COMPUTER SCIENCE

Paper 9618/11
Theory Fundamentals

Key messages

There is a greater emphasis on the application of knowledge in this syllabus. Some questions will be set in a particular context and answers will be expected to refer to that context in order to demonstrate a candidate's application of the theory.

Many candidates would benefit from a better understanding of the different command words used in the questions. A question that begins with the command word 'describe' requires a different type of answer to one that begins with the keyword 'identify'. Simply naming, for example, when the question asks for a description, it is not enough to respond with 'validation checks'.

The questions should also be read carefully and answered appropriately. For example, if a question asks for the benefits to a programmer, no marks will be awarded for generic responses that are not of benefit to the **programmer**.

General comments

If a question asks for a given number of answers such as three reasons, only the first three reasons given in the answer will be marked. It was not unusual for candidates to include several extra answers even when the answer space was numbered appropriately.

Questions about number bases and compression were usually completed successfully. Questions about the components of the CPU and protocols were more challenging.

Comments on specific questions

These comments should be read in conjunction with the published mark scheme for this paper.

Question 1

(a) (i) This question was answered well. Most of the candidates correctly converted the unsigned binary integer into denary.

(ii) This question was also answered well. Some candidates need to understand that separators, such as commas, are not required between the denary digits after conversion.

(iii) Some candidates found converting the two's complement binary integer into denary challenging. A popular incorrect answer was 231 where the binary value had been treated as an unsigned integer. Another common error was a missing minus sign with the correct denary value.

(b) This question was not answered well. Some candidates need to understand that when the question asks for the subtraction to be completed in binary no working marks will be awarded for converting the binary numbers to denary and performing a denary subtraction. Some candidates correctly completed the subtraction in binary but showed no evidence of any working in the space provided. In order to achieve the working marks, the method of working must be clearly seen.

(c) This question was answered well. Many candidates were able to correctly give one similarity and two differences between the character sets.

(d) (i) There were many good answers to this question. Some candidates need to understand that imprecise statements such as *the sampling rate is the rate of taking samples* are not enough at this level of study.

(ii) This question asked about sampling resolution, whereas the previous question asked about sampling rate. Some candidates need to ensure that they read the question carefully. A popular incorrect answer explained the effect of increasing the sampling rate on the sound file rather than the sampling resolution.

Question 2

There were some good answers to this question. Many candidates were able to match pharming, and anti-virus software to their correct description. There was some confusion with encryption and the firewall, with quite a few responses mapping encryption to verifying the authenticity of data.

Question 3

(a) This question required careful attention to the positioning of the brackets. Some candidates found this challenging. There were several logic circuits seen with two-input NOT gates and some candidates incorrectly replaced (NOT A) AND (NOT B) with a single NAND gate. Candidates should also be aware of the need for care when drawing the shapes of the logic gates, it was sometimes very difficult to differentiate between an OR gate and an AND gate, and NOT gates were often shown without the circles.

(b) This question also needed careful attention to the position of the brackets. A common incorrect answer was the opposite of the correct solution where candidates had correctly worked out the values for the bracketed expression, then omitted to invert the values for the final NOT.

Question 4

(a) While there were some good answers to this question there were a few instances where the degree of the relationships was shown the wrong way round. Some candidates need to improve their understanding of the one-to-many and many-to-one relationships.

(b) This question was answered well. It was generic and not applied to any particular scenario. Many candidates were able to correctly describe the characteristics of a database that is in Third Normal Form (3NF).

(c) (i) There was considerable confusion between rows and columns. Frequently, the description of the tuple would state that it was a row, but the example given was a column or the description mentioned a column, but the example was correct.

(ii) The `SELECT` and `FROM` clauses were often completed correctly, but the `WHERE` clause proved to be more challenging. Many candidates need to improve their understanding of the use of wildcards.

(d) There were some good correct SQL scripts. Some candidates would benefit from more practice in setting up simple multi-table databases and using the various SQL statements in section 8.3 of the syllabus to query and maintain these databases.

Question 5

(a) Few candidates understood that the stored program concept means that instructions and data are indistinguishable, and both are stored in the same memory space.

(b) (i) The roles of the three registers given in the question were not well understood. A popular incorrect description of the Program Counter stated that it was used to count the number of instructions.

Many candidates need to improve their understanding of the roles of the registers in the CPU. Some candidates also need to understand that vague statements such as *the index register holds the index* or *the status register shows the status* without any further expansion are not sufficient at this level of study.

(ii) This part question was answered well. Many candidates were able to correctly identify the two buses used.

(iii) This question asked about the purpose of the Control Unit (CU) within the CPU. There was considerable confusion between the CU and the CPU, with many answers describing the purpose of the whole CPU rather than the purpose of the CU within the CPU.

(c) Many of the responses to this question were answers that were more appropriate for question **5(d)** rather than describing the generic purpose of an interrupt.

(d) There were many varied correct answers to this question. A popular correct answer was a runtime error, such as division by zero.

Question 6

(a) (i) Indexed addressing is not well understood. Many candidates need to improve their understanding of this. A common incorrect result of the first `LDX 110` instruction was to load 110 into the accumulator which then resulted in an incorrect value stored in memory address 101. Some candidates also need to understand that storing a value in a memory location copies the value from the accumulator and does not reset the register to zero.

(ii) This question was answered quite well. Many candidates recognised that the code was swapping the values in the memory locations.

(b) (i) This question was answered well. Many candidates were able to correctly complete the `XOR` operation.

(ii) This question was answered well. Many candidates were able to correctly complete the `AND` operation.

(iii) This question was answered well. Many candidates were able to correctly complete the `OR` operation.

(iv) This question was answered well. Many candidates were able to correctly complete the shift operation. A small number of candidates confused right and left and so shifted the bits the wrong way.

(c) There were some good answers to this question. Some candidates need to improve their understanding of the tasks carried out in each pass of a two-pass assembler.

Question 7

(a) Many of the answers to this question referred to benefits of library files in general rather than specifically to DLL files. There is also a need for care in the use of the terminology. The term ‘storage’ was often incorrectly used instead of ‘memory’. Here too there were many vague answers such as *less time consuming* or *faster*.

(b) There was confusion here between memory management and hard disk management. Many responses incorrectly described the ways that the Operating System manages the file storage on the hard disk rather than the way it manages RAM. It is very important that candidates differentiate between memory and storage.

(c) This question was answered well. Almost all candidates correctly identified that a lossless compression method would be used. The most popular lossless method described was run-length encoding (RLE).

(d) The relationship between the size of the cache memory and the performance of the CPU was not generally well understood or explained. There were many imprecise answers which simply restated the information given in the question. This is a technical subject and explanations should be given in technical terms. Some candidates need to understand that answers must include more than just general knowledge.

(e) This question was answered very well. Easily the most popular correct answers were USB and HDMI.

Question 8

Here too there was a need to demonstrate some technical knowledge and terminology. Answers incorrectly referred to data packages rather than data packets and there seemed to be a widespread misunderstanding of what is meant by a protocol. A protocol is a set of rules, and as such it cannot 'do' anything. Under the CSMA/CD protocol it is the workstations that are doing the actions, not the protocol. The protocol lays down what actions should be taken, but it is the workstations that carry out the actions such as checking if the channel is busy, sending a jamming signal, etc.

Question 9

(a) This question was one that required answers in a particular context, which in this case was a modern television. Many responses were simply generic descriptions of embedded systems. There was also some confusion between embedded systems and general control systems.

(b) There were some excellent answers to this question. Some candidates need to understand that statements such as *it is programmable electrically* which repeats the information given in the expanded acronym are not enough. The question asked for one benefit, so only the first benefit given in the answer will be marked. It was not unusual for candidates to include several extra answers.

COMPUTER SCIENCE

Paper 9618/12
Theory Fundamentals

Key messages

There is a greater emphasis on the application of knowledge in this syllabus. Some questions will be set in a particular context and answers will be expected to refer to that context in order to demonstrate a candidate's application of the theory.

Many candidates would benefit from a better understanding of the different command words used in the questions. A question that begins with the command word 'describe' requires a different type of answer to one that begins with the keyword 'identify'. Simply naming, for example, when the question asks for a description, it is not enough to respond with '*validation checks*'.

The questions should also be read carefully and answered appropriately. For example, if a question asks for the benefits to a programmer, no marks will be awarded for generic responses that are not of benefit to the **programmer**.

General comments

If a question asks for a given number of answers such as three reasons, only the first three reasons given in the answer will be marked. It was not unusual for candidates to include several extra answers even when the answer space was numbered appropriately.

Questions about the impact of Artificial Intelligence and different types of graphics were usually completed successfully, the questions about networks and databases were more challenging.

Comments on specific questions

These comments should be read in conjunction with the published mark scheme for this paper.

Question 1

(a) There were some good answers to this question. Many candidates were able to match the virus checker, backup and disk repair utility software to their correct purpose. There was some confusion between defragmentation and the disk formatter, with quite a few responses mapping defragmentation to decrease in file size.

(b) (i) This was a question that needed to be read carefully. It asked about drawbacks of a compiler compared to an interpreter during program development and so candidate answers needed to be applied to program development and involved a comparison between the two translators. Many responses simply described the operation of a compiler and/or an interpreter without any reference to program development or to any drawback of one translator compared to the other.

(ii) Many candidates found this question very challenging. There is a need for greater understanding of why some programs might be partially compiled and partially interpreted. The most common answer described the use of an interpreter first during program development followed by a compiler when the program was error free to create an executable file for distribution. This is not what the question required for a response. The question refers to a situation where code in a high-level language is compiled first to an intermediate code which can then be interpreted on several different platforms.

Question 2

(a) (i) Some candidates found converting the two's complement binary integer into denary challenging. A popular incorrect answer was 150 where the binary value had been treated as an unsigned integer. Another common error was a missing minus sign with the correct denary value.

(ii) This question was answered well. A small number of candidates introduced an error by converting the binary to denary first, then to hexadecimal, rather than converting directly from binary to hexadecimal.

(iii) In order to answer this question correctly, the unsigned binary integer given needed to be converted first to denary and then the denary number represented as Binary Coded Decimal (BCD). A very common incorrect response was 95, where candidates had assumed that the value given in the question was BCD and had converted that value to denary.

(b) This question was answered well. Almost all candidates correctly added the two binary values.

Question 3

(a) This question required careful reading and the application of knowledge to a given situation. Many candidates found this challenging. There were several logic circuits with missing NOT gates and some candidates incorrectly replaced (NOT R) OR (NOT M) with a single NOR gate. Candidates should also be aware of the need for care when drawing the shapes of the logic gates. It was often very difficult to differentiate between an OR gate and an AND gate, and many NOT gates were shown without the circles.

(b) This question was answered quite well. Some candidates need to be aware that examiners cannot see the difference between pencil and pen on scanned scripts, so if a candidate has completed the working in pencil and overwritten the answer in ink the answer needs to be clearly identified. Preferably the answer should be rewritten in the space provided so that there is no chance of a double image.

(c) This was another question that needed careful reading. It asked why ROM is used in an embedded system. Many responses described an embedded system in generic terms without including anything specific about ROM. Some candidates also need to be aware that at this level of study, it is not enough to respond with statements such as *ROM is read only*, which just repeat the expanded acronym.

Question 4

(a) While there were some good answers to this question, there was also a lot of confusion between the two terms. Some candidates need to be aware that at this level of study, it is not enough to respond with statements such as *verification verifies the data and validation validates the data*.

(b) There was some confusion with check digits and parity blocks. The question asked for a description, so responses needed to include more than just a statement about adding the data values together.

(c) This question needed careful attention to the command word. The command word was 'describe', so it was insufficient to simply list the names of two validation checks.

Question 5

(a) This question required an explanation of how the given database could be normalised. Most of the answers gave generic descriptions of Third Normal Form (3NF) without any application to the given tables. Some candidates who did apply their answers need to be careful when copying table and field names.

(b) There were some good correct SQL scripts. Some candidates would benefit from more practice in setting up simple multi-table databases and using the various SQL statements in section 8.3 of the syllabus to query and maintain these databases.

(c) This question was answered well. Most of the candidates could correctly state what was meant by a candidate key.

(d) (i) This question was also answered well. Many candidates correctly described a data dictionary and included an appropriate example.

(ii) This question was not answered well. Many candidates need to improve their understanding of a logical schema.

Question 6

(a) (i) Many candidates found this question challenging. Responses included vague references to encrypting and decrypting keys, without identifying them as private and public and with little reference to which key would be used for encryption and which for decryption. There was a lot of confusion with using a finger or stylus to sign on a touchscreen device.

(ii) The question asked 'how' a firewall protects data, so statements such as *it acts like a wall stopping malicious code* are not enough at this level of study. This is a technical subject and questions such as this require answers in technical terms. Good answers would have included statements about monitoring incoming and outgoing traffic, checking against criteria and blocking transmissions that did not meet the criteria.

(b) (i) This was answered well. Many candidates were able to correctly explain the effect of an increase in the sampling rate on the accuracy of the digitised recording.

(ii) Unlike **Question 6(b)(i)**, this question asked about the effect of decreasing the sampling resolution on the size of the file. Many candidates did not read the question carefully and assumed that this question was asking the converse of the previous question, that is, about the effect of decreasing the sampling rate on the accuracy of the recording. Answers about sampling rate and accuracy did not answer the question and so did not achieve the marks.

Question 7

(a) Indexed addressing is not generally well understood. Many candidates need to improve their understanding of this. Because the value in the index register was zero at the start of the program. A significant number of candidates were able to achieve the first two marks, but then loaded an incorrect value to the accumulator on the execution of the indexed addressing in the second iteration. Some candidates also need to understand that storing a value in a memory location copies the value from the accumulator and does not reset the register to zero.

(b) (i) This question was answered well. Many candidates were able to correctly complete the `XOR` operation.

(ii) There were several correct answers to this question. Some candidates need to improve their understanding of the difference between the `AND #n` instruction and the `AND <address>` instruction. This question required that the contents of the accumulator underwent a bitwise AND with the contents of memory address 50 (0100 1101), not with the denary value 50 (0011 0010).

(iii) This question was answered well. Many candidates were able to correctly complete the shift operation. A small number of candidates confused left and right and so shifted the bits the wrong way.

(iv) As in **Question 7(b)(ii)**, there were several correct answers to this question. However, some candidates also need to improve their understanding of the difference between the `OR #n` instruction and the `OR <address>` instruction. This question required that the contents of the accumulator underwent a bit-wise OR with the contents of memory address 51 (1000 1111), not with the denary value 51 (0011 0011).

(c) There were a number of correct answers to this question, but also some confusion with the use of the square brackets to denote the contents of a register, rather than the register itself. A popular incorrect notation was CIR ← [[MDR]] for the third answer.

Question 8

(a) Most candidates correctly identified that the first action would decrease the file size. The results of the other two actions proved to be more challenging to identify. One popular incorrect choice was mistaking screen resolution for image resolution and indicating that a change in screen resolution would result in a decrease in file size. Similarly the question stated that the colour depth was 24 bits, which would apply to both the colours mentioned in the third action yet another popular incorrect answer was indicating that a change in colour increases file size.

(b) This question was answered very well. Almost all candidates understood the differences when the two types of graphic were enlarged and also the difference in the size of the file.

(c) (i) The first line of the table had already been completed, and so it was expected that candidates used the same format in their answers. This was not always the case. Some candidates need to be careful when writing their answers. It was not unusual to see only one FF in the second answer and 80 instead of 180 in the third answer.

(ii) This question was generally answered well. A small number of candidates wrote about images and pixels rather than text files and characters.

Question 9

There were some excellent, imaginative answers to this question. Popular social implications were privacy concerns and improved security. A small number of candidates described how the software operated rather than describing the social implications of using the software.

Question 10

(a) There were many correct answers to this question. The most popular incorrect answer was to join the devices to a single main cable as in a bus topology.

(b) (i) A device was often correctly named. Some candidates did not use the diagram in **part (a)** as instructed and so named all sorts of devices not shown in the diagram above. Candidates found it more challenging to give a reason why their device would have the router attached. Simply saying that the device can connect the LAN to the Internet is not enough as this was stated in the question.

(ii) This was another question that required answers in technical terms. Candidates are expected to use the correct word ‘packet’ rather than ‘package’ or ‘data’. Many answers were superficial with statements such as *the router connects the LAN to the Internet* which was given in the question. A good description of the role of the router would include, amongst other things, statements about the receipt and forwarding of packets and the appropriate use of IP and MAC addresses.

COMPUTER SCIENCE

Paper 9618/13
Theory Fundamentals

Key messages

There is a greater emphasis on the application of knowledge in this syllabus. Some questions will be set in a particular context and answers will be expected to refer to that context in order to demonstrate a candidate's application of the theory.

Many candidates would benefit from a better understanding of the different command words used in the questions. A question that begins with the command word 'describe' requires a different type of answer to one that begins with the keyword 'identify'. Simply naming, for example, when the question asks for a description, it is not enough to respond with 'validation checks'.

The questions should also be read carefully and answered appropriately. For example, if a question asks for the benefits to a programmer, no marks will be awarded for generic responses that are not of benefit to the **programmer**.

General comments

If a question asks for a given number of answers such as three reasons, only the first three reasons given in the answer will be marked. It was not unusual for candidates to include several extra answers even when the answer space was numbered appropriately.

Questions about logical operations and conversion between different number representations were usually completed successfully, the questions about hardware components and sub-netting were more challenging.

Comments on specific questions

These comments should be read in conjunction with the published mark scheme for this paper.

Question 1

(a) This question was answered well.

(b) Many candidates understood the method for calculating the size of the file. A common error was multiplication by 50 for the sampling rate rather than 50 000.

(c) There were some excellent examples given of situations where a buffer is used. Streaming video or when printing documents were popular correct choices. Many candidates found explaining the purpose of a buffer more challenging. Explaining the purpose involves answering the question *why is a buffer used?* Many of the responses described the operation of a buffer without saying why one was needed.

Question 2

(a) This question was answered very well. Almost all candidates were able to correctly identify the relevant keys from the tables.

(b) The most popular answer here was an E-R diagram. A better, more comprehensive answer is the logical schema.

(c) The `SELECT` and `FROM` clauses were often completed correctly, but the `WHERE` clause proved to be more challenging. Many candidates need to improve their understanding of the use of wildcards. There was also some confusion between the `SUM` (totalling) and `COUNT` (counting) functions.

(d) There were a small number of good correct SQL scripts. Some candidates would benefit from more practice in setting up simple multi-table databases and using the various SQL statements in section 8.3 of the syllabus to query and maintain these databases.

(e) This question was answered well. Many candidates were able to correctly complete the paragraphs.

(f) Many candidates found this question challenging. A common answer was *the query processor allows the user to enter queries*. This is not enough at this level of study. There needs to be some expansion of what is meant by a query and how the queries are processed.

Question 3

Security management, memory management and process management were usually correctly matched to the appropriate description. There was some confusion with hardware management mapped to allocating file storage rather than the installation of drivers.

Question 4

(a) (i) There were some excellent answers for the Memory Data Register (MDR). There was significant confusion with the Memory Address Register (MAR). A significant number of responses gave the role of the Program Counter (PC) instead.

(ii) This question was answered very well. Almost all candidates could correctly identify where interrupts are detected during the F-E cycle.

(b) Many candidates found describing the purpose of the system clock challenging. Describing the purpose involves answering the question *why is a system clock needed?* Many of the responses mentioned clock speed and the effects of overclocking without saying why a system clock is needed.

(c) There were some excellent suggestions for upgrades to the hardware. The most popular were increased RAM or increasing the size of the cache memory. Some candidates found it challenging to then explain why their chosen upgrade would improve the performance of the computer system. Vague statements such as, *it makes the computer system run faster* are not enough. This is a technical subject and explanations need to include more than just general knowledge.

Question 5

(a) This question required careful attention to the positioning of the brackets. While there were some excellent solutions using NAND and NOR gates, some candidates found drawing the logic circuit for this expression challenging. There were several logic circuits without the final NOT gate. Candidates should also be aware of the need for care when drawing the shapes of the logic gates, it was sometimes very difficult to differentiate between an OR gate and an AND gate, and NOT gates were often shown without the circles.

(b) This expression was more straightforward than the one in **part (a)** and there were many correct solutions. Some candidates need to understand that care must be taken when altering binary values. If a binary digit is overwritten, the result can look like ϕ. It is not clear if this is a zero or a one. If binary digits need to be changed the original value should be crossed out completely and the new digit written alongside.

Question 6

(a) (i) The first two instructions were often completed correctly. A small number of candidates loaded the zero into the accumulator instead of the index register. Indexed addressing is not well understood. Many candidates need to improve their understanding of this. A common incorrect result of the first `LDX 110` instruction was to load 110 into the accumulator which then resulted in an incorrect subtraction. Some candidates also need to understand that the output column should not be used for the result of comparisons.

(ii) This question was not answered well. Many candidates need to improve their understanding of relative addressing.

(b) (i) This question was answered well. Many candidates were able to correctly complete the `AND` operation.

(ii) This question was answered well. Many candidates were able to correctly complete the `OR` operation.

(iii) This question was answered well. Many candidates were able to correctly complete the shift operation. A small number of candidates confused right and left and so shifted the bits the wrong way.

(c) This question was answered well. Many candidates were able to correctly name another instruction group.

Question 7

(a) Device **A** was usually correctly identified. Many of the answers for Cloud **X** and application **B** were brand names or company names rather than generic names, which cannot be credited as stated on the front cover of the question paper.

(b) There were some interesting answers to this question. The most popular advantage referred to the areas covered while the most popular disadvantage was the problem of interference.

(c) (i) Many candidates found the questions about subnetting challenging. Improved security was mentioned in a few answers, but generally there needs to be an improved understanding of this topic.

(ii) There was a small number of correct answers to this question. Many candidates need to improve their understanding of the use of subnet masks.

Question 8

(a) (i) There were some very good answers to this question. Some candidates need to understand that not all open source software is free of charge.

(ii) Many candidates were able to correctly explain that a commercial license enabled a fee to be charged for the software. Some candidates need to improve their understanding of the regulations about copying of software. Copyright does not prevent the software being copied; it makes it illegal to do so without the owner's permission.

(b) There were some interesting and imaginative answers to this question. Some candidates need to understand that the question asked for **one** economic impact, so only the first economic impact included in the answer will be marked. It was not unusual for candidates to include several.

Question 9

(a) (i) This question was answered very well. Most candidates were able to correctly convert the unsigned binary value to hexadecimal.

(ii) This question was also answered very well. Most candidates were able to correctly convert the unsigned binary value to denary.

(b) Many candidates were able to give one benefit, usually the ease of conversion between Binary Coded Decimal (BCD) and denary. Stating a second benefit proved more challenging.

Question 10

(a) Many candidates need to improve their understanding of feedback in control systems. There was considerable confusion with the sort of feedback obtained from a survey of users of the system, rather than an explanation of how system output affects subsequent system input.

(b) (i) Identification of a suitable sensor was often correct. Candidates found justifying why that sensor would be used more challenging. Statements such as, *a pressure sensor will detect any pressure on the vehicle* are too vague at this level of study.

(ii) There were some good answers to this question. Statements such as *an embedded system is one that is embedded in another system* is not sufficient at this level. A better statement is *an embedded system is one that is integrated into a larger system*.

COMPUTER SCIENCE

Paper 9618/21

Paper 2 Fundamental Problem-solving and Programming Skills

Key messages

This paper addresses the application of practical skills or 'Computational Thinking'. These skills involve analysing and understanding the requirement as well as designing and presenting a solution. Requirements are often presented using a scenario and candidates need to understand this before formulating their answer.

This subject makes use of a technical vocabulary. These specific words and phrases need to be used correctly.

Candidates should familiarise themselves with the meanings of the command words used in this paper and form their answers accordingly.

Candidates need to read each question carefully to make sure they understand what is being asked and should not assume that because a question shares key phrases with those from a past paper that the required answer is the same. An example in this paper is **Question 4(b)** which addresses the implementation of a linked list using an array. Many candidates seemed to overlook the word 'implemented' and gave an abstract description of how a linked list operates.

For guidance, an example answer for this question is given below:

Two 1D arrays are declared. One holds the data as characters, the other contains the pointers as integers. Array elements with the same index make up one node. A variable contains the index number of the first node. This is the start pointer. The pointer in the last node is set to –1. This is a null pointer and is not a valid index number.

Candidates should be encouraged to attempt all questions. Even the more advanced questions that ask for an algorithm to be written in pseudocode are at least partially accessible to most candidates.

General comments

If answers are crossed out, the new answers must be written clearly so that the text may be read easily. Many candidates make use of blank pages for rough work when preparing their final answer. In these cases, it is extremely helpful if this text is crossed out.

If typed answers are provided, then it is very helpful if these are organised so that individual answers do not span page breaks. This is particularly important for pseudocode answers. If the question involves completing a table, they typed answers should clearly indicate any unfilled rows.

The following comments relate to the use of pseudocode

Familiarity with fundamental programming concepts is vital. Lack of understanding is often illustrated by the confusion between a literal value and an identifier (such as when using a string as a filename), or the misuse of `OUTPUT` in place of `RETURN`. Many candidates appear unaware of the use of parameters, often replacing parameters to a procedure or function with a series of prompts and inputs within the body of the subroutine.

Functions and operators

The functions and operators that are available for use in pseudocode answers are described in the Insert which accompanies the paper, and candidates are strongly encouraged to refer to this information when writing their answers. Candidates should be aware that the use of language-specific functions or methods that do not appear in the Insert will not gain credit.

The concept of a function returning a value is not well understood.
As an illustration, given a function with header as shown:

```
FUNCTION Valid(InString : STRING) RETURNS BOOLEAN
```

then a valid expression using this function would be

```
IF Valid(MyString) = TRUE THEN
```

An example of invalid use seen frequently is

```
CALL Valid(Mystring)
IF RESULT = TRUE THEN
```

There is also an increasing tendency to use the complete module header when using a function or procedure, for example:

```
IF Valid(MyString : STRING) : RETURNS BOOLEAN = TRUE THEN
```

Invalid pseudocode

The following examples illustrate invalid pseudocode constructs of the kind that have been seen in candidate answers over recent series.

- A comparison statement:
  `IF x = 3 OR 4 OR 5 THEN`
- A count-controlled loop:
  `FOR Index ← 1 TO EOF(Filename)`
- A comparison:
  `IF Num ← 43 THEN`
- Use of the logical operator `AND` as a continuation symbol:
  `OUTPUT Staff(Index).EmployeeNumber AND Index ← Index + 1`
- Count-controlled loops where the assignment operator is omitted:
  ```
  FOR X 1 TO 100
  FOR Y TO 50
  ```

Comments on specific questions

Question 1

(a) Many candidates gained at least some credit. The most common correct points given were that they are tried and tested so are free from errors and that they perform a function that you may not be able to program yourself. Vague answers such as 'less mistakes can occur' were commonplace and these did not gain credit.

Although the question specified a single programmer, many answers related to programming teams.

(b) (i) Many candidates gained a mark for an answer along the lines of making the program easier to design/code/test/understand.

A common mistake was to refer to the actual inputting of data by the user rather than the underlying program, and answers such as 'easier to input the data' did not gain credit. As in the previous part, many vague answers were seen, especially those relating to space-saving.

(ii) The majority of answers were fully correct. Many answers referred to the use of the variable as an index or gave a clear description of how it was used to identify an individual element (a candidate or mark). Some vague answers suggested that the variable should be used to 'give a value', which did not gain credit.

(c) Many fully correct answers were seen, with most candidates gaining at least some credit.

Rows 2 and 4 were straightforward and were normally the ones where credit was given. Some answers for row 4 were not sufficiently precise, such as stating that there 'should only be one parameter'. In the case of row 1, some candidates found it more challenging to describe the error.

Row 3 was correctly identified as 'NO ERROR' by many candidates, although a few answers suggested that the digit 5 should have been inside the brackets.

Candidates should note that the question asks to 'State the error' and not to provide an attempt at a corrected statement.

Question 2

(a) Some very good explanations of abstraction were seen. Many candidates did not give an acceptable description. Many confused the processes of abstraction and decomposition and some referred to hiding data from the user (some suggesting user-permissions). Several answers related to a later stage in the development cycle and referred to removing data from an already written program.

The items to be stored needed to relate to an individual car hire event. Items that could be inferred from another item were not generally given credit. For example, if a Customer ID was stored then it would not also be necessary to store the customer name and telephone number.

(b) A very small number of good answers were seen, suggesting operations relevant to the given scenario. In general, however, this was not well answered. Few candidates seemed to understand the meaning of the term 'operation' in this context. Many suggested pseudocode operators such as `MOD` or `DIV`. Many answers referenced generic operations such as 'storage' or 'validation' without linking these to the scenario.

Question 3

A small number of very good answers were seen, giving steps that could easily be mapped to a program. Many answers did not give a viable description of the required steps for this simple algorithm. This suggests that candidates have had little experience in describing an algorithm. Steps were often vague and frequently simply repeated phrases from the question.

Common mistakes included:

- inputting all 200 values
- referring to a value to store the biggest value but initialising it to something other than the first array element (or not at all)
- assuming that the algorithm should search for a specific value that would be given at the start
- swapping adjacent array element values
- referring to a loop but not specifying the number of iterations
- vague comparisons between the current element, the next element and some maximum value.

Question 4

(a) (i) Many fully correct answers were seen, often with very clear diagrams that simply added the two data values in two new nodes, linked appropriately, and with dotted lines used to show links from the original linked list that had been broken (although this latter point was not a requirement). Some

answers appeared to show multiple lists. Sometimes it was clear that these were intended to represent intermediate steps, but this was not asked for.

Some diagrams were confused. Common mistakes included:

- allocating data values 'A' and 'K' to pointer cells (usually value 'A' as the Start pointer)
- leaving node 'J' in the list
- not assigning a null pointer to node 'K' or pointing node 'K' to node 'L'.

(ii) A very small number of good answers were seen, usually referring to the need to just change the pointers.

This was not well answered by most, with few candidates gaining credit. Many confused answers were seen, such as the suggestion that 'in a linked list data is stored permanently' indicating that candidates had little understanding of the question. Many answers did not make the connection with the task from part **(a)(i),** as directed in the question, and offered very generic answers.

(iii) Comments for the previous part apply equally here. The very small number of correct answers tended to address the additional complexity of the linked list, although reference to the need to store pointers (in addition to storing data) was seen occasionally.

(b) A small number of good answers were seen. As mentioned above, many answers did not address how the linked list could be implemented and instead gave a description of abstract linked list operations.

Question 5

A small number of very good solutions were seen.

There were many poor answers, with candidates making little or no viable attempt at a solution. The majority of candidates gained little credit.

Often the declaration of variables for array indices was omitted. In many cases the error was more fundamental, such as lack of an `ENDPROCEDURE` statement or the passing of unrequired parameters. The passing of both arrays was permitted, but the declaration of `Array1` and `Array2` as local variables was not.

Most solutions contained a count-controlled loop. Some good solutions were seen where the loop ranged from 1 to 200 (for `Array2`) and within the loop the three corresponding element values from `Array1` were extracted and processed. Solutions where the loop counter was used as an index to `Array1` often ran into a problem where the loop range was given as 1 to 600 and so within the loop an attempt would have been made to access elements with index values over 600.

Several solutions incorporated two loops: an external loop from 1 to 600 and an internal loop from 1 to 200. In these cases, each calculated value would be written to every element within `Array2`.

An occasional error involved dividing just the third value by 3, rather than summing three values and dividing the total by 3. Weaker solutions often calculated the average of the three index values rather than the average of the elements.

It was rare for the result to be converted to an integer. This was sometimes addressed by use of the `INT()` function, and correct use of the `DIV` operator was also seen.

Question 6

(a) There were many correct answers, which gave an out-of-range value for one or more of the variables. There were also many answers that gave in-range values for each of the variables. (In these cases, the bracket contents would have evaluated to `TRUE` and so the condition, including the `NOT` would have evaluated to `FALSE`).

Common mistakes included simply repeating the line of pseudocode or suggested a value for the `InString` parameter rather than the variables given in the line of pseudocode. Some answers gave single values without indicating which variable they related to.

(b) (i) This was well answered by most candidates.

Common mistakes included:

- value of `Index` extending beyond 12
- initialisation values not shown
- value of `Others` extending beyond 9.

(ii) This was well answered by most candidates. A small number made no attempt at this question.

Question 7

(a) (i) This was an algorithm in two parts: first the expression had to be split into three parts and then these needed to be evaluated. A small number of excellent solutions were seen.

Many candidates made little or no attempt at a solution. A significant number of solutions attempted to `INPUT` the three parts of the expression.

Many candidates recognised the need to store the three parts of the expression. In some solutions the variable declarations were placed before the procedure header. These were not then treated as local variables.

Most solutions attempted a loop to go through the string character by character. Commonly these were count-controlled based on the length of the string, but conditional loops which terminated when an operator character were also seen.

Many solutions included the correct use of the `MID()` function.

A mistake seen often was the attempt to extract a single character by treating the string as an array. Another common mistake when testing an individual character was the use of the following conditional test:

```
IF NextChar = '+' OR '-' OR '*' OR '/' THEN
```

Having located the operator character, many solutions attempted to extract the two values using substring functions. The detail of the substring operation (either the start or the length of the substring) was often not correct.

The use of `STR_TO_NUM()` featured in many of the more accomplished solutions. Confusion was often present in weaker solutions regarding the use of variables of type Integer or String. Often attempts were made to add single characters to variables of type Integer.

A common mistake was for the variable used to not have been assigned a value. Where the result of the calculation was assigned to a variable, this had to have been declared of type `REAL` to accommodate the result of the division operation.

A common mistake was to attempt to evaluate the expression by simply combining the three parts of the expression. For example:

```
Result ← Value 1 Operator Value2
```

(ii) This part was not well answered.

Common mistakes included:

- passing more than one parameter or none
- returning a value of a type other than `REAL`.

(b) This part was not well answered.

A small number of candidates identified the potential for a runtime error being generated by an attempted divide by zero. A few answers referred to the possibility of an overflow and this was also given credit. Many candidates suggested incorrectly formatted strings, contrary to the wording of the question.

Question 8

(a) Several excellent solutions were seen. There were also some answers where candidates made little or no attempt at a solution.

Candidates should be encouraged to declare local variables immediately below the header, rather that haphazardly throughout the body of the module.

The following mistakes were common:

- placing quotation marks around the parameter name
- not specifying the file name when attempting to close it.

Another error involved solutions that immediately returned `TRUE` as soon as 10 non-blank lines had been read, which would skip the required `CLOSEFILE`.

Many candidates correctly identified the need for a loop. A common mistake was to omit the filename from the `EOF()` function.

Several solutions merely counted all the lines in the file without checking their contents. Some solutions did not read a line.

A common mistake was to count the number of blank lines rather than count number of non-blank lines. A small number of candidates chose to count both and then perform a calculation before the final value was returned. This was over-complicated and tended not to work correctly.

Very few solutions terminated the loop when 10 non-blank lines had been read.

If the correct count operation had been implemented, then successful Boolean return usually followed.

(b) This was not well answered.

Many answers bore little resemblance to the required pseudocode statement. Where a procedure call was recognisable, a common mistake was to omit the quotation marks around the file name.

(c) Just a small number of good solutions were seen. Most candidates made little or no viable attempt.

Only a very small number of answers correctly made use of the functions `Checkfile()` and `CheckLIne()`.

As mentioned earlier, the use of functions is misunderstood by most candidates and the use of `CheckFile()` was rarely correct. The following example illustrates a typical error:

```
CALL CheckFile()
IF Result = TRUE THEN
```

Solutions that correctly made use of `CheckFile()` often did not end if the value returned was `FALSE`. Those that did gain credit here usually contained the remainder of the pseudocode within an appropriate `ELSE` clause.

Better solutions often contained a correct attempt at a conditional loop until `EOF()`.
`ErrCount > MaxErrors` was usually not addressed. If the conditional loop was addressed correctly then there usually followed an attempt to read a line from the file, but the attempt to pass this as a parameter to `CheckLine()` was less successful than the read operation itself. Some solutions attempted to treat the file as an array.

The same comments that relate to the use of `CheckFile()` apply equally to the use of `CheckLine()`. Additionally, in many cases the parameter to `CheckLine()` was the file name itself rather than a line read from the file.

The decision as to which message to output often compared a count with the literal value 20 rather than with the value of the supplied parameter. Several solutions included the output statements inside the loop.

(d) A small number of correct answers were seen, usually referring to errors relating to block structures or the use of an invalid data type. Many candidates made no attempt. Contrary to the question, many answers referred to logic or run-time errors

COMPUTER SCIENCE

Paper 9618/22

Paper 2 Fundamental Problem-solving and Programming Skills

Key messages

This paper addresses the application of practical skills or 'Computational Thinking'. These skills involve analysing and understanding the requirement as well as designing and presenting a solution. Requirements are often presented using a scenario and candidates need to understand this before formulating their answer.

This subject makes use of a technical vocabulary. These specific words and phrases need to be used correctly.

Candidates should familiarise themselves with the meanings of the command words used in this paper and form their answers accordingly.

Candidates need to read each question carefully to make sure they understand what is being asked and should not assume that because a question shares key phrases with those from a past paper that the required answer is the same. An example in this paper is **Question 3(b)** which addresses the implementation of a stack using an array. Many candidates seemed to overlook the word 'implemented' and gave an abstract description of how a stack operates.

For guidance, an example answer for this question is given below:

A 1D array is declared to store the string data for the stack. There needs to be enough elements in the array to hold the complete stack. Each value on the stack will be stored as one element in the array. An integer is declared as the stack pointer. The stack pointer contains the array index. It is initialized to –1 to indicate that the stack is empty. When an item is pushed onto the stack it is stored at the location given in the stack pointer and the stack pointer is incremented. When an item is popped off the stack the stack pointer is first decremented and the array element at this location is returned.

Question 6(b) highlighted a lack of understanding concerning what could be returned by an INPUT statement. For example, many candidates attempted to differentiate between the input of 12 'as a number' compared to 'as a string'.

Candidates should be encouraged to attempt all questions. Even the more advanced questions that ask for an algorithm to be written in pseudocode are at least partially accessible to most candidates.

General comments

If answers are crossed out, the new answers must be written clearly so that the text may be read easily. Many candidates make use of blank pages for rough work when preparing their final answer. In these cases, it is extremely helpful if this text is crossed out.

If typed answers are provided, then it is very helpful if these are organised so that individual answers do not span page breaks. This is particularly important for pseudocode answers. If the question involves completing a table, they typed answers should clearly indicate any unfilled rows.

The following comments relate to the use of pseudocode

Familiarity with fundamental programming concepts is vital. Lack of understanding is often illustrated by the confusion between a literal value and an identifier (such as when using a string as a filename), or the misuse of `OUTPUT` in place of `RETURN`. Many candidates appear unaware of the use of parameters, often replacing parameters to a procedure or function with a series of prompts and inputs within the body of the subroutine.

Functions and operators

The functions and operators that are available for use in pseudocode answers are described in the Insert which accompanies the paper, and candidates are strongly encouraged to refer to this information when writing their answers. Candidates should be aware that the use of language-specific functions or methods that do not appear in the Insert will not gain credit.

The concept of a function returning a value is not well understood.
As an illustration, given a function with header as shown:

```
FUNCTION Valid(InString : STRING) RETURNS BOOLEAN
```

then a valid expression using this function would be

```
IF Valid(MyString) = TRUE THEN
```

An example of invalid use seen frequently is

```
CALL Valid(Mystring)
IF RESULT = TRUE THEN
```

There is also an increasing tendency to use the complete module header when using a function or procedure, for example:

```
IF Valid(MyString : STRING) : RETURNS BOOLEAN = TRUE THEN
```

Invalid pseudocode

The following examples illustrate invalid pseudocode constructs of the kind that have been seen in candidate answers over recent series.

- A comparison statement:
  `IF x = 3 OR 4 OR 5 THEN`
- A count-controlled loop:
  `FOR Index ← 1 TO EOF(Filename)`
- A comparison:
  `IF Num ← 43 THEN`
- Use of the logical operator `AND` as a continuation symbol:
  `OUTPUT Staff(Index).EmployeeNumber AND Index ← Index + 1`
- Count-controlled loops where the assignment operator is omitted:
  `FOR X 1 TO 100`
  `FOR Y TO 50`

<u>Comments on specific questions</u>

Question 1

(a) (i) Many candidates gained some credit, usually for reference to reusability. Credit was also often given for stating that the use of a subroutine could make one or more development stages easier. Reference to a part of the algorithm 'performing a specific task' was rarely seen. Candidates often seemed to misunderstand the question and gave answers that related to the use of library routines. Where these were specific to library routines, no credit was given.

(ii) Many fully correct answers were seen. The most frequent incorrect answer given for the term used was 'variable'. Incorrect answers for the use of the identifiers often referred to the values being INPUT into the procedure and there were many vague references to values being used 'within the program' but not specifically within the procedure.

(b) Around half of the answers seen correctly identified beta testing. Of these, many gave good descriptions. Answers that simply referred to the program being 'tested', were not sufficient, as this was given in the question. Incorrect answers were normally based around some other form of testing. These sometimes were given credit for an appropriate description.

Many candidates made no attempt at this question.

(c) A range of performance was seen, with rows 2 and 3 usually the most successfully answered.

The most common mistake was the omission of quotation marks around the D, which are needed to indicate it is a character (either single or double quotes were acceptable).

Question 2

(a) (i) A small number of very good answers were seen, giving steps that could have been easily mapped to a program.

Many answers did not give a viable description of the required steps for this simple algorithm. This suggests that candidates have had little experience in describing an algorithm. Steps were often vague and frequently simply repeated phrases from the question.

Common mistakes included:

- inputting all 35 names and marks at the start without mention of how the data would be stored
- omitting the prompt
- not specifying the file mode
- referring to a loop but not specifying the number of iterations
- repeatedly opening the file within the loop
- a vague comparison test, often relating to multiple values (e.g. 'find all marks less than 20')
- writing the mark as well as the name to the file.

(ii) Around half of the candidates could explain that file storage was non-volatile.

Very few gave the alternative scenario-specific explanation: that the data would not need to be re-entered the next time the program is run. A number of answers suggested that data would be 'easier to process' when stored in a file, which did not gain credit.

(iii) Fewer correct answers were seen for this than for the previous part. Some answers referred to `WRITE` mode being used to create a new file but did not make it clear that existing data would be over-written.

Many candidates made no attempt at this question.

(b) This part was attempted by virtually all candidates and was very well answered.

The question did not explicitly state that rows followed in sequence (i.e. the 'Next state' on one row was assumed to be the start state for the next) and allowance was made for this.

Question 3

(a) This was well answered by the majority.

Those who identified a queue usually gained further credit for referring to the FIFO operation and many of these were also credited for mentioning the use of pointers. Reference to a circular queue was rarely seen.

Those who identified a linked list also tended to gain credit for mentioning the use of a start pointer. Further credit was given less frequently compared to answers describing a queue, suggesting that perhaps a linked list is less well understood.

ADT terminology was often confused, for example with the description of push and pop operations being applied to a queue.

(b) A small number of good answers were seen.

As mentioned earlier, many answers did not address how the stack could be implemented and instead gave descriptions of abstract stack operations.

(c) This was well answered by most candidates, with many being fully correct. Responses to the question were either very good or very poor. A small number of candidates provided no meaningful answer and several made no attempt. A common mistake was to omit the stack pointer after each group of operations, despite this being emphasised in the question.

Question 4

(a) This was a straightforward translation from a program flowchart to pseudocode and was well answered by most candidates.

Most candidates identified the loop, although some omitted the increment of `Index` or included this in only one branch of the enclosed conditional clause. Where a loop was omitted, the tendency was to replace it with a conditional clause (IF Index < 7).

Other common mistakes included:

- not inputting the `UserID` before passing it as a parameter
- incorrect use of the `GetAverage()` function
- missing `ENDIF`
- replacing the final call to `Update()` with an `OUTPUT` statement.

(b) Relatively few candidates correctly identified one of the two possible answers. Many answers seen suggested that the question had not been understood. Many candidates made no attempt at this question.

Question 5

Responses to this question were either very good or very poor. Only a very small number of fully correct answers were seen.

The condition to call `Sub1()` was the one most usually given, and in many of these cases the condition involving `Sub2()` was also given. Correct statements involving `Sub3()` and `Sub4()` were much less frequent. Many solutions included a reference to identifier `B` or relied purely on the value of identifier `C`.

Many candidates did not follow the instruction to give four separate `IF..THEN...ENDIF` statements. Conditional statements with `ELSE` clauses were seen regularly, and `CASE` statements were attempted by a few.

A number of answers included the use of a comma separator, which was not accepted for pseudocode. For example:

```
IF A, B, C = TRUE THEN
```

Question 6

(a) A small number of very good solutions were seen. There were many cases where candidates made little or no viable attempt at a solution, and the majority of candidates gained little credit.

Most candidates recognised the requirement for a loop and many of these included an attempt at repeated multiplication (or division, as appropriate) as the algorithm attempted to iterate towards the given parameter value.

The decision as to whether the parameter value had been reached was often attempted in some form. The use of `MOD` and `DIV` operators was seen regularly and often these were used correctly in the given algorithm.

(b) Some very good answers were seen, addressing input strings that were invalid for different reasons. Many candidates gave several strings which were all invalid in the same way (e.g. all non-numeric) and valid input strings were often given.

As mentioned earlier, many answers indicated confusion regarding what could be expected from user input. In addition to '12 as an integer' and '12 as a string', several candidates suggested that Boolean values could be directly input. Some suggested standard pseudocode functions as input.

In many cases the 'Reason for choice' did not match the given Input.

Question 7

(a) This part was essentially just a linear search with a slight twist. The question was not generally well answered. Several candidates made no attempt at this question.

Many weaker solutions only managed the procedure heading and ending (including parameters) . A common mistake was to omit the `ENDPROCEDURE` statement.

Many loops were count-controlled rather than conditional. Credit was available for a count-controlled loop but in many cases the given loop ran from 1 to 800 (the range of error numbers) rather than from 1 to 500 (the bounds of the array) and so this was not given. An occasional error seen was the attempt to use the string function `LENGTH()` to return the number of elements in the array.

Where conditional loops were used, the terminating condition often included an attempt at checking after element 500 had been tested and if found, to output message 1. Where both conditions were tested, these were usually combined correctly in the case of a pre-condition loop, but often the logical operator which combined the two tests was incorrect in the case of post-condition loops.

An attempt to test whether the current element matched the supplied error number was often incorrect. In several cases, the error number was compared with the array index rather than the array element. The incorrect use of the assignment operator '←' in a comparison expression was commonplace.

Those who attempted to form an output string using the concatenation operator '&' often did not first convert the line number to a string. Those that chose to output a comma-separated list fared better in this respect.

Often the output statement was placed inside the loop so an output would have been generated for each non-matched error number.

(b) This was a standard bubble sort, with an added requirement of needing to apply the swap operation to both arrays. Several excellent solutions were seen. There were also many answers where candidates made little or no attempt at a solution.

Credit was given to most viable attempts at the procedure heading and ending.

In many cases the conditional loop was count-controlled so the potential for an efficient solution was reduced.

Many weaker solutions lacked an inner loop. Credit was given to many viable solutions for the correct range, but in several cases the upper bound would have resulted in the element comparison attempting to use an index that was greater than the upper bound of the array.

The task required elements from both arrays to be swapped, which in turn required two temporary variables of different data types. Credit for this was only given in a small number of comprehensive solutions.

(c) (i) This part was well answered. Most answers referred to the fact that `ErrCode` should be of type integer.

(ii) This part was not well answered. The benefit most often credited was that the program was easier to design, write, test etc. Other benefits were only occasionally mentioned.

(iii) Quite a few candidates were unsuccessful at this part. Common mistakes included incorrect array upper bound (800 instead of 500) and use of an incorrect data type rather than `ErrorRec`.

COMPUTER SCIENCE

Paper 9618/23

Paper 2 Fundamental Problem-solving and Programming Skills

Key messages

This paper addresses the application of practical skills or 'Computational Thinking'. These skills involve analysing and understanding the requirement as well as designing and presenting a solution. Requirements are often presented through the use of a scenario and candidates need to understand this before formulating their answer.

This subject makes use of a technical vocabulary. These specific words and phrases need to be used correctly.

Candidates should familiarise themselves with the meanings of the command words used in this paper and form their answers accordingly.

Candidates need to read each question carefully to make sure they understand what is being asked and should not assume that because a question shares key phrases with those from a past paper that the required answer is the same. An example in this paper is **Question 4(b)(ii)** which requires a description of the declaration and initialisation of the variables and data structures used to implement a queue using an array. Many candidates seemed to overlook the word 'implement' and gave an abstract description of how a queue operates.

For guidance, an example answer for this question is given below:

An array needs to be declared which will be able to store at least 10 characters. Two integer variables are needed to store the front and rear pointers for the queue. Both of these need to be initialised to –1 to represent an empty queue. An additional integer variable can be used to store the number of items in a queue which should be given a value of 0 when declared.

Candidates should be encouraged to attempt all questions. Even the more advanced questions that ask for an algorithm to be written in pseudocode are at least partially accessible to most candidates.

General comments

If answers are crossed out, the new answers must be written clearly so that the text may be read easily. Many candidates make use of blank pages for rough work when preparing their final answer instead of an additional answer booklet. It is recommended that additional answer booklets are used, not blank pages.

The following comments relate to the use of pseudocode

Familiarity with fundamental programming concepts is vital. Lack of understanding is often illustrated by the misuse of `OUTPUT` in place of `RETURN`. Many candidates also appear unaware of the use of parameters, often replacing parameters to a procedure or function with a series of prompts and inputs within the body of the subroutine.

Functions and operators

The functions and operators that are available for use in pseudocode answers are described in the Insert which accompanies the paper and candidates are strongly encouraged to refer to this information when

writing their answers. Candidates should be aware that the use of language-specific functions or methods that do not appear in the Insert will not gain credit.

Invalid pseudocode

The following examples illustrate invalid pseudocode constructs of the kind that have been seen in candidate answers over recent series.

- A comparison statement:

```
IF Actual > Max + 2 OR < Min - 2 THEN
```

instead of

```
IF Actual > Max + 2 OR Actual < Min - 2 THEN
```

- A comparison statement:

```
IF Num ← 43 THEN
```

instead of

```
IF Num = 43 THEN
```

- Count-controlled loops where the assignment operator is omitted:

```
FOR Index 1 TO 500
```

instead of

```
FOR Index ← 1 TO 500
```

Comments on specific questions

Question 1

(a) The vast majority of answers were completely correct.

(b) (i) The most common correct answer was simply a statement regarding planning e.g. 'Planning the website is easier'. About half the candidates were successful with this part. Sometimes the answer was too vague or just a reuse of the reference to steps given in the question.

(ii) There are many possible correct answers to this question; the most common was 'requirement specification'. A substantial number of candidates did not name a correct document. Examples of incorrect documents include 'initial plan', 'flowchart' and 'trace table'. Some other answers were often too vague.

(c) (i) Many answers were too vague to gain credit. The most common correct answer mentioned development of prototypes. Few answers mentioned the importance of client involvement.

(ii) There were quite a few completely correct answers to this part. The most common correct answers relating to benefits were 'quicker development' and 'the use of prototypes'. The drawbacks of RAD were less well known than the benefits.

(d) Many candidates gained at least partial credit for this part.

Question 2

A majority of candidates were fully successful in this part. The most common modules correctly identified included those that enabled a charity to be selected or dealt with making a payment.

Question 3

This year a distinct improvement in describing an algorithm to solve a straightforward problem was noted. Most candidates gained some credit and just under half received full credit for their answer. A common omission from the answers given was not declaring and initialising the two sum variables.

Question 4

(a) A majority of candidates were successful in this part. A common error was to give an actual data type such as 'string' as the answer.

(b) (i) Nearly half of candidates were fully successful in this part. Quite a few candidates made no attempt. The syllabus, with regard to queues (Section 10.4), specifically mentions that candidates should be able to add, edit and delete data from these structures. Of those candidates who correctly answered this question, a significant number used the terms 'enqueue' and 'dequeue' correctly.

(ii) A substantial minority of candidates gained full credit for this question. Quite a few made no attempt. As mentioned in the Key messages section of this report, many candidates seemed to overlook the word 'implement' and gave an abstract description of how a queue operates rather than how an array could be used to implement a queue.

Question 5

(a) (i) Almost half of the candidates were fully successful in this part, which is a noted improvement on similar pseudocode questions in previous years. Although many candidates correctly passed four parameters of the required type, most did not use ByRef for the values that had to be returned. Most candidates correctly used string handling functions correctly for at least one of the data items that had to be extracted.

(ii) Only a small minority of candidates were fully successful in this part, with quite a few making no attempt. The syllabus states that candidates should be able to use the terminology associated with procedures, and in the notes and guidance section it specifically includes the procedure interface. An answer that would have gained full marks is 'The procedure interface provides a mechanism to allow the calling program to pass data that defines the four parameters used in Unpack()'.

(b)(i) Just under half of the candidates provided the correct answer to this question. The correct answer was `LineData.Cost ← 12.99`

Some incorrect answers showed a misunderstanding of how the fields of a record can be accessed such as:

```
Cost(LineDate) ← 12.99
LineData:Cost ← 12.99
StockItem.Cost ← 12.99
```

(ii) Most candidates attempted this part, but only a small minority of candidates were fully successful. A good answer would be: 'The new function will return an item of type StockItem and a local variable of type StockItem needs to be declared and used'.

(c) Over half the candidates were fully successful in their answers.

(d) Most candidates gained some credit for this part, with many giving fully accurate answers.

Candidates sometimes provided vague answers such as 'The walkthrough method is used to check for errors'. To gain credit, answers need to be more specific and detail how the errors are actually identified.

Question 6

(a) Over half of candidates were fully successful at this part.

Common errors were to give max and min values that were not at least 5 grams apart or to give a component weight of 0 grams.

(b) Again, over half of candidates were fully successful at this part. This is a noted improvement for this type of pseudocode question over previous years.

Most candidates gained credit for the following:

- function heading and ending including parameters
- declaration of local variable for Result.

A varied range of correct solutions were seen with a few not actually using local variables. Credit could still be given for the second bullet point above, provided the approach used was an appropriate alternative.

Question 7

(a) Less than half of candidates were fully successful in this part, although most candidates gained some credit. The solution required the use of a loop with the better solutions using a WHILE loop. Some good attempts at solving the problem were also made using a FOR loop.

Credit was most often given for:

- declaration and initialisation of local integer for counting the error numbers found
- appropriate prompt and two inputs
- loop to end of the array.

It should be noted that the first of these was often spoilt because the variable was not initialised, and the second because no appropriate prompt was used.

Sometimes candidates attempted to form an output string using the concatenation operator '&' but did not first convert the count or the line numbers to a string**.**

Few candidates gained full credit for the conditional loop.

(b) (i) A large minority of candidates were fully successful in this part. Many candidates gained at least some credit.

Candidates using a FOR loop could not be awarded credit for the requirement of a 'Conditional loop terminating when item added **OR** end of array reached' unless an immediate return was used.

Credit was most commonly given for:

- test for unused element in a loop
- assignment of values to arrays//save index of first blank location and assign after loop.

Most candidates did not address the need to 'Call `SortArrays()`' although the need to keep the `ErrCode` array in ascending order was a requirement. Some candidates correctly wrote their own sort code and were given credit.

Many candidates counted the used elements, rather than the unused elements as asked for in the question.

(ii) A minority of candidates were fully successful in this part. Many answers were too vague and did not describe the steps of the algorithm required.

Common errors were:

- not specifically mentioning how many times the loop required would iterate or what would cause the loop to end
- simply writing ‘ErrCode would be searched for in the array’ without describing how this would be achieved.

COMPUTER SCIENCE

Paper 9618/31
Advanced Theory

Key messages

Candidates need to demonstrate a detailed study of the topics covered by the syllabus, ensuring they make good use of technical terminology as appropriate for this advanced theory paper. Candidates who have studied the relevant theory, and who have also practiced and made use of the relevant tools and techniques, are more likely to be able to solve the problems set on the examination paper.

Candidates are advised to answer each question in an appropriate manner for the command word of the question; for example, a question beginning with 'explain' requires more detail than a question beginning with 'identify'. If a question asks for working to be shown, candidates must also ensure that they do this, in order to gain full credit.

Candidates must always make sure that they answer questions in the context of any scenario described in the question, rather than in generic terms.

Candidates are further advised to make use of the published pseudocode guide when preparing for this examination, for example in the areas of user-defined data types or algorithm construction, and answer questions requiring pseudocode answers using this syntax.

General comments

Candidates should make sure that any response they give does not simply restate the question, for example **Question 11(a)** requires a definition of a number of Object-Oriented Programming terms, including inheritance. Candidates who simply used the word inherit in their response without clearly stating what this means in this context have not demonstrated that they understand the meaning of the term.

Candidates need to read questions carefully before beginning their answer, for example in **Question 12(b)(ii)** candidates are asked to describe the meaning of O(log n) as it applies to a binary search algorithm, which means that candidates were only required to give a description of this particular Big O notation in relation to binary searches. Some candidates compared the binary search algorithm with the linear search algorithm instead.

Comments on specific questions

Question 1

(a) Many candidates achieved full credit by converting the given binary floating-point number into denary and showing their working. Some candidates did not correctly demonstrate how their exponent was calculated and how this was applied to the mantissa to find the final answer. Some candidates achieved some credit for their working where they demonstrated correct methods with the given data.

(b) Many candidates correctly demonstrated the conversion of a negative denary number to two's complement notation. A high proportion of candidates recognised the link between this question part and part **(a)** as being the negative version of the number, so that the exponent would be the same and only the mantissa would change.

(c) (i) The majority of candidates correctly stated a reason why the given binary number was not normalised, with a range of answers, including the fact that the two most significant bits of the mantissa were the same digit.

(ii) The vast majority of candidates achieved at least some credit, which was most likely for correctly stating the mantissa in its normalised form. A smaller number of candidates gave the correct exponent.

Question 2

A wide range of quality of responses was seen for this question, with a significant number of candidates demonstrating their knowledge of the functions of the Transport layer and/or the Internet layer of the TCP/IP protocol suite. Some candidates gave answers that were not specific enough; for example, for the internet layer, 'source and destination IP addresses are added to the packet header' is acceptable, whereas 'it uses IP addresses' is not enough.

Question 3

(a) The vast majority of candidates identified that an enumerated data type is a non-composite data type, but they also needed to identify the unique aspect of this data type, i.e., that it includes an ordered list of possible values.

(b) A high proportion of candidates stated the fact that the pointer non-composite data type is used to reference a memory location.

(c) Most candidates achieved at least partial credit for declaring the required data type. Some candidates used an incorrect pseudocode keyword, such as `DECLARE`, where they should have used `TYPE`.

(d) (i) This was a generally well answered question. Errors included the use of `Pet` as the variable name rather than as the data type as stated in the question, or not using the correct pseudocode keyword, `DECLARE`.

(ii) Many candidates performed well on this part. Errors included candidates adding additional pseudocode keywords that were not required, such as `DECLARE,` candidates not using the given field names in their responses, or candidates assigning data of the wrong data type, such as strings or characters without quotation marks.

Question 4

This was a generally well answered question with the vast majority of candidates able to match some or all of the given stages of compilation with their correct descriptions.

Question 5

(a) Some candidates were able to give the correct Reverse Polish Notation (RPN) for the given infix expression.

(b) (i) A high proportion of candidates were able to state the correct infix version of the given RPN expression.

(ii) The vast majority of candidates successfully evaluated their infix expressions from **part (b)(i)** by applying the given values for the variables a, b, c and d.

Question 6

(a) The vast majority of candidates gained at least partial credit for stating that a private key is unpublished, that it has a matching public key or that it is used to decrypt data that was encrypted with its matching public key.

(b) Candidates generally understood the process of asymmetric encryption as a process where one key is used to encrypt the message and another key is used to decrypt it. Sometimes candidates

did not clearly state which key was used to encrypt and/or decrypt the message, or if they did, it was not explicitly clear that the two keys were linked. For example, a good answer would be 'the message is encrypted using the receiver's public key and it is then decrypted using the receiver's private key'.

(c) Candidates generally understood that when using a digital signature to verify a message after it has been received, two message digests are compared. It was less clear from candidates as to how these two digests were derived, namely the one sent as part of the digital signature and the other generated at the receiver's end using the same hashing algorithm.

Question 7

(a) A high proportion of candidates correctly completed the Karnaugh map (K-map) for the given Boolean expression. Sometimes, all the cells in the K-map were not filled in with either a 0 or a 1. Some other errors were seen where the 1s and 0s were in the wrong cells.

(b) Many candidates correctly identified two groups of four 1s in the correct K-map. Incorrect answers seen included candidates who placed a single loop around all six 1s or candidates who placed multiple loops around every group of two 1s.

(c) A high proportion of candidates who identified correct loops in part **(b)** stated a simplified sum-of-products to represent these loops. Many of these went on to further simplify this Boolean expression to its simplest form.

Question 8

(a) This question was generally well answered, with candidates explaining that virtual memory is an area of secondary storage used to temporarily extend available memory in a computer. A number of candidates incorrectly gave answers that related to virtual machines.

(b) The vast majority of candidates were able to state a difference between paging and segmentation. The most common response was that paging allows memory to be divided into fixed size blocks and segmentation divides memory into variable size blocks. A few candidates incorrectly stated these the wrong way round, and a number of other candidates only gave a statement for either paging or segmentation, not both, which does not therefore give a difference.

Question 9

(a) The vast majority of candidates achieved some credit for this part. Most correct answers were centred around deep learning using artificial neural networks and/or being modelled on the human brain. Some candidates referred to the inclusion of many hidden layers to facilitate deep learning.

(b) Many candidates who achieved credit for this question gave responses centred on uses where hidden patterns would need to be identified, or patterns that would be too complex or time consuming for humans to process. Some candidates gave answers that would be better placed in part **(a)**, because they described what deep learning is rather than reasons it would be used.

Question 10

(a) Some candidates achieved some credit for identifying which statements applied to Reduced Instruction Set Computers (RISC) or Complex Instruction Set Computers (CISC). Many candidates gave too many incorrect responses to gain any credit.

(b) The vast majority of candidates achieved some credit by describing the process of pipelining during the fetch-execute cycle in RISC processors. Some candidates gave good responses to illustrate how different instructions are broken into subtasks and how these may then be processed simultaneously.

Question 11

(a) Some candidates performed well on this part, related to the terms used in Object-Oriented Programming (OOP). Candidates whose responses used OOP terminology throughout generally fared better. Candidates should not use the name of a given term to explain its meaning, for

example, for inheritance, 'a new class of objects inherits the attributes and methods from a parent class', would be not enough and would be better, for example, as 'a new class of objects derives the attributes and methods from a parent class'. Candidates should use the terms property, method, class or object, as applicable, when defining OOP terms.

(b) The vast majority of candidates achieved some credit for this part, and many solutions scored highly. However, very few candidates achieved full marks. Typical errors included missing ENDCLASS as the final statement, not using correct variable names in the declarations and/or the constructor. Some variables are already in use within the given program, so must also be used in the candidates' responses.

Question 12

(a) The vast majority of candidates achieved some credit for this part, and many solutions scored highly. However, very few candidates achieved full marks. Typical errors included an incorrect `IF` statement at the start of the `REPEAT` loop, incorrect assignment of new `Lower` and `Upper` limits or incorrect termination condition for the `REPEAT` loop.

(b) (i) Most candidates correctly stated the Big O notation of a linear search as O(n).

(ii) Many candidates correctly stated that O(log n), as the Big O notation for the binary search algorithm, is a time complexity that uses logarithmic time. Some candidates correctly noted that the search time increases linearly as the number of items in the search list goes up exponentially. However, very few candidates had completely correct answers. Some candidates misinterpreted the question and compared the binary search algorithm to the linear search algorithm in terms of time efficiency.

COMPUTER SCIENCE

Paper 9618/32
Advanced Theory

Key messages

Candidates need to demonstrate a detailed study of the topics covered by the syllabus, ensuring they make good use of technical terminology as appropriate for this advanced theory paper. Candidates who have studied the relevant theory, and who have also practiced and made use of the relevant tools and techniques, are more likely to be able to solve the problems set on the examination paper.

Candidates are advised to answer each question in an appropriate manner for the command word of the question; for example, a question beginning with 'explain' requires more detail than a question beginning with 'identify'. If a question asks for working to be shown, candidates must also ensure that they do this, in order to gain full credit.

Candidates must always make sure that they answer questions in the context of any scenario described in the question, rather than in generic terms.

Candidates are further advised to make use of the published pseudocode guide when preparing for this examination, for example in the areas of user-defined data types or algorithm construction, and answer questions requiring pseudocode answers using this syntax.

General comments

Candidates should make sure that any response they give does not simply rephrase the question, for example, **Question 9(b)** requires a description of a number of scheduling routines, including the shortest job first routine. Candidates who simply wrote that the shortest job is processed first, or similar, have not said enough.

Candidates need to read questions carefully before beginning their answer, for example in **Question 11(c)** candidates are asked to explain how the simplified linked list structure given in the question could be improved. Candidates were expected to explain how the spaces where data had been deleted from the list could be re-used using a free list pointer. Many candidates gave suggestions on how the search algorithm could be improved instead.

Comments on specific questions

Question 1

(a) The majority of candidates correctly gave the largest positive two's complement binary number that could be stored using 11 bits for the mantissa and 5 bits for the exponent.

(b) Many candidates were successful, by converting the given binary floating-point number into denary and showing their working. Some candidates did not correctly demonstrate how their exponent was calculated and how this was applied to the mantissa to find the final answer. Some candidates earned credit for their working where they demonstrated correct methods with the given data.

(c) A large proportion of candidates recognised that underflow occurs where a number is too small to be represented in the available system. Few candidates identified that these very small numbers were likely to be generated following a calculation.

Question 2

(a) The vast majority of candidates correctly identified at least one additional stage in the compilation process, other than lexical analysis and syntax analysis.

(b) Most candidates correctly identified that syntax analysis reports syntax errors or checks that the code matches the grammar of the language. Relatively few candidates gave completely correct answers.

Question 3

(a) The vast majority of candidates demonstrated an understanding of the use of protocols when used in communication between computers, but very few candidates expanded their answers sufficiently.

(b) The vast majority of candidates were able to identify at least one of the missing layers in the given TCP/IP protocol stack.

(c) Candidates were generally aware of the purpose of the IMAP protocol and many of these candidates gave sufficiently detailed answers to gain full credit. Some candidates incorrectly identified this protocol as used for sending emails.

Question 4

(a) Most candidates achieved at least some credit for declaring an enumerated data type. Some candidates used incorrect pseudocode keywords, such as `ENUM` or `DECLARE`, where they should have used `TYPE`.

(b) Many high scoring responses were seen for this question. Some did not use the correct pseudocode keywords in the correct places, for example, `TYPE`, `ENDTYPE` and `DECLARE`, or for not correctly applying an enumerated data type as stated in the question.

(c) (i) This was a generally well answered question, but some errors included the use of `FLIGHT` as the variable name rather than as the data type as stated in the question, or not using the correct pseudocode keyword, `DECLARE`.

(ii) Many candidates achieved highly in this part, but some errors seen included candidates adding additional pseudocode keywords that were not required, such as `DECLARE,` candidates not using correct field names in their responses, or candidates assigning data of the wrong data type, such as strings without quotation marks.

Question 5

This was a generally well answered question with many candidates describing a virtual machine, stating up to two benefits and up to two drawbacks of virtual machines. Some candidates correctly answered only some of these sections. A small number of candidates incorrectly gave answers that were related to virtual memory.

Question 6

(a) The vast majority of candidates could state that symmetric encryption uses a single key and asymmetric encryption uses two keys. Some candidates expanded their answers to state that the symmetric single key was used by all, but only one of the asymmetric keys was available to everyone.

(b) There were a wide range of answers seen. Candidates generally recognised that a Certificate Authority (CA) is responsible for issuing a digital certificate, so the first step of the process would be to make an application to the CA. After this the CA will perform a verification process on the organisation and if that is successful, the certificate is issued.

Question 7

The majority of answers that gained credit did so for identifying the use of labelled input data for supervised learning and/or the use of unlabelled data for unsupervised learning, either in these terms or as descriptions. Other responses that gained credit were that for supervised learning, future output can be predicted based on past data and unsupervised learning is able to identify hidden patterns in data. It was rare to see a fully correct response to either supervised learning or unsupervised learning,

Question 8

(a) A wide range of responses were seen. Correct answers contained either two NAND gates or two NOR gates. Correct connections include the output from each of the logic gates passing back to become one of the inputs of the other logic gate. The inputs would be Set and Reset, but the order of these depended on the type of correct logic gate used. Many correct answers were seen, but also, a wide range of incorrect logic gates were seen.

(b) Candidates were generally aware that a flip-flop is used to store a binary digit.

(c) Candidates mostly gave good attempts at simplifying the given Boolean expression. Many candidates had partial success, with some candidates giving fully correct answers. One of the common errors noted in the application of De Morgan's Laws was where an overarching bar, or NOT, over two Boolean terms was separated to give each term its own individual NOT, but the operator that began as an AND sign remained as an AND sign, rather than being changed to OR.

Question 9

(a) Most candidates demonstrated an understanding of the need for scheduling in process management. The best answers expressed three clear points, such as enabling multitasking, keeping the CPU busy at all times and allowing high priority jobs to be completed first.

(b) Most candidates had some success with this part. The best responses from some candidates gave good descriptions of each of the given scheduling routines that said more than was implied from their name. These candidates would also go on to state specific benefits that could be achieved in each case.

Question 10

(a) Some candidates were successful in this part. Candidates whose responses used OOP terminology throughout generally performed better. Candidates should not use the name of a given term to explain its meaning, for example, 'getter is a method used to get the value of a property', but instead, should say something like 'getter is a method used to return the value of a property'. Candidates should use the terms property, method, class or object, as applicable, when defining OOP terms.

(b) The vast majority of candidates attempted this question with most candidates achieving some success. Typical errors noted were the setting of `Telephone` as an integer data type, not including two setters and/or two getters or incorrect setter parameters.

Question 11

(a) The vast majority of candidates recognised how `NextPointer` was used to order the linked list. Many candidates gave fully correct answers.

(b) This was well answered by the majority of candidates. However, very few candidates achieved full marks. Typical errors included an incorrect `WHILE` condition, incorrect assignment of the next value of `Pointer`, missing out `THEN` in the `IF` statement, or including `Flower(Found)` in the OUTPUT statement when the flower is not found, which could not work as the flower is not in the array.

(c) A small number of good answers were seen where candidates improved the linked list structure, by making use of a free list pointer to re-use the unused elements where data had been deleted. A

high proportion of candidates misinterpreted the question and suggested methods to improve the algorithm rather than the simplified linked list structure.

COMPUTER SCIENCE

Paper 9618/33
Advanced Theory

Key messages

Candidates need to demonstrate a detailed study of the topics covered by the syllabus, ensuring they make good use of technical terminology as appropriate for this advanced theory paper. Candidates who have studied the relevant theory, and who have also practiced and made use of the relevant tools and techniques, are more likely to be able to solve the problems set on the examination paper.

Candidates are advised to answer each question in an appropriate manner for the command word of the question; for example, a question beginning with 'explain' requires more detail than a question beginning with 'identify'. If a question asks for working to be shown, candidates must also ensure that they do this, in order to gain full credit.

Candidates must always make sure that they answer questions in the context of any scenario described in the question, rather than in generic terms.

Candidates are further advised to make use of the published pseudocode guide when preparing for this examination, for example in the areas of user-defined data types or algorithm construction, and answer questions requiring pseudocode answers using this syntax.

General comments

Candidates should make sure that any response they give does not simply restate the question, for example **Question 11(a)** requires a definition of a number of Object-Oriented Programming terms, including inheritance. Candidates who simply used the word inherit in their response without clearly stating what this means in this context have not demonstrated that they understand the meaning of the term.

Candidates need to read questions carefully before beginning their answer, for example in **Question 12(b)(ii)** candidates are asked to describe the meaning of O(log n) as it applies to a binary search algorithm, which means that candidates were only required to give a description of this particular Big O notation in relation to binary searches. Some candidates compared the binary search algorithm with the linear search algorithm instead.

Comments on specific questions

Question 1

(a) Many candidates achieved full credit by converting the given binary floating-point number into denary and showing their working. Some candidates did not correctly demonstrate how their exponent was calculated and how this was applied to the mantissa to find the final answer. Some candidates achieved some credit for their working where they demonstrated correct methods with the given data.

(b) Many candidates correctly demonstrated the conversion of a negative denary number to two's complement notation. A high proportion of candidates recognised the link between this question part and part **(a)** as being the negative version of the number, so that the exponent would be the same and only the mantissa would change.

(c) (i) The majority of candidates correctly stated a reason why the given binary number was not normalised, with a range of answers, including the fact that the two most significant bits of the mantissa were the same digit.

(ii) The vast majority of candidates achieved at least some credit, which was most likely for correctly stating the mantissa in its normalised form. A smaller number of candidates gave the correct exponent.

Question 2

A wide range of quality of responses was seen for this question, with a significant number of candidates demonstrating their knowledge of the functions of the Transport layer and/or the Internet layer of the TCP/IP protocol suite. Some candidates gave answers that were not specific enough; for example, for the internet layer, 'source and destination IP addresses are added to the packet header' is acceptable, whereas 'it uses IP addresses' is not enough.

Question 3

(a) The vast majority of candidates identified that an enumerated data type is a non-composite data type, but they also needed to identify the unique aspect of this data type, i.e., that it includes an ordered list of possible values.

(b) A high proportion of candidates stated the fact that the pointer non-composite data type is used to reference a memory location.

(c) Most candidates achieved at least partial credit for declaring the required data type. Some candidates used an incorrect pseudocode keyword, such as `DECLARE`, where they should have used `TYPE`.

(d) (i) This was a generally well answered question. Errors included the use of `Pet` as the variable name rather than as the data type as stated in the question, or not using the correct pseudocode keyword, `DECLARE`.

(ii) Many candidates performed well on this part. Errors included candidates adding additional pseudocode keywords that were not required, such as `DECLARE,` candidates not using the given field names in their responses, or candidates assigning data of the wrong data type, such as strings or characters without quotation marks.

Question 4

This was a generally well answered question with the vast majority of candidates able to match some or all of the given stages of compilation with their correct descriptions.

Question 5

(a) Some candidates were able to give the correct Reverse Polish Notation (RPN) for the given infix expression.

(b) (i) A high proportion of candidates were able to state the correct infix version of the given RPN expression.

(ii) The vast majority of candidates successfully evaluated their infix expressions from **part (b)(i)** by applying the given values for the variables a, b, c and d.

Question 6

(a) The vast majority of candidates gained at least partial credit for stating that a private key is unpublished, that it has a matching public key or that it is used to decrypt data that was encrypted with its matching public key.

(b) Candidates generally understood the process of asymmetric encryption as a process where one key is used to encrypt the message and another key is used to decrypt it. Sometimes candidates

did not clearly state which key was used to encrypt and/or decrypt the message, or if they did, it was not explicitly clear that the two keys were linked. For example, a good answer would be 'the message is encrypted using the receiver's public key and it is then decrypted using the receiver's private key'.

(c) Candidates generally understood that when using a digital signature to verify a message after it has been received, two message digests are compared. It was less clear from candidates as to how these two digests were derived, namely the one sent as part of the digital signature and the other generated at the receiver's end using the same hashing algorithm.

Question 7

(a) A high proportion of candidates correctly completed the Karnaugh map (K-map) for the given Boolean expression. Sometimes, all the cells in the K-map were not filled in with either a 0 or a 1. Some other errors were seen where the 1s and 0s were in the wrong cells.

(b) Many candidates correctly identified two groups of four 1s in the correct K-map. Incorrect answers seen included candidates who placed a single loop around all six 1s or candidates who placed multiple loops around every group of two 1s.

(c) A high proportion of candidates who identified correct loops in part **(b)** stated a simplified sum-of-products to represent these loops. Many of these went on to further simplify this Boolean expression to its simplest form.

Question 8

(a) This question was generally well answered, with candidates explaining that virtual memory is an area of secondary storage used to temporarily extend available memory in a computer. A number of candidates incorrectly gave answers that related to virtual machines.

(b) The vast majority of candidates were able to state a difference between paging and segmentation. The most common response was that paging allows memory to be divided into fixed size blocks and segmentation divides memory into variable size blocks. A few candidates incorrectly stated these the wrong way round, and a number of other candidates only gave a statement for either paging or segmentation, not both, which does not therefore give a difference.

Question 9

(a) The vast majority of candidates achieved some credit for this part. Most correct answers were centred around deep learning using artificial neural networks and/or being modelled on the human brain. Some candidates referred to the inclusion of many hidden layers to facilitate deep learning.

(b) Many candidates who achieved credit for this question gave responses centred on uses where hidden patterns would need to be identified, or patterns that would be too complex or time consuming for humans to process. Some candidates gave answers that would be better placed in part **(a)**, because they described what deep learning is rather than reasons it would be used.

Question 10

(a) Some candidates achieved some credit for identifying which statements applied to Reduced Instruction Set Computers (RISC) or Complex Instruction Set Computers (CISC). Many candidates gave too many incorrect responses to gain any credit.

(b) The vast majority of candidates achieved some credit by describing the process of pipelining during the fetch-execute cycle in RISC processors. Some candidates gave good responses to illustrate how different instructions are broken into subtasks and how these may then be processed simultaneously.

Question 11

(a) Some candidates performed well on this part, related to the terms used in Object-Oriented Programming (OOP). Candidates whose responses used OOP terminology throughout generally fared better. Candidates should not use the name of a given term to explain its meaning, for

example, for inheritance, 'a new class of objects inherits the attributes and methods from a parent class', would be not enough and would be better, for example, as 'a new class of objects derives the attributes and methods from a parent class'. Candidates should use the terms property, method, class or object, as applicable, when defining OOP terms.

(b) The vast majority of candidates achieved some credit for this part, and many solutions scored highly. However, very few candidates achieved full marks. Typical errors included missing ENDCLASS as the final statement, not using correct variable names in the declarations and/or the constructor. Some variables are already in use within the given program, so must also be used in the candidates' responses.

Question 12

(a) The vast majority of candidates achieved some credit for this part, and many solutions scored highly. However, very few candidates achieved full marks. Typical errors included an incorrect `IF` statement at the start of the `REPEAT` loop, incorrect assignment of new `Lower` and `Upper` limits or incorrect termination condition for the `REPEAT` loop.

(b) (i) Most candidates correctly stated the Big O notation of a linear search as O(n).

(ii) Many candidates correctly stated that O(log n), as the Big O notation for the binary search algorithm, is a time complexity that uses logarithmic time. Some candidates correctly noted that the search time increases linearly as the number of items in the search list goes up exponentially. However, very few candidates had completely correct answers. Some candidates misinterpreted the question and compared the binary search algorithm to the linear search algorithm in terms of time efficiency.

COMPUTER SCIENCE

Paper 9618/41
Practical

Key messages

Candidates need to submit only their evidence document, and all evidence of their code and screenshots from testing must be included within this Microsoft Word document. This should be a single document and not a zip file.

When completing the evidence document, candidates should make sure their evidence is in the correct answer space and should not delete any other answer spaces.

Program code should be copied into each answer space and checked to make sure all required program code is present, including all closing statements and brackets. Candidates using Python must make sure that indentation is included and accurate in their answer. Screenshots must not be used for program code (unless stated) because the resolution of the image often does not allow for it to be read clearly.

Candidates must make sure all inputs to that program are also included when using screenshots to show the outputs of their program. The screenshot must be clear and legible. Some screenshots were too small or pixelated, and it was not possible to read the values that were output. Screenshots must be of black text on a white background. A black background with coloured text is often illegible.

General comments

Candidates must read each question carefully and consider all requirements, for example the values that need to be passed as parameters and those that need to be read in from the user. Candidates need to make sure they are answering the question given and not changing the design to one that they prefer, for example how a class or array are used to store and manipulate data.

When converting pseudocode, candidates need to make sure they consider their language and the differences between this and the pseudocode provided, adjusting their solution appropriately.

Comments on specific questions

Question 1

This question required candidates to read data from a file, search the data for a value and sort the data.

(a) Many candidates were able to declare a 1-dimensional array and give it the appropriate identifier. Fewer candidates declared or identified the need for the 100 elements in the array. When a list is being used instead of an array such as in Python, candidates still need to identify the criteria of the array; this could be by initialising 100 integer values in the array or using an appropriate comment to demonstrate their understanding of the limitation of the array.

(b) There were a range of approaches used to read the data from the file. Some candidates looped until end of file was encountered and stored each value in the array. Another way was to read all values into a variable or a list and then split these into individual elements.

This question required candidates to use appropriate exception handling, which was not attempted by some candidates. The most common use was when opening the file. Some candidates did not make appropriate use of exception handling. They did not include program statements within the

exception. It is expected that an appropriate output is generated from the exception. This could be an inbuilt function called or an output statement giving a description of the error when caught.

A common error was to not close the file once opened or to close it in an inappropriate position, such as within the loop that is reading in the values. Where the file opening statement also closes the file, an additional closing statement is not required.

(c) Answers often included the input of the data to search for. Some candidates took this as a parameter to the function which did not meet the requirement of the question. The question stated that the function asks the user to enter a number to search for. The question gave constraints for this input. It must be a whole number (integer) between 1 and 100 inclusive. This required candidates to validate the input. The most common validation was checking the value between 1 and 100. Fewer candidates checked that the input was an integer, some candidates chose to cast the input as an integer which was acceptable. A common error was including the check for valid data, but then not dealing with an invalid input appropriately, for example outputting an error message and then still using that invalid input to search in the array.

Many candidates were able to search the array appropriately, commonly using a linear search. Some candidates did not count the number of times the input appears, instead returning the index in the array where it was found.

(d) (i) Many candidates called the correct subroutines. Fewer candidates used the return value from `FindValues()` appropriately. Some candidates instead chose to edit the function to output the message, which did not meet the requirements of the question.

Subroutine (function and procedure) calls must match those the candidates have created. For example, if a function takes 2 parameters, then when the function is called, it must pass 2 values as parameters. If a candidate edits the function at this stage such as to no longer take 2 parameters, then they will need to update this in their answer to that question part so that the program can be followed logically.

(ii) Many candidates were able to generate the correct output for this test. Some candidates did not input the number required by the question and therefore did not meet the requirement.

(e) Candidates were often able to create a bubble sort algorithm accurately. There were a range of algorithms, some more efficient than others although efficiency was not a requirement in the question.

The question also required the procedure to print the contents of the sorted array. This was often missed by candidates who did not output the contents in their solution or outputted the data in the main program instead of the procedure.

The question also required the procedure to be called from the main program. Some candidates included their main program to show this procedure call, which met the requirements, whilst some candidates did not include this.

Question 2

This question required candidates to produce a program using object-oriented programming to produce a prototype for a card game.

(a) (i) Many candidates accurately declared the class and its constructor. Fewer candidates passed the required parameters to the constructor, with some candidates assigning set values to these instead. Many candidates identified the attributes as private either through the declaration or by including a comment stating the intention (as in Python). Some candidates did not include the data type in the declaration of these attributes.

(ii) Many candidates were able to define the two methods and return the appropriate attribute. When using Python, class methods required `self` to be passed as a parameter. Some candidates did not include this in their methods and therefore declared functions external to the class.

Some candidates attempted to pass a value into the method and then return this value, or incorrectly store this value in an attribute.

(iii) This part required candidates to declare a new object of type `Card` for each of the cards given in the table. Some candidates declared a variable of type `Card` for each, whilst some candidates created an array of type `Card` and stored them in this array.

Many candidates found this question challenging. A common incorrect answer included declaring two variables for each card, one to store the number and one to store the colour.

(b) (i) Some candidates found this question challenging and attempted to use inheritance within this class declaration, with the class `Hand` inheriting from `Card`. The class `Hand` instead has an array of objects of type `Card`. Some candidates were able to define this class accurately, independently of `Card`.

Few candidates passed the required parameters to the constructor, which required 5 objects of type `Card`. Many candidates were able to initialise the correct values to the two other attributes in the class, although some candidates attempted to pass these as parameters instead of storing the given values.

(ii) This part required a parameter to be passed to the get method, and then the value of the card at the parameter index returned. Some candidates did not pass a parameter and therefore could not return the value at this index. Some candidates attempted to read a value in from the user and then use this value, which did not meet the requirements.

(iii) Some candidates found this question challenging and passed the integer and string values to the constructor instead of passing `Card` objects.

(c) (i) This part required the writing of a function that made use of the classes. Some candidates attempted to write a method for one of the classes instead of an independent function.

Candidates needed to pass a `Hand` object to the function. Some candidates missed this or passed an individual card which did not allow them to then loop through each `Card` object.

Some candidates were able to identify the need to loop through each `Card` and access its colour. Fewer candidates made correct use of the get method to access the colour.

The correct score was often added, but few candidates also added the number for each card in the hand to this total.

Some candidates output the final score instead of returning the score as required by the question.

(ii) The candidates who attempted this part often called the correct function; sometimes, they passed an incorrect value, or only called it once instead of once for each player. The return value was often accurately stored and appropriate messages output. Some candidates did not use accurate selection statements for the output, for example using an if statement to determine if player 1's score was greater and using a separate if statement to check if player 2's score was greater with an else for a draw. This would mean that when player 1 wins it would also output that it was a draw.

(iii) There were several accurate outputs that showed that player 2 wins.

Question 3

This question required candidates to create a binary tree that matched the design in the question.

(a) Many candidates attempted this question but were unable to define a 2-dimensional array accurately. This array had 20 elements by 2 elements, and candidates needed to initialise this quantity of elements. A common error was the bounds being 1 less, for example 19 elements by 1 element.

(b) This part provided candidates with the data to store in the array they initialised in part **(a)**. Many candidates were able to do this successfully, although some candidates had errors in the values used.

(c) In this part, candidates were required to read the pseudocode function, complete the missing statements and convert it into their chosen programming language.

When converting a pseudocode algorithm to their chosen language candidates needed to consider the differences between pseudocode and their chosen programming language. This will differ depending on the language they use. For example, the pseudocode includes both indices for the array in one set of brackets such as `(Root, 1)`. In Python, and Java, this needs changing to two sets of brackets such as `(Root)(1)`. In VB.NET, this needs changing to round brackets such as `(Root, 1)`. This change was often missed by candidates who copied the pseudocode directly.

The first missing space was often incorrectly given as `-1`, whereas this is the condition when the value is found and therefore the current index needed to be returned.

Many candidates accurately identified the `>` in the second space and `Root` in the third space. The final space was often given correctly, although a common error was `2`.

(d) This part required a recursive procedure to create a post order traversal, with the order of traversal and output given in the question. Many candidates did attempt a recursive algorithm with the most common mark awarded for the output.

There were a range of methods of checking if they were at a leaf node, many of which were accurate. A common error was to give the incorrect order such as by having multiple outputs at different positions in the algorithm, which resulted in multiple outputs and hence not in the correct order.

(e) (i) Some candidates were able to call the function and procedure they had defined. A common error was not matching the parameters in their own answers to previous question parts, for example mis-matching parameters, or not including the number 15 as a parameter to `SearchValue()`.

(ii) Many candidates did not attempt this part. Few of those who did had an accurate output that gave the correct position for the number 15 and the post order in the correct order.

COMPUTER SCIENCE

Paper 9618/42
Practical

Key messages

Candidates need to submit only their evidence document, and all evidence of their code and screenshots from testing must be included within this Microsoft Word document. This should be a single document and not a zip file.

When completing the evidence document candidates, should make sure their evidence is in the correct answer space and should not delete any other answer spaces.

Program code should be copied into each answer space and checked to make sure all required program code is present, including all closing statements and brackets. Candidates using Python must make sure that indentation is included and accurate in their answer. Screenshots must not be used for program code (unless stated) because the resolution of the image often does not allow for it to be read clearly.

Candidates must make sure all inputs to that program are also included when using screenshots to show the outputs of their program. The screenshot must be clear and legible. Some screenshots were too small or pixelated, and it was not possible to read the values that were output. Screenshots must be of black text on a white background. A black background with coloured text is often illegible.

Data structures can be designed in many ways; some may be more common than others. For example, the tail pointer in a queue could point to the last element in the queue, or the next space. Candidates need to consider the information given in the question to identify how the given data structure is designed to make sure they are writing a solution that meets all the requirements.

General comments

Candidates must read the question carefully and consider all requirements, for example which values are to be passed as parameters and which are to be read in from the user. Candidates need to make sure they are answering the question given and not changing the design to one that they prefer, for example how a class or array are used to store and manipulate data.

Comments on specific questions

Question 1

This question required candidates to store data in an array and then sort and output the array contents.

(a) Many candidates were able to declare an array, but fewer declared a 2-dimensional array with the correct number of elements. Candidates must make sure they are declaring arrays that meet the requirements. Where a specific number of elements is provided and the programming language does not require this in the declaration, candidates must demonstrate their understanding of the requirement, for example by initialising the given number of elements, or by using an appropriate comment that indicates the number of elements.

(b) Many candidates were able to initialise the value for `NumberOfJobs` within the procedure, but fewer initialised all the required array elements. A common error was to initialise an incorrect number of array elements, for example looping through 19 elements in one dimension instead of the required 20.

(c) Many candidates were able to create an appropriate function that took the required parameters. Some candidates attempted to take both values as input within the procedure instead of as parameters which did not meet the question requirements.

There were a range of approaches to checking if there was space within the array. The variable `NumberOfJobs` was given in the question to allow direct access to the next element in the array. Some candidates chose to loop through all the array elements until they found one that was empty (had a value of –1 in one of the 2-dimensions). A common error was the condition being used to check if it was full, for example checking if `NumberOfJobs` was equal to 99, or if it was greater than 100.

(d) Many candidates called the required subroutines, but fewer used the given data to store each job and its priority. Some candidates read the values as input but did not show these specific jobs being stored, for example by calling the procedure with these values.

Candidates need to make sure their subroutine calls match their declarations, for example if they write a function that takes two parameters, their function call must also pass two parameters.

(e) This part required candidates to write an insertion sort on the data. Many candidates were able to use at least one suitable loop in their sorting algorithm. A common error was in the second loop where a bubble sort was written instead of an insertion sort. Each array element had two values, job number and priority. Some candidates only swapped one of these data items and did not swap both.

There were a range of different insertion sorts used. One swapped the data items down until they were in the correct position; a second method removed the data item then moved each item in the array up until the position was found and the data inserted in position.

Some candidates included the empty array elements in the insertion sort. These were all initialised to –1 in an earlier question part which meant that these were now the first elements in the array instead of the jobs with the highest priority.

(f) This part required each of the jobs to be output. Some candidates printed all data within the array that included the empty values. Stronger answers checked for these empty values or used the value within `NumberOfJobs` to control which elements were output. Many candidates were able to output the required message accurately and on the same line.

(g) (i) Many candidates were able to call their procedures accurately.

(ii) This output required evidence of the insertion of each array element and the output of the data in order. Some candidates cropped the screenshot to only include the priorities.

Question 2

This question required candidates to produce a program using object-oriented programming to create a character and manipulate its coordinates.

(a) Many candidates accurately declared the class. Fewer candidates created the constructor accurately with the three required parameters. Some candidates identified the attributes as private through the declaration by including a comment stating the intention (as in Python). Some candidates did not include the data type in the declaration of these attributes.

(b) Many candidates were able to define the three methods and return the appropriate attribute. When using Python, class methods required `self` to be passed as a parameter. Some candidates did not include this in their methods hence declared functions external to the class.

Some candidates attempted to pass a value into the method and then return this value, or store incorrectly this value in an attribute.

(c) Many candidates were able to declare the method `ChangePosition()` accurately and pass in the required parameters. Some candidates replaced the attribute values with the parameters instead of adding them to the current contents.

Some candidates only passed one parameter and therefore only changes one of the values.

(d) This part required candidates to read data for ten characters from a given text file and create an object of type `Character` for each one.

Candidates often successfully opened the given text file, but fewer closed this in an appropriate place in the algorithm. Where the file-opening statement also closes the file, an additional closing statement is not required.

There were a range of approaches to reading in the data. Some candidates looped until end of file was reached or looped 10 times; some candidates read all the data into an appropriate structure and then split the data from this.

(e) Some candidates found the accessing of the character names challenging. The private attributes meant that the use of the get methods was required to access the name from the object. A common error was comparing the whole object to the input by not accessing the name attribute from the object. The solution required searching through each of the 10 characters until the correct object was found. A common error was that not all objects were checked, for example only comparing the name of 9 objects.

The question required repetition until the user enters a name that exists in the array (is one of the character names). This was missed by some candidates who returned a null value, for example −1 if the name was not found after looping once. To complete the solution this would need to ask for another input inside a loop; some candidates looped repeatedly until found but did not input a different name each time, hence creating an infinite loop.

(f) Many candidates were able to compare the input to the four possible inputs accurately, but some candidates did not repeat this until a valid option was input. Few candidates were able to use the correct method with the object identified from their solution to part **(e)**. A common error was to call the method as a procedure and therefore not make use of the object correctly.

(g) (i) This part required accessing the updated coordinates for the selected object. A common error was to output a variable such as `xCoordinate`, without reference to how this was accessed from the object.

(ii) Some candidates were able to enter the required inputs and produce an output showing the correct result. Some candidates did not enter the required input data accurately, whilst some candidates had cropped screenshots that did not show all the inputs and how their solution responded to each.

Question 3

This question required candidates to create and manipulate a queue data structure.

(a) This part identifies the purpose of the head and tail pointer in this queue. The head pointer points to the first element in the queue, and hence an appropriate null value would need to be an index not within the array, such as −1. The tail pointer points to the next free space in the queue, when there are no items in the queue and it is being set up, then it needs to point to the first free space (index 0). Some candidates gave appropriate values for these pointers, but a common error was declaring the head pointer to index 0, or the tail pointer to −1.

(b) In this part candidates needed to store the parameter value in the next position in the array. The tail pointer was identified as pointing to the next free space; therefore, it is the position where it needs to be stored. This is before incrementing the pointer. Some candidates incremented the tail pointer before storing the data or did not increment the pointer after storing the data.

Some candidates did not identify the need to check if the queue was already full or comparing the data in the head pointer to check if the queue was full instead of the tail pointer.

(c) Many candidates used a loop to call the `Enqueue()` function, but fewer had an accurate loop that sent the numbers 1 to 20 to as parameters. A common error was to loop from 0 to 19 and not add 1 to this value within the loop or to use an incorrect ending condition so the loop only ran 19 times.

The question stated that 'Successful' should be output if all numbers are enqueue and 'Unsuccessful' otherwise. A common error was to output this message for each value that was enqueued, so 'Successful' would be output several times even if all numbers were not successfully enqueued.

(d) This part provided a pseudocode iterative function that totalled all the values in the queue. Candidates were required to convert this into a recursive function. Candidates found this question challenging with many not writing a recursive function, instead they rewrote it using another loop.

Some candidates wrote an accurate recursive solution and there were many different approaches that all performed the same actions as the given function. Some candidates attempted to write a new function that used different parameter values and did not calculate the total in a similar way to the given function, hence not rewriting the same function.

(e) (i) Some candidates were able to call their function with the correct value. The parameter could vary depending on the candidate's solution to part **(d)**. Some candidates did not include a parameter. Some candidates called the function but did not output the return value.

(ii) Some candidates provided a screenshot showing the correct output.

COMPUTER SCIENCE

Paper 9618/43
Practical

Key messages

Candidates need to submit only their evidence document, and all evidence of their code and screenshots from testing must be included within this Microsoft Word document. This should be a single document and not a zip file.

When completing the evidence document, candidates should make sure their evidence is in the correct answer space and should not delete any other answer spaces.

Program code should be copied into each answer space and checked to make sure all required program code is present, including all closing statements and brackets. Candidates using Python must make sure that indentation is included and accurate in their answer. Screenshots must not be used for program code (unless stated) because the resolution of the image often does not allow for it to be read clearly.

Candidates must make sure all inputs to that program are also included when using screenshots to show the outputs of their program. The screenshot must be clear and legible. Some screenshots were too small or pixelated, and it was not possible to read the values that were output. Screenshots must be of black text on a white background. A black background with coloured text is often illegible.

General comments

Candidates must read each question carefully and consider all requirements, for example the values that need to be passed as parameters and those that need to be read in from the user. Candidates need to make sure they are answering the question given and not changing the design to one that they prefer, for example how a class or array are used to store and manipulate data.

When converting pseudocode, candidates need to make sure they consider their language and the differences between this and the pseudocode provided, adjusting their solution appropriately.

Comments on specific questions

Question 1

This question required candidates to read data from a file, search the data for a value and sort the data.

(a) Many candidates were able to declare a 1-dimensional array and give it the appropriate identifier. Fewer candidates declared or identified the need for the 100 elements in the array. When a list is being used instead of an array such as in Python, candidates still need to identify the criteria of the array; this could be by initialising 100 integer values in the array or using an appropriate comment to demonstrate their understanding of the limitation of the array.

(b) There were a range of approaches used to read the data from the file. Some candidates looped until end of file was encountered and stored each value in the array. Another way was to read all values into a variable or a list and then split these into individual elements.

This question required candidates to use appropriate exception handling, which was not attempted by some candidates. The most common use was when opening the file. Some candidates did not make appropriate use of exception handling. They did not include program statements within the

exception. It is expected that an appropriate output is generated from the exception. This could be an inbuilt function called or an output statement giving a description of the error when caught.

A common error was to not close the file once opened or to close it in an inappropriate position, such as within the loop that is reading in the values. Where the file opening statement also closes the file, an additional closing statement is not required.

(c) Answers often included the input of the data to search for. Some candidates took this as a parameter to the function which did not meet the requirement of the question. The question stated that the function asks the user to enter a number to search for. The question gave constraints for this input. It must be a whole number (integer) between 1 and 100 inclusive. This required candidates to validate the input. The most common validation was checking the value between 1 and 100. Fewer candidates checked that the input was an integer, some candidates chose to cast the input as an integer which was acceptable. A common error was including the check for valid data, but then not dealing with an invalid input appropriately, for example outputting an error message and then still using that invalid input to search in the array.

Many candidates were able to search the array appropriately, commonly using a linear search. Some candidates did not count the number of times the input appears, instead returning the index in the array where it was found.

(d) (i) Many candidates called the correct subroutines. Fewer candidates used the return value from `FindValues()` appropriately. Some candidates instead chose to edit the function to output the message, which did not meet the requirements of the question.

Subroutine (function and procedure) calls must match those the candidates have created. For example, if a function takes 2 parameters, then when the function is called, it must pass 2 values as parameters. If a candidate edits the function at this stage such as to no longer take 2 parameters, then they will need to update this in their answer to that question part so that the program can be followed logically.

(ii) Many candidates were able to generate the correct output for this test. Some candidates did not input the number required by the question and therefore did not meet the requirement.

(e) Candidates were often able to create a bubble sort algorithm accurately. There were a range of algorithms, some more efficient than others although efficiency was not a requirement in the question.

The question also required the procedure to print the contents of the sorted array. This was often missed by candidates who did not output the contents in their solution or outputted the data in the main program instead of the procedure.

The question also required the procedure to be called from the main program. Some candidates included their main program to show this procedure call, which met the requirements, whilst some candidates did not include this.

Question 2

This question required candidates to produce a program using object-oriented programming to produce a prototype for a card game.

(a) (i) Many candidates accurately declared the class and its constructor. Fewer candidates passed the required parameters to the constructor, with some candidates assigning set values to these instead. Many candidates identified the attributes as private either through the declaration or by including a comment stating the intention (as in Python). Some candidates did not include the data type in the declaration of these attributes.

(ii) Many candidates were able to define the two methods and return the appropriate attribute. When using Python, class methods required `self` to be passed as a parameter. Some candidates did not include this in their methods and therefore declared functions external to the class.

Some candidates attempted to pass a value into the method and then return this value, or incorrectly store this value in an attribute.

(iii) This part required candidates to declare a new object of type `Card` for each of the cards given in the table. Some candidates declared a variable of type `Card` for each, whilst some candidates created an array of type `Card` and stored them in this array.

Many candidates found this question challenging. A common incorrect answer included declaring two variables for each card, one to store the number and one to store the colour.

(b) (i) Some candidates found this question challenging and attempted to use inheritance within this class declaration, with the class `Hand` inheriting from `Card`. The class `Hand` instead has an array of objects of type `Card`. Some candidates were able to define this class accurately, independently of `Card`.

Few candidates passed the required parameters to the constructor, which required 5 objects of type `Card`. Many candidates were able to initialise the correct values to the two other attributes in the class, although some candidates attempted to pass these as parameters instead of storing the given values.

(ii) This part required a parameter to be passed to the get method, and then the value of the card at the parameter index returned. Some candidates did not pass a parameter and therefore could not return the value at this index. Some candidates attempted to read a value in from the user and then use this value, which did not meet the requirements.

(iii) Some candidates found this question challenging and passed the integer and string values to the constructor instead of passing `Card` objects.

(c) (i) This part required the writing of a function that made use of the classes. Some candidates attempted to write a method for one of the classes instead of an independent function.

Candidates needed to pass a `Hand` object to the function. Some candidates missed this or passed an individual card which did not allow them to then loop through each `Card` object.

Some candidates were able to identify the need to loop through each `Card` and access its colour. Fewer candidates made correct use of the get method to access the colour.

The correct score was often added, but few candidates also added the number for each card in the hand to this total.

Some candidates output the final score instead of returning the score as required by the question.

(ii) The candidates who attempted this part often called the correct function; sometimes, they passed an incorrect value, or only called it once instead of once for each player. The return value was often accurately stored and appropriate messages output. Some candidates did not use accurate selection statements for the output, for example using an if statement to determine if player 1's score was greater and using a separate if statement to check if player 2's score was greater with an else for a draw. This would mean that when player 1 wins it would also output that it was a draw.

(iii) There were several accurate outputs that showed that player 2 wins.

Question 3

This question required candidates to create a binary tree that matched the design in the question.

(a) Many candidates attempted this question but were unable to define a 2-dimensional array accurately. This array had 20 elements by 2 elements, and candidates needed to initialise this quantity of elements. A common error was the bounds being 1 less, for example 19 elements by 1 element.

(b) This part provided candidates with the data to store in the array they initialised in part **(a)**. Many candidates were able to do this successfully, although some candidates had errors in the values used.

(c) In this part, candidates were required to read the pseudocode function, complete the missing statements and convert it into their chosen programming language.

When converting a pseudocode algorithm to their chosen language candidates needed to consider the differences between pseudocode and their chosen programming language. This will differ depending on the language they use. For example, the pseudocode includes both indices for the array in one set of brackets such as `(Root, 1)`. In Python, and Java, this needs changing to two sets of brackets such as `(Root)(1)`. In VB.NET, this needs changing to round brackets such as `(Root, 1)`. This change was often missed by candidates who copied the pseudocode directly.

The first missing space was often incorrectly given as `-1`, whereas this is the condition when the value is found and therefore the current index needed to be returned.

Many candidates accurately identified the `>` in the second space and `Root` in the third space. The final space was often given correctly, although a common error was `2`.

(d) This part required a recursive procedure to create a post order traversal, with the order of traversal and output given in the question. Many candidates did attempt a recursive algorithm with the most common mark awarded for the output.

There were a range of methods of checking if they were at a leaf node, many of which were accurate. A common error was to give the incorrect order such as by having multiple outputs at different positions in the algorithm, which resulted in multiple outputs and hence not in the correct order.

(e) (i) Some candidates were able to call the function and procedure they had defined. A common error was not matching the parameters in their own answers to previous question parts, for example mis-matching parameters, or not including the number 15 as a parameter to `SearchValue()`.

(ii) Many candidates did not attempt this part. Few of those who did had an accurate output that gave the correct position for the number 15 and the post order in the correct order.