was found.

The better answers included: re-use of modules is more difficult; two threads running simultaneously could try to modify the value.

(c) Many imprecise answers were seen here. Candidates need to make it clear that the scope of a local variable is the module/block in which it is declared (not just used).

(d) Only the better candidates noticed that initialising the array Marks would need a dummy or rogue value: an integer other than those in the range 0 to 100, as these are valid marks for the assignments. A suitable initial value would be –1. Null is also a valid method of initialising this array, although it must not be confused with the value zero. Candidates need to appreciate that as this array was required to store integers, an empty string or a space was not a suitable initial value.

(e) Candidates who had practical experience or programming showed no difficulty of writing an array declaration and a loop to initialise the array.

(f) Most candidates made some attempt at a loop and a running total of marks. There were few attempts at ignoring the marks element of assignments that had not been marked yet. As the array would not necessarily contain marks in all elements, the average mark could not be arrived at by dividing the total mark by 30. Candidates needed to include a count of how many mark elements were other than the initial value.

(g) The better candidates could correctly distinguish between a function and a procedure: a function always returns one value. A procedure may return either none, one or several values (by using reference parameters). Many candidates correctly reasoned that the code written in part **(f)** could be written as a function because just one value, `AvMark`, needed to be returned to the calling program.

(h) Most candidates gave the correct answers to parts **(i)** and **(ii)**. Only the better candidates could combine the information given in the previous parts to write a user-defined function CalculateRounded. This showed again that those with practical experience could provide good answers, including how to write the function header with one real parameter and how to return the calculated integer value.

Question 4

(a) Sound output, voice recognition, and enlarged fonts were the most frequently credited answers. Hardware such as microphone, speakers and braille keyboards were not given credit as these are not software design features.

(b) There seems to be some confusion about how syntax errors are detected. Testing is not appropriate here as the program will not even compile when there are syntax errors. Only the better candidates mentioned integrated development environments (IDEs) which recognise syntax errors while code is typed into the editor. Other credit-worthy answers referred to compiler/interpreter/translator checking that the rules of the language are being followed.

Candidates mostly referred to the presence of logic errors when unexpected results occur and finding such errors using black-box or white-box testing.

COMPUTING

Paper 9691/23
Written Paper

Key message: To do well in this component, candidates need to have practical programming experience in the programing facilities of a chosen high-level language (such as Basic, Visual Basic or Pascal). Candidates need to design and write their own programs, adapt programs written by others, and write programs from pseudocode prepared by others. Candidates who performed well in this paper were the ones who had done a reasonable amount of programming, had produced pseudocode, and had completed trace tables.

Question 1

(a) Most candidates answered this question well.

(b) Most candidates answered this part question well. However, some candidates needed to be better prepared as the arithmetic had to be done in a different order from the way the question has been asked in the past.

(c) Many creditworthy choices of key field identifier were seen, such as `ExamID`. The scenario made choices such as `SubjectCode` or `StudentID` inappropriate. Suitable data types were Integer or String.

Candidates who had practical experience of programming with records, stood out in this question and many correct answers were seen. Some candidates seem to be unsure which programming language they were using as some answers were an amalgamation of Visual Basic 6 and VB.net. The question clearly listed the identifiers and candidates were expected to use these in their answers. The weaker answers did not use data types recognised by the stated programming language (such as Text or Date/Time).

(d) The better candidates gave answers which included points such as: modularisation allows debugging of a small section at a time; makes the program easier to maintain; produces reusable code. Weaker answers did not distinguish between easy to debug and easier to debug. Candidates need to understand the difference. A significant number of candidates thought that more than one programmer could be employed here. Candidates need to appreciate the scenario given in the question and not rely too much on answers seen in mark schemes of past examination papers.

(e) Very few creditworthy answers were seen for this question. Candidates need practical experience of using the `EOF` function, preferably also stepping through their program to see the effect. It was not sufficient to state that `EOF` stood for End Of File. The question asked what this function does: it detects a marker written to the file immediately after the last record. Whenever the function is called it returns a Boolean value to report whether or not this marker has been reached.

Question 2

(a) Many candidates correctly wrote (`Mark>=0`) OR (`Mark<=100`). Candidates need to appreciate the need for repeating the variable name in the second part of the expression. Full marks were awarded for those candidates who gave the reverse of this statement followed by an indication that this was invalid.

(b) The majority of candidates were able to complete the trace table correctly, correct it and give the type of error. Rather fewer candidates correctly explained that the output from this pseudocode is the number of assignments with a mark greater than 70. Many incorrect answers only referred to the test data given in the question. Nowhere in the question was any reference to the mark of 70 being a boundary of pass/distinction, but many answers referred to how many assignments Meena

had gained a distinction on. Candidates need to be careful not to read more into a question than is actually provided.

(c) Many candidates correctly replaced the `REPEAT … UNTIL` statements for `WHILE … ENDWHILE` statements. Fewer candidates provided the correct Boolean expression for this. A common mistake was not to include all the other statements within the loop that were given in the original pseudocode. The better candidates could give an acceptable reason why a `FOR –ENDFOR` loop would not be appropriate. Practical experience should help candidates appreciate that the number of records in the file must be known before a `FOR` loop is executed, which clearly is not the case in this scenario.

Question 3

(a) It was not sufficient to state that global variables are declared in the main program. Candidates need to be aware that this must be at the beginning of the main program code.

(b) Very few good answers were seen here. It was not sufficient to state that different modules could change the value in a global variable or that errors could occur. Creditworthy answers referred to the fact that it was difficult to find where variable values were changed when an error was found.

The better answers were: re-use of modules is more difficult; two threads running simultaneously could try to modify the value.

(c) The better candidates noticed that initialising the array Marks would need a dummy or rogue value: an integer other than those in the range 0 to 100, as these are valid marks for the assignments. A suitable initial value would be -1. Null is also a valid method of initialising this array, although it must not be confused with the value zero. Candidates need to appreciate that as this array was required to store integers, an empty string or a space was not a suitable initial value.

(d) Candidates who had practical experience of programming showed no difficulty of writing an array declaration and a loop to initialise the array.

(e) Many candidates answered this question well. Their response indicated that they had programmed this sort of problem.

(f) Candidates needed to improve their knowledge of functions.

(g) The better candidates could correctly distinguish between a function and a procedure: a function always returns one value. A procedure may return either none, one or several values (by using reference parameters). Many candidates correctly reasoned that the sub-routine to produce the average mark could be written as a function because just one value, AvMark, needed to be returned to the calling program.

Question 4

(a) Sound output and voice recognition and enlarged fonts were the most frequently credited answers. Hardware such as microphone, speakers and braille keyboards were not given credit as these are not software design features.

(b) Candidates gained marks for using the ideas they had put forward in **(a)**. The better designs were simple, clear and showed the possibility of changing font size and starting sound options.

(c) This question was answered well.

Question 5

Candidates need more practical experience of programming with nested `IF` statements. Although this question required an answer using pseudocode, programming experience will help candidates to better understand how to use `IF, ENDIF` and `ELSEIF` statements.

COMPUTING

Paper 9691/31
Written Paper

General

On topics where issues had been highlighted in previous reports there was evidence of a marked improvement in the quality of candidate answers. Where a different question framework was used to assess candidates' understanding – for example, **Question 4(c)(ii)** – candidates coped well this, with a number of candidates scoring full marks on a topic which is considered challenging. This was less so on **Question 8**, when candidates were not asked for the definition of a robot, but given the definition and then asked to apply these statements to their chosen example. The answers provided needed to show further understanding.

Question 1

Most candidates were able to score well on this question. Part **(a)** required the candidate to recognise that this was a many-to-many relationship and then resolve how the data model would be implemented.

A common error for part **(c)** was to suggest that the relationship was formed using the Patient's PATIENT ID as a foreign key in the WARD table. This incorrect response was often followed by the candidate correctly describing that "one ward would house many patients", therefore candidates needed to improve their understanding of this topic area.

Part **(d)** showed a simple DML (i.e. SQL) script for the first time and candidates had no difficulty with this. However, there were very few answers which gained the full two marks, generally due to an imprecise answer. Answers often incorrectly suggested:

- it was information about a specific single ward which would be displayed
- that the script would calculate a number of beds

or, simply expressed in words the given condition.

A concise statement such as "*a list of wards with available/empty beds"* secured the full two marks.

Question 2

This question was generally well answered with all candidates able to score. For **(b)(i)** the expression of recursion was often weak and candidates wrongly described recursion in the context of a programming language function or procedure. Candidates however, were able to recognise that rule 3 was recursive.

The approach taken to analyse the three given expressions was either a 'top down' or 'bottom up' approach and the mark scheme allowed for either. Key points looked for in the answers were:

- for "1" that the analyse used all four rules and either started or ended with rule 4
- for "001" as above and that rule 3 had to be used more than once.

Many correct answers were seen for **(c)**. The expected changes were an additional rule to define a dollar character, followed by an expansion of rule 4.

Common mistakes were to include the dollar character as an additional character in rule 1 and to omit the original expression from the amended definition of `<BinaryString>`.

Question 3

This question was generally well answered. Despite a new approach for parts **(a)**, **(b)** and **(c),** candidates were able to work out that the modes of addressing were direct, indexed and indirect. Candidates did as instructed and annotated the diagram in **(c)** to illustrate the reference first to address 203 and then back to address 200.

For the trace table in part **(d),** many answers scored the full five marks. For part **(e)** – despite the large number of mark points to secure the marks – very few candidates scored full marks. Answers require more than the general statement that each one assembly language is translated into exactly one machine code instruction. Stronger answers described the instruction being divided into the op code and operand part, and then the op code mnemonic looked up in an op-code table to find the matching machine code. For the symbol table, the labels are added to a symbol table (and then on the second pass) the assembler can calculate the actual address for each label.

Question 4

For part **(a)** answers seen here were much stronger than seen on previous papers. Many answers correctly included an explanation that the issue is about whether or not space allocation is determined at the compilation stage or can be managed at run-time. For part **(b),** most candidates scored the full three marks for drawing the four nodes of the tree and most could follow this for part **(c)(i),** by identifying one of the two procedure calls.

See the earlier comment relating to part **(c)(ii)**.

Question 5

This question proved slightly more demanding than questions from previous papers assessing the topic of compilers. Candidates understood what the keywords were, but then often omitted to state that the keyword table contained the matching token for each keyword.

Most candidates scored the second mark by stating that the symbol table would contain all the identifier names found in the source code.

For part **(b)(i)**, only a general explanation of what 'optimisation' meant was wanted and the answers expected were that the code would occupy less main memory (or candidates suggested the optimisation stage would reduce the amount of code) and that the code would execute faster.

Question 6

Batch processing was generally understood, but candidates often did not score the full three marks due to poor expression e.g. the data will be *"processed at once"* where the clear intention was to convey that all the data is *"processed at the same time'*.

Part **(a)(ii)** rarely scored the mark with the very common wrong answer 'real time'. Candidates need to develop their knowledge in this area. A simple statement that "there is continual user interaction with the software" was required. One candidate explained this with the example *"like when the user is word processing".*

For part **(b)(iii)**, a little more was expected than answers which merely suggested changing the size of the partitions. Answers expected were to suggest that not all of a program needs to be loaded at the start of execution, so a system of paging would be introduced for one or both of the partitions. Some answers suggested some form of compaction when a program terminates, which effectively always leaves one hole in the partition and so makes maximum use of the free memory.

For part **(c)**, popular answers were the operating system (or some specific modules from it e.g. the interrupt handler) and a utility such as the virus checker. Credit was not given for classification headings such as 'system software' or 'utilities'.

Part **(d)** was well answered. Candidates were familiar with the terms 'ready' and 'blocked', but often did not gain the second mark for the explanation, making an ambiguous statement which could have described either (say) a process in the blocked or ready state.

Question 7

Part **(a)** was well answered and most candidates were able to score something for **(b)(i)**. A common error was to define the array and/or `ThisValue` of type CHAR.

Some candidates did not attempt part **(ii)** with no labelling found. Stronger candidates secured the full two marks for **(iii)**. Many secured the mark for the positioning of 'LISBON' which matched with their function definition in **(i)**. The common error was a statement which was not properly formed with the `CityWasFound` variable not on the left hand side of the statement.

Question 8

Many candidates did not gain the first mark. Examples were often imprecise e.g. "the car manufacturing process"; something more specific was required such as "*for the spray painting of cars in the car manufacturing process*".

The example however, of what the candidate meant often became clear and marks were gained for parts **(a)(ii)** and part **(b)**.

For part **(c)**, most candidates correctly stated a 'real time' operating system would be required.

COMPUTING

Paper 9691/32
Written Paper

General

On topics where issues had been highlighted in previous reports there was evidence of a marked improvement in the quality of candidate answers. Where a different question framework was used to assess candidates' understanding – for example, **Question 4(c)(ii)** – candidates coped well this, with a number of candidates scoring full marks on a topic which is considered challenging. This was less so on **Question 8**, when candidates were not asked for the definition of a robot, but given the definition and then asked to apply these statements to their chosen example. The answers provided needed to show further understanding.

Question 1

Most candidates were able to score well on this question. Part **(a)** required the candidate to recognise that this was a many-to-many relationship and then resolve how the data model would be implemented.

A common error for part **(c)** was to suggest that the relationship was formed using the Patient's PATIENT ID as a foreign key in the WARD table. This incorrect response was often followed by the candidate correctly describing that "one ward would house many patients", therefore candidates needed to improve their understanding of this topic area.

Part **(d)** showed a simple DML (i.e. SQL) script for the first time and candidates had no difficulty with this. However, there were very few answers which gained the full two marks, generally due to an imprecise answer. Answers often incorrectly suggested:

- it was information about a specific single ward which would be displayed
- that the script would calculate a number of beds

or, simply expressed in words the given condition.

A concise statement such as "*a list of wards with available/empty beds"* secured the full two marks.

Question 2

This question was generally well answered with all candidates able to score. For **(b)(i)** the expression of recursion was often weak and candidates wrongly described recursion in the context of a programming language function or procedure. Candidates however, were able to recognise that rule 3 was recursive.

The approach taken to analyse the three given expressions was either a 'top down' or 'bottom up' approach and the mark scheme allowed for either. Key points looked for in the answers were:

- for "1" that the analyse used all four rules and either started or ended with rule 4
- for "001" as above and that rule 3 had to be used more than once.

Many correct answers were seen for **(c)**. The expected changes were an additional rule to define a dollar character, followed by an expansion of rule 4.

Common mistakes were to include the dollar character as an additional character in rule 1 and to omit the original expression from the amended definition of `<BinaryString>`.

Question 3

This question was generally well answered. Despite a new approach for parts **(a)**, **(b)** and **(c),** candidates were able to work out that the modes of addressing were direct, indexed and indirect. Candidates did as instructed and annotated the diagram in **(c)** to illustrate the reference first to address 203 and then back to address 200.

For the trace table in part **(d),** many answers scored the full five marks. For part **(e)** – despite the large number of mark points to secure the marks – very few candidates scored full marks. Answers require more than the general statement that each one assembly language is translated into exactly one machine code instruction. Stronger answers described the instruction being divided into the op code and operand part, and then the op code mnemonic looked up in an op-code table to find the matching machine code. For the symbol table, the labels are added to a symbol table (and then on the second pass) the assembler can calculate the actual address for each label.

Question 4

For part **(a)** answers seen here were much stronger than seen on previous papers. Many answers correctly included an explanation that the issue is about whether or not space allocation is determined at the compilation stage or can be managed at run-time. For part **(b),** most candidates scored the full three marks for drawing the four nodes of the tree and most could follow this for part **(c)(i),** by identifying one of the two procedure calls.

See the earlier comment relating to part **(c)(ii)**.

Question 5

This question proved slightly more demanding than questions from previous papers assessing the topic of compilers. Candidates understood what the keywords were, but then often omitted to state that the keyword table contained the matching token for each keyword.

Most candidates scored the second mark by stating that the symbol table would contain all the identifier names found in the source code.

For part **(b)(i)**, only a general explanation of what 'optimisation' meant was wanted and the answers expected were that the code would occupy less main memory (or candidates suggested the optimisation stage would reduce the amount of code) and that the code would execute faster.

Question 6

Batch processing was generally understood, but candidates often did not score the full three marks due to poor expression e.g. the data will be *"processed at once"* where the clear intention was to convey that all the data is *"processed at the same time'*.

Part **(a)(ii)** rarely scored the mark with the very common wrong answer 'real time'. Candidates need to develop their knowledge in this area. A simple statement that "there is continual user interaction with the software" was required. One candidate explained this with the example *"like when the user is word processing".*

For part **(b)(iii)**, a little more was expected than answers which merely suggested changing the size of the partitions. Answers expected were to suggest that not all of a program needs to be loaded at the start of execution, so a system of paging would be introduced for one or both of the partitions. Some answers suggested some form of compaction when a program terminates, which effectively always leaves one hole in the partition and so makes maximum use of the free memory.

For part **(c)**, popular answers were the operating system (or some specific modules from it e.g. the interrupt handler) and a utility such as the virus checker. Credit was not given for classification headings such as 'system software' or 'utilities'.

Part **(d)** was well answered. Candidates were familiar with the terms 'ready' and 'blocked', but often did not gain the second mark for the explanation, making an ambiguous statement which could have described either (say) a process in the blocked or ready state.

Question 7

Part **(a)** was well answered and most candidates were able to score something for **(b)(i)**. A common error was to define the array and/or `ThisValue` of type CHAR.

Some candidates did not attempt part **(ii)** with no labelling found. Stronger candidates secured the full two marks for **(iii)**. Many secured the mark for the positioning of 'LISBON' which matched with their function definition in **(i)**. The common error was a statement which was not properly formed with the `CityWasFound` variable not on the left hand side of the statement.

Question 8

Many candidates did not gain the first mark. Examples were often imprecise e.g. "the car manufacturing process"; something more specific was required such as "*for the spray painting of cars in the car manufacturing process*".

The example however, of what the candidate meant often became clear and marks were gained for parts **(a)(ii)** and part **(b)**.

For part **(c)**, most candidates correctly stated a 'real time' operating system would be required.

COMPUTING

Paper 9691/33

Written Paper

General

There was evidence from candidate responses of improvement in the standard of answers seen for some topic areas. For example, in **Question 4(d)**, where a different style of question had been used to assess candidates' understanding of what is traditionally a challenging concept – recursion, candidates were prepared and showed good understanding of the topic area. Similarly, in **Question 1(d)**, this was the first paper where candidates were presented with a data manipulation language script and required to state its purpose; again many candidates scored marks on this question.

1 This question was well answered by most candidates. Most realised that for **(a)(ii)** the solution was to include a link table. However, candidates often then did not express the key point as to how the two relationships formed would be implemented. Candidates described what the primary key of the new link table would be - which was irrelevant as to how the relationships were formed. Candidates need to develop their knowledge in this area.

For part **(d)**, see the comment in the 'General' section. However, some candidates failed to get the full two marks by suggesting that information about a single or selected ward would be displayed. Also candidates incorrectly suggested that a total number of wards would be calculated and displayed. A concise statement such as "*Displays the ward name for all wards with vacant/unoccupied/available beds*" would have secured the two marks. It is a reasonable expectation that on future papers candidates would be asked to write a DDL script for a query or some form of data maintenance task (i.e. add-delete-amend record(s)).

2 This question was generally well answered with candidates able to state that BNF is a meta-language which describes the grammar used for the expression of the syntax of a high-level language.

Candidates often did not score any marks for **(b)(i)** stating that recursion is used in procedure or function calling. The correct context was required – that it is used for rules which are defined in terms of themselves (or calls itself).

There were two possible approaches which could have been used for **(b)(iii)**. The first would start with the rule for <List> and then expand this to the given expression (i.e. a top-down approach). The alternative was to start with the given expression (i.e. a bottom-up approach). Both approaches were acceptable and the mark scheme reflected this. The key point looked for in the answer for the third expression was that rule 6 had to be used more than once.

Candidates needed to develop their knowledge for part **(c)** as few correct answers were seen. The expected solution was one additional rule and one rule change (rule 7):

<DoubleCharacter> ::= <Character> < Character>
<ListItem> ::= <Character> | <DoubleCharacter>

3 Most candidates were able to work out that **(a)** and **(b)** used indexed and indirect addressing, respectively. Part **(d)** was well answered often gaining the full four marks. For part **(e)**, despite a large number of ways in which the candidate could have gained credit, answers which scored the full three marks were rare, therefore candidates needed to further develop their understanding of this topic area as some confused 'the assembly process' with 'compilation'. The key word in the question stem was 'how', so statements such as '1-t-1 mapping' and 'translates assembly language into machine code' were insufficient. Key points expected were that the instruction is divided into its op code and operand; the op code will then be looked up in an op code table for the matching machine code. Addresses are added into a symbol table as the assembler scans the instructions in

sequence. On a second pass through the program the assembler can compute an actual or relative address and add this binary value to the object code.

4 Explanations for a static or dynamic structure were clear and answers showed a clear improvement from previous papers; candidates did now include the key point that a static structure has its size set at compile time and cannot therefore be changed at execution time.

Part **(c)** often gained full marks. See the comments in the 'General' section regarding **(d)**.

Answers for the final part **(d)(iii)** were however, often weak and did not use the correct terminology that the trace is 'unwinding' from the current procedure call, or 'backtracking' to the calling procedure.

5 Answers for **(b)(i)** often did not include that the keyword table contains the matching token used for each keyword. Descriptions for the symbol table were weak with many candidates describing a table containing character or operator symbols such as +, –, $, etc.

6 For part **(a)**, candidates recognised that application X needed batch processing. However, few answers made reference to the scenario tasks and dates to illustrate the key features of this form of processing.

Part **(b)** – despite being a new style of question – was well answered. Candidates worked out that STAFF17 could be loaded and a good explanation followed that Partition 3 would be too small to support the number of terminal users trying to log on.

For part **(iii)**, the most popular answers were the operating system, or some specific module within e.g. the various schedulers or interrupt handler. Another popular answer was virus checking software. Credit was not given for answers which gave general software classification headings such as 'system software' or 'utilities'.

For part **(iv)**, candidates often confused the term 'runnable' given in the stem of the question, with 'ready', suggesting they were terms which described a different state. Candidates need to improve their understanding of these terms.

Part **(c)** – paging – was well understood although candidates often for part **(ii)** did not give as an answer that made the most fundamental point about paging - that not all of the pages need to be present in main memory for program execution to start.

7 Part **(a)** was well answered with candidates often scoring the full four marks. A common error for **(b)** was to give a return value data type of Integer. The mark scheme was sympathetic to candidates who clearly had experience of using a variety of programming languages and had carried the data type descriptors with which they were familiar to use in their pseudocode. For example, 'float', 'single', 'real', and 'currency' would have all been acceptable to define hours worked or for the return value from CalcPay.

8 Definitions of simulation usually scored one or two marks. The answers seen for part **(b)** needed further development. Two fundamental concepts were often missing from candidate answers; that sensors act as an input device to relay data back to a computer system and that this computer system then processes – using a program – this input data.

COMPUTING

Paper 9691/04

Project

General comments

This report provides general feedback on the overall quality of project work for GCE Advanced 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 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. 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 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 Cambridge International Examinations 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 for example feasibility studies and cost benefit analysis.

Project Reports and Presentation

The presentation of most of the reports was to a very high standard, with reports word-processed and properly bound. 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 re-typed/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

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. 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. Teachers should comment as to why they awarded the marks for each section and alert Moderators to any creditworthy material that has been found in a different section or in an appendix that has not been cross-referenced by the candidate. 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

This is a brief introduction for anyone who is unfamiliar with the organisation and the area under investigation. Most candidates identified the organisation and many identified the methods used; better candidates described both the organisation and the methods used; the best candidates also included a description of the origin of the data and indicated the form of this data.

(ii) Investigation and Analysis

In order to gain good marks candidates must clearly document client and user involvement in their investigation. Candidates need to consider carefully the evidence obtained from interviews, observation of the existing system and study of documents currently in use; then ask follow up questions to fill in any gaps in the knowledge obtained about the current system or requirements needed for the new system. Alternative approaches need to be discussed in depth as they would be applied to the candidate's proposed system.

In order to be awarded marks in the top band of the marking grid candidates must provide evidence of the use of formal analysis techniques such as structure diagrams, dataflow diagrams or system flowcharts. The detailed requirements specification produced based on the information collected must include the specific client and information requirements of the system and not just concentrate on hardware and software.

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.

In order to obtain good marks for this sub-section, 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

In order to obtain good marks for this sub-section, candidates should 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 should describe the limitations of their intended system not just provide a list of general statements that could apply to any system. The estimate of the size of any files required should be based on evidence provided by their client on the number of records needed for each entity.

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, data structures used and evidence of tailoring of software packages. This should match the design specification in **(c)(i)** and be annotated by the candidate.

(ii) Programming

It is important that the programming code in this subsection is written by the candidate and not produced as a result of tailoring a software package. Marks should only be awarded to code that has been written by the candidate.

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 and ensuring that their program could be maintained by writing self-documented code.

(iii) Testing

Evidence of testing needs to be supported by a well-designed test plan that includes the identification of appropriate test data, including valid, invalid and extreme cases, together with 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, for example by production of reports specified in the design section. This omission meant candidates were unable to gain marks in the highest band for this sub-section.

(iv) Installation

Most candidates provided an implementation plan that identified the need for client/user testing, user training and system changeover.

For good marks to be awarded, written evidence from the client and/or user(s) must be included in order to show that the system has been seen and used, and the candidate's plans have been agreed.

Centres are reminded that appropriateness of structure and exploitation of available facilities are not required for this sub-section of the report.

(e) Documentation

(i) Systems Maintenance Documentation

This sub-section of the report is now a systems maintenance document. Please see page 43 of the 2013 syllabus for details of what should be included.

For top marks to be awarded the candidate must explain how adaptive maintenance could be undertaken.

(iii) User Documentation

This section was completed to a good standard by most candidates. Centres are reminded that for full marks the candidate must include an index and a glossary of the terms used in the guide. The

guide needs to be complete with 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.

(f) Evaluation

Centres are reminded, in order to gain high marks candidates need to provide a detailed evaluation that includes the content 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.

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

Candidates need to consider each objective set in their design section, **(c)(i)**, and explain how their project work met the objective or explain why it was not possible to meet that objective.

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

A response must be provided directly 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.

Centres are reminded that possible extensions and the good and bad points of their final system are not required for this sub-section of the report.