COMPUTER SCIENCE

Paper 9618/11
Theory Fundamentals

Key messages

There is a greater emphasis on the application of knowledge in the 9618 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.

Candidates must have a clear understanding of the different command words used in the questions. A question that begins with the command word 'Explain' requires a different type of answer to one that begins with the keyword 'State'.

This is a technical subject and at this level of study it is expected that answers will be given in technical terms. Use of the correct terminology is very important.

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.

General comments

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

Questions about logic expressions and security were usually completed successfully, the questions about low-level computing 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 was quite a bit of confusion between the different names. Many answers used the terms megabyte or kilobyte instead of gigabyte and mebibyte or kibibyte instead of gibibyte. Simple statements such as, '*a terabyte is smaller than a tebibyte*' are not enough for credit at this level. Great care needs to be taken when writing the prefixes. Some answers that would otherwise have been correct stated that a tebibyte is also 1024 gigabytes instead of 1024 gibibytes.

(b) There were many correct answers to this question. Some candidates need to improve their understanding of two's complement binary. A negative value does not begin with a zero.

(c) Replies which first converted the denary number to 8-bit binary almost invariably completed this conversion correctly and then continued to correctly convert the two nibbles to hexadecimal. Those choosing to divide by 16 to convert the given denary value directly to hexadecimal frequently gave an incorrect value for the second hexadecimal digit which was the remainder from the division.

(d) This question was answered well. Weaker answers did not complete the addition in binary. Instead converting both values to denary, adding them together and then converting the result back to binary. This was not what was required.

Question 2

(a) (i) This question proved challenging. Candidates must have a firm understanding of the different types of computer memory. A frequent incorrect answer in the first blank space was '*computer*'. The rubric of the question stated that the paragraph was about computer memory. Marks are not awarded for simply repeating statements given in the question.

(ii) In this part, many of the responses simply repeated the expansion of the acronym. For example, '*EEPROM can be erased electrically*' does not show any further understanding than the expansion of the acronym, which is already given in the stem of the question.

(b) There were some excellent complete answers to this question. Some confused a magnetic hard disk with an optical disk and wrote about the use of lasers and pits and lands. Candidates need to look carefully at the number of marks allocated to each question. If there are 5 marks for a question, candidates should ensure that they give at least five distinct points in their answer.

(c) (i) This question was answered very well overall. A small number of scripts had overwritten a 1 with a 0 or vice versa, so that it was not clear what their final answer was. If a binary digit needs to be changed the original should be completely crossed out and the new value clearly shown to avoid any misinterpretation.

(ii) This question was also answered very well. There was some confusion between the XOR gate and a NOR gate, and a small number of candidates gave incorrect Boolean notation rather than using the names of the gates.

Question 3

(a) All candidates should be clear on the difference between security of data and privacy of data. Simple statements such as, '*security is how safe the data is, and privacy is how private the data is*' are insufficient at this level. Answers about security should refer to prevention of loss or corruption of the data and answers about privacy should include the prevention of unauthorised access.

(b) There were some very good answers to this question. Many candidates were able to correctly identify two threats and state an appropriate security measure. Some candidates found describing the threats more challenging. This question was set in the context of a teacher writing examination papers and answers were expected to refer to that context. Some responses identified threats that were inappropriate in the given scenario.

Question 4

(a) There were some good answers to this question. Many of the answers were very generalised for example, '*more secure*' or '*more efficient*' which are not detailed enough for credit at this level. Quite a few answers included statements about relational databases that applied equally to files. Candidates should be aware that restrictions such as access rights can be applied to files in the same way as to a database. A small number of candidates compared a relational database to a paper-based filing system.

(b) This question proved very challenging to most. Candidates need to improve their understanding of the First, Second and Third Normal Forms. Even respondents who correctly identified that the database was in 3NF found it challenging to give a justification.

(c) (i) Writing this SQL script also proved very challenging, especially the clauses for setting up the Primary and Foreign Keys. The clauses to set up the three attributes were often correct. Candidates need to keep in mind that many data types from programming languages do not apply in SQL.

(ii) There were better SQL scripts in response to this question. Some did not show understanding that quotation marks are required around text constants and that the SQL function to find the average is AVG. Candidates have a firm understanding of SQL for higher marks. The best way to do this is to practice setting up a simple multi-table database and then using SQL to manipulate the data contained therein.

(d) This question begins with the command word 'Explain', so simply naming a validation check or a method of verification is not enough. The question is also set in the context of the marks awarded to candidates, so it would be expected that this would be referred to in answers. Explanations were frequently generic, describing verification and validation rather than saying how verification and validation checks are used when the test mark is entered into the database.

Question 5

(a) Most candidates were able to match at some of the IDE features to the corresponding description. The IDE features most often correctly identified were single-stepping and breakpoint. There was considerable confusion between context-sensitive prompt and dynamic syntax check.

(b) This was very well answered. Many replies correctly identified two suitable software licences. Candidates should understand the difference between Freeware and Free Software.

(c) Some responses found this question challenging. There was much confusion about the use of a program library, with answers describing the organisation of files and ease of access.

Question 6

(a) (i) There were some very good answers to this question. Candidates should be clear on the understanding of the use of registers in the Fetch-Execute cycle. Registers are simply storage locations; it is the contents of the registers that changes not the registers themselves.

(ii) Many replies found this question challenging. Few mentioned the need to check the priority of the interrupt, and that for the interrupt to be serviced its priority needed to be higher than the priority of the current process. There was considerable confusion about the use of the Interrupt Service Routines (Interrupt Handlers) with answers often suggesting that a single ISR would be used no matter what the cause of the interrupt.

(b) Many responses were able to correctly identify a factor that would affect the performance of a computer system. Clock speed and number of cores were popular choices. Describing the impact was more challenging. Many answers just reworded the question, for example, '*increasing the clock speed would improve the performance*'. At this level that answer is not enough, there needs to be some explanation of how or why the performance is improved. What effect does the factor identified have on the overall system?

(c) (i) This question was answered very well. Most candidates were able to correctly complete the given instruction.

(ii) This question was also answered very well. Most correctly completed the given instruction.

(iii) There was some confusion between left and right. A common incorrect answer was 0000 0110 where respondents had performed a logical shift of 4 places to the right rather than the left. There was also some confusion between a logical shift and a cyclic shift. Another popular incorrect answer was 1101 0110, where candidates had performed a left cyclic shift of 4 places rather than a left logical shift.

COMPUTER SCIENCE

Paper 9618/12
Theory Fundamentals

Key messages

There is a greater emphasis on the application of knowledge in the 9618 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 'Explain' requires a different type of answer to one that begins with the keyword 'State'.

This is a technical subject and at this level of study it is expected that answers will be given in technical terms. Use of the correct terminology is very important.

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.

General comments

If a question asks for a given number of answers, for example, 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 representation of images and manipulation of binary values 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 many correct answers to this question. Some answers seemed to confuse the number of pixels wide by the number of pixels high (the image resolution) with the number of bits used to represent each colour (the bit depth).

(b) (i) This question was also answered well, the most common incorrect answer was 9, presumably because there were 9 different colours in the diagram.

(ii) The question asked for the answer in bytes, the answer to **part 1(b)(i)** was in bits. This was an example of a situation where the questions needed to be read carefully. Some answers correctly calculated the size of the image in bits, but then omitted to divide by 8 to bring the answer to bytes.

(c) This was a question where answers needed to be exact. Vague answers such as '*a **change** in colour depth will **increase** the size of the file*' were common. This answer does not specify whether the change in colour depth is an increase or a decrease. A precise answer is, '*an **increase** in the colour depth will **increase** the size of the file*'.

(d) Many answers correctly named or described a method of lossless compression. Run-length encoding was easily the most popular choice. Some answers did not achieve both of the marks because their answer did not explain how the method would have been applied to the row of pixels given in the question. This question was set in a specific context and needed an answer to be applied rather than being completely generic.

Question 2

(a) There were a small number of good answers to this question, but many candidates found it very challenging to explain why the lane detection system was an embedded system. Vague answers such as, '*it is an embedded system because it is embedded in the car*' will not gain credit at this level of study. Many answers simply repeated the information given in the question. There was also considerable confusion with generic control systems. A popular incorrect answer was, '*it is an embedded system because it is a control system*'.

(b) There were a number of good correct answers to this question, some candidates were clearly able to identify features that would be held in both types of memory for the context given. There was some confusion between primary and secondary memory, with quite a few answers the wrong way round. Again, this question was asked in a context and so answers needed to be applied to that context. Generic answers were insufficient.

(c) There were many correct answers to this question. The statement most likely to have been incorrectly identified was the last one.

Question 3

(a) Some respondents found this question challenging. The most popular correct answers were OR #255 or XOR #154. Several correctly realised that the operand for an OR instruction would be 1111 1111 in binary, but then wrote that as #1 instead of #255.

(b) There were the same challenges here as in **Question 3(a)**. The most popular correct answer was XOR #255, but again some candidates incorrectly interpreted 1111 1111 in binary as #1.

(c) This question had a strong response with most candidates correctly converting the positive binary integer to hexadecimal.

(d) There was some confusion between left and right. A common incorrect answer was 0000 1111 where respondents had performed a logical shift of 3 places to the right rather than the left. There was also some confusion between a logical shift and a cyclic shift. Another common incorrect answer was 1111 0011, where replies had performed a left cyclic shift of 3 places rather than a left logical shift.

(e) The conversion of the two denary numbers to binary was usually completed correctly. The question asked for the conversion to 8-bit binary. A small number of candidates did not include all 8 bits in the conversion of denary 12. The question also asked for the subtraction to be completed in binary. It was frequently very difficult to see where this had been done with rough working out scattered all over the blank space below the question. Candidates should be encouraged to use the blank space for rough work but then to copy the final working and answer into the answer space provided and to cross out the rough work. Some answers performed the subtraction the wrong way round.

Question 4

(a) Candidates should be clear on the differences between the need to keep the data secure, and the need to keep the system secure. The question asked why the school needed to protect them both from unauthorised access, so vague answers about needing to prevent hacking were not enough. Hacking is another term for unauthorised access and that was given in the question. Similarly answers that stated that the data needed to be secure because it was about staff and candidates again just repeated information given in the question. Stronger answers included a statement about ensuring that the data was not changed or deleted.

(b) The majority of candidates were able to correctly identify two threats to the data and also give appropriate prevention methods. Describing the threats was more challenging. At this level of study answers should contain more than just general knowledge. For instance, the description of a virus,

would be expected to include that it was malicious, possibly able to replicate, and what effect the virus could have on the data. Some answers included two threats that were not sufficiently different, for example, malware and virus.

(c) This question was one which needed very precise use of language and some responses struggled to describe encryption because they did not use the appropriate terminology. '*Jumbled up data*' is not enough of a description, a better answer is '*data is turned into cipher text*'. Candidates need to understand that encryption does not prevent the data being read or accessed; encryption ensures that if the data is read or accessed by unauthorised personnel, that data cannot be understood without the appropriate key.

Question 5

(a) This was a standard decomposition of a many-to-many relationship into two one-to-many relationships with the use of a linking table and many candidates correctly recognised it as such. A small number of answers showed the one-to-many relationships the wrong way around.

(b) This question proved very challenging. Very few answers were applied to the situation modelled in the database. Answers were mostly either a generic definition of a primary key such as, '*both together uniquely identify a record*', or a repetition of the description of a composite key given in the question, such as '*it is a composite key because it is made up of two foreign keys*'.

(c) There were some excellent complete answers to this question, although some omitted the quotation marks on the last answer.

(d) Many answers correctly completed the `SELECT ... FROM` clause. There were also a number of correct `WHERE` clauses, with a variety of different correct date formats. `WHERE ReleaseDate = "January 2022"` was popular. The `COUNT` construction was less well completed.

(e) This question was answered very well. Many correctly completed at least four of the spaces, Care must be taken when copying from the question, field names and primary keys are both plural in the given list. The statement about the logical schema was the one most likely to be completed incorrectly.

Question 6

(a) The question asked for the operation of a compiler. Many answers included benefits of the translator that are not part of its operation. The use of language is important here too. Vague statements such as, '*the compiler **compiles** the code...*' are not enough, it needs a different word to convey the idea of translation. While many answers stated that an executable file would be produced, very few included the caveat that this was done only when the source code was free from errors.

(b) Here too, the question asked for the operation of an interpreter and many answers included benefits of the software that are not part of its operation. Some responses correctly stated that each line is translated and then executed immediately. It is the immediate execution of each translated line of code that is important.

(c) The question starts with the command word 'Explain'. This means that more than just a list of features is required. Many candidates were able to describe suitable features of an IDE used when writing a program. Describing appropriate features for testing was more challenging. There was some confusion between the removal of syntax errors and testing the logic of the program and ensuring that the program produced the correct results.

Question 7

This question was answered very well. Almost all candidates completed the truth table correctly.

Question 8

There were some excellent, imaginative answers to this question. Self-driving cars and playing chess against the computer were popular choices and descriptions were often very detailed and complete. Responses

should not use brand names in their answer, but use statements like, '*a virtual assistant, such as Siri*' instead. Candidates need to take care when reading the question. This question asks for a description of just one application, not a list.

Question 9

(a) Many answers correctly identified the two missing devices. Describing a WNIC and a WAP proved to be far more challenging. Answers such as, '*a WNIC acts like a NIC in a wireless network*' or '*a WAP provides a wireless access point*' are not enough to describe the devices. The first statement needs a description of the role of the NIC and the second statement is simply an expansion of the acronym. Many answers were given in terms of connection solely to the internet, which was not what the question asked.

(b) This question was answered very well. Many responses were able to give three appropriate differences between the two types of cable. Candidates need to understand that if a question asks for three differences, and when the answer space is numbered correspondingly, only three answers will be marked.

(c) There were a few very good answers to this question. Candidates need to understand that CSMA/CD is a protocol, a set of rules, and as such the protocol cannot 'do' anything. It is the workstation that monitors the communication channel and the workstation that sends a jamming signal, not the protocol.

COMPUTER SCIENCE

Paper 9618/13
Theory Fundamentals

Key messages

There is a greater emphasis on the application of knowledge in the 9618 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.

Candidates must clearly understand the different command words used in the questions. A question that begins with the command word 'Explain' requires a different type of answer to one that begins with the keyword 'State'.

This is a technical subject and at this level of study it is expected that answers will be given in technical terms. Use of the correct terminology is very important.

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.

General comments

If a question asks for a given number of answers, for example, 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 logic expressions were usually completed successfully, the questions about cloud computing 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) (i) The one-to-one correspondence between a character and it's numeric equivalent was often described well. Scripts found achieving the second mark more challenging. Vague statements such as, '*a character set is a set of characters*', is insufficient for credit.

(ii) This question was answered very well. Many were able to correctly name two character sets and give an appropriate difference between them. ASCII and Unicode were the most popular choices.

(iii) The question asked for a description of lossless compression applied to a text file. Some answers did not read the question carefully and gave generic answers about lossless compression, others wrote about compressing an image file.

(iv) Again, most responses stated that lossy compression of a text file would result in corruption of the file, but found achieving the second mark more challenging. Candidates need to be clear that if there are two marks allocated to an answer, then two distinct points should be made.

(b) This question was answered quite well. Candidates usually scored at least three or four marks. Some candidates gave the answer to the second statement as 8-bit binary rather than denary. A

common incorrect answer for the third statement was 0101 0111, where candidates had converted the denary value 87 to 8-bit binary rather than BCD.

Question 2

(a) There were some very good answers to this question. Candidates should be clear on the use of registers in the Fetch-Execute cycle. Registers are simply storage locations; it is the contents of the registers that changes not the registers themselves.

(b) This question proved to be challenging. Few responses mentioned the need to check the priority of the interrupt, and that for the interrupt to be serviced its priority needed to be higher than the priority of the current process. There was considerable confusion about the use of the Interrupt Service Routines (Interrupt Handlers) with answers often suggesting that a single ISR would be used no matter what the cause of the interrupt.

Question 3

(a) This question was answered very well. Most of the candidates were able to give the correct accumulator values after each instruction had been carried out.

(b) Some responses struggled to naming two other instruction groups, descriptions were often much better. The names that would be expected are those in ***section 4.2*** of the syllabus.

(c) This question was very well answered. Most were able to identify the correct result of each instruction.

Question 4

(a) The command word in this question was 'Describe'. A list of management tasks is not enough for a description. Vague statements such as, '*the input/output management task manages input and output*' are insufficient for credit at this level. Descriptions should demonstrate some technical knowledge.

(b) (i) There were some very good answers to this question. Candidates need to be aware that answers that return the wording in the question are not enough. For example, '*backup software backs up the data...*' does not describe what the software does. Candidates should also have a firm understanding of defragmentation. It is the scattered fragments of individual files that are made contiguous.

(ii) This question was very well answered, almost all correctly named another utility software. Hard disk formatting program was a popular choice.

Question 5

(a) (i) This question was also very well answered, almost all were able state a correct reason why data needs to be kept secure.

(ii) There were some good answers to this question. Candidates should be aware that more than a single word answer is required. For example, anti-virus software needs to be installed, switched on and kept up to date.

(b) (i) Some respondents found remembering the names of the methods challenging. There were some good descriptions, although the distinction between manual and system checking was less well described. Candidates must read the question carefully. The question asked about verification when transferring data from a paper document; quite a few answers described verification of data during transfer across a network.

(ii) This question was another of those set in the context of a date of birth and answers were expected to be applied to that context. Weaker answers gave generic descriptions of the three validation checks, with no reference to how they could be applied to a date of birth. The question also asked for an example, only the better answers included this. An example of a good answer is, '*Range check – this would be used to check that the month in the date of birth was a number between 1 and 12 inclusive*'.

(iii) This question proved to be challenging. Answers frequently described validation and verification in generic terms, without explaining why the data might still be incorrect even after validating and verifying it. Vague statements such as, '*the data might be incorrect because the person lied on their application form*' do not explain why this was not picked up in the validation and verification process.

Question 6

(a) Many answers were able to achieve two marks by identifying the primary and foreign keys and describing how they were used to link the tables. Further description of the relationship was more challenging. Again, it required the application of the theoretical knowledge to a particular situation which some responses seemed to find much more demanding.

(b) (i) While a small number of answers were completely correct answers, many found it to be more challenging, especially the syntax for the joining of the two tables. The AND clause was usually correct. All candidates need to understand that quotation marks are required around text constants.

(ii) There were a small number of correct answers to this question. Most found writing the SQL script challenging and some need to improve their understanding of SQL. The best way to do this is to practice setting up a simple multi-table database and then using SQL to manipulate the data contained therein.

(c) The question asked for a description of each of the three Normal Forms and the strongest answers displayed knowledge of the theory and gave correct descriptions. A small number of responses described how to get from one Normal Form to the next, which was not what the question asked.

Question 7

(a) This question was answered very well. A small number of answers confused the XOR and NOR operations.

(b) This question was also answered very well. Almost all were able to complete the truth table correctly.

Question 8

(a) Many found defining cloud computing very challenging and many of the answers given could apply equally to any general client-server configuration. Candidates need to understand that answers which repeat the original wording, such as, '*cloud computing is computing the cloud*', will not be given credit.

(b) Many of the answers to this question did not make the distinction between the cloud, that is the computer services provided, and the data in the cloud. Descriptions of a public cloud frequently stated incorrectly that the data in the cloud was accessible to anyone. Descriptions of a private cloud frequently stated incorrectly that the data in the cloud was only accessible to a single person. As in **Question 8(a)** answers such as 'a *public cloud can be used by the public*' will not be given credit at this level.

(c) Lack of internet access was the most common correct disadvantage seen. Giving two advantages of cloud computing was more challenging. Many of the responses were broad statements that could have applied equally to general storage and computer usage rather than specifically to cloud computing. There was a frequent misunderstanding that data in the cloud is automatically less secure and little realisation that security, backup etc. are dependent on the provider of the cloud services and the quality will thus vary accordingly.

COMPUTER SCIENCE

Paper 9618/21
Fundamental Problem-solving and Programming Skills

Key messages

This paper addresses the application of practical skills including both computational and algorithmic thinking. This often involves analysing and understanding the requirements of a scenario as well as designing and presenting a solution. Candidates need to be able to identify the key elements of each requirement which, for example, could include the need for an iterative structure or methods to access data stored in a string or a file. The development of these skills requires practice.

This subject makes use of many technical words and phrases. These have specific, defined meanings and they need to be used correctly.

Answers should be as precise and specific as possible. 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 particular question shares some key phrases with those from a past paper, the required answer is the same. Candidates should also be aware that answering a question by repeating phrases from the question will not gain marks.

Candidates should be encouraged to attempt all questions. Even the more advanced questions that ask for an algorithm to be written in pseudocode contain accessible marks.

General comments

If answers are crossed out, the new answers must be written clearly so that the text may be read easily and the correct mark awarded. A significant number of candidates attempted to squeeze answers in at the edge of the answer lines making these difficult to read. Candidates should be encouraged to use a continuation booklet if required.

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.

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 often replaced parameters to a procedure or function with a series of prompts and inputs within the body of the subroutine.

The functions and operators that are available for use in pseudocode answers are described in the Insert which accompanies the paper. 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 following examples illustrate invalid pseudocode constructs of the kind that have been seen in this series:

- An invalid Boolean statement:
  ```
  WHILE Number >= 100 AND <= 200
  ```
- Invalid loop constructs:
  ```
  FOR Index ← 1 TO EOF(Filename)
  ```
- Omitting the assignment operator:
  ```
  FOR X 1 TO 100
  ```

- A language specific statement:
  `FOR WebName in Secret`

Comments on specific questions

It is recommended that the following specific comments be read in conjunction with the published mark scheme for this paper.

Question 1

(a) The number of correct answers produced by candidates was small and a range of computing terms often relating to program design were seen.

(b) (i) Many candidates gained full marks and most gained at least two marks.

A common mistake was not reading the information about the use of `MyDOB` variable carefully enough and answering as if a variable was being used 'To **store** the numbers of days until my next birthday' rather than 'To **calculate** the number of days until my next birthday'.

(ii) Many correct answers were seen, showing a good understanding of the use of Boolean variables.

(c) Most candidates performed well on this part.

Some candidates did not use quotes to indicate a string value in the second row of the table but then correctly used quotes for the fourth row.

Question 2

(a) Many candidates gained full marks for this question.

A substantial number of candidates just gave the answer 'Y' for the third row of the table instead of the correct answer 'Button-Y'.

(b) Many candidates obtained full marks for this question.

A common mistake was to replicate the output of 'none' given in two of the rows to the other two rows.

Question 3

(a) Many candidates performed well on this question.

Some answers seen were often too vague or restated 'To allow users to book seats' which was taken from the question.

Good answers often went through the process of selecting a film, booking specific seats for a screening of that film and then paying for the booking made.

(b) Although the question specifically indicated that there are only two specific program modules (functions and procedures) this question was challenging for candidates with a range of general computing terms given as answers.

Question 4

Abstract data types are one of the topics covered on paper 2 for the new 9618 series that was not covered on paper 2 of the old 9608 series.

(a) The majority of candidates gained at least one of the two marks available for this question with many gaining both marks.

A common mistake was for candidates to answer with a value rather than a memory address on the second row of the table.

(b) Only a few candidates gained all four marks for this question.

The mark that was rarely given was for drawing the pointer correctly (MP 1) with most candidates not attempting to add a pointer.

Question 5

Only a small number of comprehensive answers were seen.

Where marks were gained, this was usually for MP 1, 2, 3, 4 and 6.

The explanation of how to extract a field from a line read from a file was rarely detailed enough to be awarded MP5 with very few mentioning the use of the MID or other substring functions.

Only a few candidates gained MP8 for stating the file had to be closed.

Many answers mentioned searching for a value or finding a value but did not approach the level of detail required when describing an algorithm.

Several candidates provided an algorithm written in pseudocode.

Question 6

(a) It was pleasing to see a range of different solutions gaining full marks for this question. Two of the common approaches gaining full marks were those included in the mark scheme as example solutions. Another common approach gaining full marks was to convert the number to a string then extract and test the right most character.

Many candidates recognised the need for a loop going between 100 and 200 but some lost this mark point (MP2) for not using the correct syntax for a loop.

Many candidates gained MP1, but a common reason for this mark being lost was for not initialising the counter variable.

On many occasions MP4 was lost because the candidate did not output the required value, although they had correctly identified this as ending in a '7'.

(b) Candidates found this part very challenging.

Full marks were given for candidates using nested `IF THEN ELSE` constructs or `IF ELSEIF` constructs. A few candidates produced solutions using just `IF` statements that gained full marks.

Question 7

(a) A small number of very good solutions were seen.

Many candidates did not create the function header correctly, which is a fundamental requirement for this algorithmic type question, and so did not achieve MP1.

Most candidates attempted to create a reverse string, but many did not initialise the new string being created.

Two different approaches were seen when candidates attempted to access MP5. Some candidates converted the original string to one of the same case, while others converted the case of each character one at a time. Both approaches, when done correctly, were creditworthy.

(b) Many candidates found this part challenging.

The skills being assessed are similar to those required when drawing a flowchart, but some candidates seemed to find this alternative way of accessing these skills more challenging.

A common mistake was treating strings in the same way as arrays rather than using the appropriate substring function such as `MID`.

Question 8

(a) This question proved to be challenging to most candidates with only a handful of good solutions seen.

Few candidates used the `RAND` function correctly. Candidates are advised to consult the Insert provided in this regard. A common mistake was to pass the range of values required as an argument to the `RAND` function. Examples of the erroneous use of the `RAND` function seen were `RAND(97 – 122)` and `RAND(97,122)`.

Another common mistake with generating random integer values was not realising that `Y ← RAND(X)` where `X` is an integer, produces a real number with the value assigned to `Y` in the range `0 <= Y < X`.

Furthermore, many candidate solutions did not produce an integer value in the required range; the `INT` function needs to be used along with the `RAND` function.

(b) A few very good solutions were seen for this more unusual linear search question.

Most solutions used a `FOR` loop rather a conditional loop which meant that MP2 could not be awarded as the loop did not terminate when a match was found.

Many candidates did not gain MP6 as the output was included in the loop meaning that the warning message was output each time a match did not occur. Even when the warning message occurred after the loop in a correctly formed selection statement, MP6 could not be awarded because the Boolean variable had not been initialled before the loop.

(c) Few candidates identified the correct testing method. Common wrong answers included white box testing, acceptance testing and black box testing. Often, descriptions offered by candidates did not cover both mark points in the required detail.

(d) When attempted most candidates gained one of the two marks available.

One mark was available for giving an example of a password that would be acceptable such as the one given in the stem at the start of **Question 8**.

When candidates attempted to give two test strings of passwords that would break a validation rule many did not gain credit for the second password example as it broke the same validation rule as the first example. Each of the example passwords given had to contain a different error.

(e) **Candidates found this part challenging.**

Many candidates repeated the question or restated the question rather than describing an approach that could be taken. Often the answers were vague and lacked the detail required that would allow a solution to be produced based on the description given.

COMPUTER SCIENCE

Paper 9618/22
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. Questions often make use of a scenario description to present the requirements. Candidates need to be able to identify the key elements of each requirement (for example, the need for an iterative structure) when designing their solution.

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

Answers should be as precise and specific as possible. 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 particular question shares some key phrases with those from a past paper, the required answer is the same. Candidates should also be aware that answering a question by repeating phrases from the question will not gain marks.

Candidates should be encouraged to attempt all questions. Even the more advanced questions that ask for an algorithm to be written in pseudocode usually contain accessible marks.

General comments

If answers are crossed out, the new answers must be written clearly so that the text may be read easily and the correct mark awarded.

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 when the answer has been written completely in the relevant space on the question paper to avoid ambiguity.

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 often replaced parameters to a procedure or function with a series of prompts and inputs within the body of the subroutine.

The functions and operators that are available for use in pseudocode answers are described in the Insert which accompanies the paper. 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 following examples illustrate invalid pseudocode constructs of the kind that have been seen in this series:

- An invalid boolean statement:
  ```
  IF x = 3 OR 4 THEN
  ```

- Invalid loop constructs:
  ```
  FOR Index ← 1 TO EOF(Filename)
  ```

- The invalid use of the logical operator AND
  OUTPUT Staff(Index). EmployeeNumber AND Staff(Index)Name

- Count-controlled loops where the assignment operator is omitted:
  `FOR X 1 TO 100`

Comments on specific questions

It is recommended that the following specific comments be read in conjunction with the published mark scheme for this paper.

Question 1

(a) The number of correct answers seen was few and a range of general computing terms were seen.

(b) Many candidates gained full marks for this part. Some candidates suggested a series of numbers to indicate specific stages, but this did not provide sufficient detail.

A small number of candidates suggested unrelated computing terms.

(c) Candidates found this part challenging. Where marks were awarded, this was usually for the first two answers on the mark scheme.

General computing terms were widely seen and many candidates suggested 'identifier names' despite the question asking for **other** pieces of information.

A small number of answers related to the use of databases.

(d) Full marks were awarded in many cases.

Some candidates offered a correction to each statement rather than stating the error.

A common mistake for the first row was to state that 'status cannot be both true and false'

A number of candidates suggested that 'the value must not be negative' for the final row; the example given in the Insert showed a negative value.

Question 2

Most candidates seemed to have some experience of producing a program flowchart but the number of completely correct solutions seen was extremely small.

Most solutions included some sort of loop.

Common algorithm mistakes included:

- Not initialising a count variable or initialising it *inside* the loop.
- Attempting to use the `IsPrime()` function without assigning the return value.
- Not including any sort of separator in the output statement, or including an unsuitable message, often relating to the number input rather than the count.
- Confusing the number input with the count.
- Testing for `num > 0` rather than num `< 0`.
- Treating the return value from `IsPrime()` as non-Boolean.

Generic program flowchart errors commonly seen included:

- Outputs from flowchart boxes going nowhere or several lines joining together combining inputs and outputs to give a meaningless 'flow'.
- Decision diamonds with only a single output, or unlabelled outputs.
- Attempting to include a large part of the algorithm within a single flowchart box. For example 'count the number of times the number is prime'.

Question 3

(a) (i) This question proved very challenging for most candidates.

Some candidates gained credit for the two parameters. Many candidates added labels to the existing parameters, despite the question directing them otherwise.

(ii) Many correct answers were seen, suggesting that candidates were able to map the module headers to the original diagram.

(b) Many candidates gained marks for 'easier to...' answers relating either to the problem or the implementation of a solution. The ability to assign problems to different teams was also commonly seen.

Many answers were vague and did not relate sufficiently to the problem or to the sub-problems. Candidates should familiarise themselves with the meaning of the command words used in the questions and form their answers accordingly.

A small number of candidates gave benefits of structure charts, rather than benefits of decomposition. Use of vague terminology was commonplace, for example, 'it makes testing more convenient'.

Question 4

(a) Mark points MP1 and MP3 were the most accessible and were given in many cases. Many candidates did not gain MP2 because the `CLOSEFILE` statement was either omitted or lacked the filename. Others chose to open the file in something other than `READ` mode.

Many candidates attempted to give the parameter an invalid name such as `Thisfile.txt`. Solutions often incorrectly placed the parameter name in quotation marks when used in subsequent statements and some appended '`.txt`' to an otherwise valid parameter name.

A significant number of candidates recognised the need to 'shuffle' the three variables and some ingenious solutions were seen. Perhaps the most successful were those that contained three sets of output statements and then chose the appropriate set depending on the number of the last line read.

Many solutions attempted to treat the text file as a random-access file. Statements such as: `LineX ← Myfile.txt(Index)` were commonly seen.

The use of the `EOF()` function was widely misunderstood, as illustrated by the statement:
`FOR Index ← 1 TO EOF() – 3`

Where solutions were based on a 'two-loop' approach, it was very rare for the file to be closed and re-opened between the two loops. Another common error with this type of solution was the failure to actually read lines within the second loop when attempting to get to `Lastline - 3`.

A common error was the attempt to use the `LENGTH()` function to obtain the number of lines in the file.

(b) Candidates found this part challenging.

Where marks were awarded, this was usually for reference to the need for an additional parameter, the introduction of an array to store the lines and the use of a loop for the final output.

Many candidates suggested simply increasing the number of variables, rather than introducing an array.

The description of each change was often vague and imprecise. For example, 'a loop will be used' or 'a count will be used'. Many descriptions included an attempt to input the line number rather than have it passed as a parameter and several candidates referred to 'checking that the file contained enough lines' even though the question stated that this could be assumed.

Many candidates misunderstood the question and suggested that the file name needed to be output.

Question 5

(a) Many candidates correctly identified a pre-condition loop.

Although the question asked for the condition that ends the loop, many answers described the condition for the loop to continue. Several candidates gave a general answer such as 'when the condition is satisfied'.

(b) **T**his part was answered well by many candidates.

Common mistakes were the presence of a final '0' in the `ThisChar` column and additional output values in the `OUTPUT` column. The `Flag`c> value of 'Not Flag' was seen on several occasions.

(c) This part was answered well by some candidates.

Most marks were gained from a reference to 'the modules being put together and tested' but few identified the fact that individual modules would have been first tested individually.

Many answers referred only to the test data and the use of 'integrated test data' was suggested on several occasions.

Confusion between integration testing and other forms of testing (such as stub, black-box or user) was common.

Question 6

A small number of very good solutions were seen.

Most candidates recognised the requirement for a loop. The second mark point (MP2) was the one most commonly awarded.

Many candidates seem unfamiliar with the need to initialise variables and consequently MP1 was often not given.

MP3 was attempted by many, often correctly. Common faults included the use of `LEFT()` rather than `MID()` and the placement of the comma comparison test within a `WHILE` loop rather than an `IF` clause. Many solutions made the mistake of attempting to treat the string as an array.

MP4 was seen correctly only as part of better solutions. A common mistake was to form a total by summing individual digits rather than by extracting the whole number and then converting it to an integer. A very small number of solutions formed a correct total by multiplying the existing total by ten before adding the value of the next digit.

MP5 was also only seen in better solutions. Some solutions used indexing to isolate a number rather than using a substring and, where correct, these gained MP4, MP5 and MP6.

MP6 was awarded quite often. In many cases the conditional clause testing for a comma contained no `ELSE` clause and so this part of the algorithm was omitted.

Many solutions declared the variable `Average` as an integer and so did not gain MP7. Others attempted to use string concatenation on a numeric value without the use of the `STR_TO_NUM()` function.

A small number of solutions were based solely on the example string given in the question.

Question 7

(a) A very small number of good solutions were seen.

A number of solutions gained only the first mark point (MP1). A common mistake was to omit the data types of the parameters in the function header.

Several solutions attempted to subtract one string from another by using the 'minus' operator.

Common faults included:

- Using either the `RIGHT()` or `LEFT()` functions without assigning the return value.
- Treating the string as an array.

(b) Many different types of 'check' were suggested, such as type, parity, presence and length.

Very few answers related to the given scenario and many focused on type checking.

Question 8

(a) Some very good solutions were seen for this linear search question. These were based on both a conditional loop and a count-controlled loop.

A common mistake with conditional loop solutions was to only check a Boolean flag, meaning that the `MID()` function would be called with a parameter value that extended beyond the length of the string.

Several solutions attempted to use `LEFT()` rather than `MID()` to extract an individual character and some made the mistake of converting the characters to a known case before making the comparison.

(b) Some good solutions were seen. Mark points that were commonly awarded were usually for MP1, MP4, and MP8.

Many attempts at the conditional loop (MP2) omitted the check on the array index value. Sometimes this functionality was unsuccessfully attempted by placing a count-controlled loop *within* the conditional loop.

Very few attempts at MP3 were seen.

The use of the `Decrypt()` function was common (MP4) but often the return value was unassigned.

An attempt at MP5 was often seen, but usually this ran from 1 to 500 and so would recognise the original password as a duplicate of itself. In some more competent solutions, the index of the original password was skipped.

Many solutions contained an attempt at MP6. Often this was incomplete or confused the contents of column 1 and column 2.

On detecting a duplicate, many solutions went on to set a flag, but often this did not act as a loop terminator so MP7 was missed.

MP9 was often attempted, but in many cases the logic prior to the output was incorrect.

(c) Only a small number of comprehensive answers were seen.

Where marks were gained, this was usually for MP2, MP3 and MP5.

The description of the use of the `Exist()` function often did not make it clear that a new random character would be generated if the current one already existed in the password.

Many answers did not approach the level of detail necessary to be described as an 'algorithm'.

Some answers addressed the task of validating an existing password, rather that generating a new one.

Several candidates provided an algorithm written in pseudocode.

COMPUTER SCIENCE

Paper 9618/23
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. Questions often make use of a scenario description to present the requirements. Candidates need to be able to identify the key elements of each requirement (for example, the need for an iterative structure) when designing their solution.

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

Answers should be as precise and specific as possible. 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 simply because a particular question shares some key phrases with those from a past paper that the required answer is the same. Candidates should also be aware that answering a question by repeating phrases from the question will not gain marks.

Candidates should be encouraged to attempt all questions. Even the more advanced questions that ask for an algorithm to be written in pseudocode usually contain accessible marks.

General comments

If answers are crossed out, the new answers must be written clearly so that the text may be read easily and the correct mark awarded.

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.

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.

The functions and operators that are available for use in pseudocode answers are described in the Insert which accompanies the paper. 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 following examples illustrate invalid pseudocode constructs of the kind that have been seen in this series and which resulted in marks being lost:

- An invalid Boolean statement:
```
IF x = 3 OR 4 THEN
```

- Invalid loop constructs:
```
FOR Index ← 1 TO EOF(Filename)
```

- The invalid use of the logical operator `AND`:
```
OUTPUT Staff[Index].EmployeeNumber AND Staff[Index].Name
```

- Count-controlled loops where the assignment operator is omitted:
  `FOR X 1 TO 100`

Comments on specific questions

It is recommended that the following specific comments be read in conjunction with the published mark scheme for this paper.

Question 1

(a) This part was very well answered. Most candidates gained at least 3 marks and many full mark answers were seen.

(b) (i) This part was very well answered. Many candidates gained full marks.

(b) (ii) This part was answered reasonably well. Where marks were awarded, this was usually for an example such as 'to tell if a parcel has arrived or not'. Many answered were vague and not sufficient to gain marks, such as 'to control the temperature of a pool'.

Question 2

The majority of candidates correctly identified corrective and adaptive maintenance, and many went on to provide an adequate reason for corrective maintenance.

The reason given for adaptive maintenance was often too vague for a mark to be awarded.

Several candidates used unrelated computing terms, and a small number of candidates did not attempt this question.

Question 3

(a) Many fully correct answers were seen, usually quoting the relationship between modules and the parameters that passed between them.

(b) This part was well-answered by the majority of candidates, with nearly all diagrams being recognisable as structure charts.

Common mistakes were the omission of the decision diamond and treating the ByRef parameter as a separate parameter plus a separate return value.

A very small number of candidates gave no answer or provided a diagram that was more akin to a program flowchart.

Question 4

Some very good answers were seen, correctly testing boundary values as well as abnormal values.

Some candidates made errors by repeating normal test data values within the range 151 to 154.

Several answers missed that the question specified that the value was an integer value and suggested real numbers, strings or in some cases Boolean test values.

Question 5

(a) (i) This part was well-answered by the majority of candidates.

Common mistakes included omitting the closing `ENDTYPE` or the individual `DECLARE` keywords. A small number replaced the `TYPE` clause with a procedure declaration.

The data types were normally correct.

Some candidates attempted to apply the information from the comment as part of the declaration. For example:

```
DECLARE EmployeeNumber : INTEGER > = 1
```

A small number made no attempt at this question.

(ii) Most candidates were able to declare an array of 500 elements, but only around half of these were of the correct record type, the others usually being declared as type `STRING`.

(b) (i) This part was generally not answered well.

Most candidates did not make the connection between a known way of indicating an unused element and a subsequent processing activity such as searching. Many attempted to describe an advantage to the *user* and the suggestions that it would 'save space' or 'save time' were commonly seen.

Answers that gained this mark usually referred to a processing activity such 'finding an available element' or 'filtering out unused elements'.

A small number of candidates made no attempt at this question.

(ii) A lack of precision was widespread, and very few correct answers were seen. A small number of candidates specified that an individual field should be assigned a particular value. Where this was done, it was most often to assign an empty string to `Name` or zero to `EmployeeNumber`.

The most common mistake was to not take into account that this was an array of records with fields made up of different data types, and to attempt to initialise a complete array element to a value such as an empty string.

A small number did not attempt this question.

(c) This linear search question was well-answered by the majority of candidates.

MP1 and MP2 were commonly given.

In many cases the statements referencing individual elements were ambiguous or incomplete so MP4 was not given, but many solutions benefitted from a follow-through mark for MP5 provided the record number and field were indicated.

Very few solutions addressed MP3.

Question 6

(a) This question drew a range of responses. Some very good solutions were seen with many candidates gaining full marks. However, there were also many poor responses, with candidates making little or no viable attempt at a solution.

Better solutions often gained MP1.

Many candidates recognised the need for a loop, and MP4, MP5 and MP6 were often (although not always) given as a set of three marks.

MP2 was given in most solutions.

MP3 and MP7 often went together and were sometimes missed due to invalid or incomplete `IF ... THEN ... ENDIF` structures. In this respect those solutions that implemented immediate `RETURN` statements within separate `IF ... ENDIF` clauses were more likely to gain these marks.

(b) As for the previous part, some very good answers were seen, with most candidates attempting this question. However, a significant number of answers gained one mark or less.

Common mistakes included:

- Inverting the test in label A but not correctly.
- Not assigning the return value from `Factorial()` in label D.
- Specifying an output that did not match that given in the question.

Question 7

(a) (i) Many candidates correctly identified the potential sequence error.

Common mistakes included:

- Suggesting that the first three lines would be output rather than the last three.
- Suggesting that some of the lines would be blank.
- Describing various file-related issues contrary to the wording of the question.

(ii) Many weak responses were seen to this question part. Many candidates found describing an algorithm particularly challenging. Descriptions were generally vague and did not approach the level of detail necessary to be described as an 'algorithm'.

For the 'Description' part, many answers only provide a re-worded version of the answer given to **7(a)(i)**.

Marks for the 'Explanation' part were usually given for a description of a 'shuffle' operation or a way of counting the number of lines in the file.

(b) This part was correctly answered by many candidates. Some responses suggested that the file should contain 'exactly three lines' but this was not a complete enough answer.

(c) A challenging question that was correctly answered by only a very small number of candidates.

Many solutions did not satisfy the requirement for a 'single pseudocode statement'.

Many candidates made no attempt at this question.

Question 8

(a) Many candidates gained full marks for this question. Few candidates gained MP3.

A number of candidates produced a diagram which did not have the links between stages, and several made no attempt at the question.

(b) Very few marks were awarded for this question.

Many answers referred simply to 'pointing to the next/previous stage' without making any reference to anything being passed between the stages.

Where a mark was awarded, it was usually for the description of the upward arrow, where the answer referred to an error being found which needed to be addressed by the previous stage.

(c) Well answered by the majority of candidates, with RAD being the most common answer.

A small number of candidates did not attempt this question.

Question 9

(a) As for **Question 6 (a)**, this question attracted a differing range of responses and a wide range of marks were awarded.

A small number of excellent solutions were seen.

Many poor responses were seen, with some making little or no viable attempt at a solution.

Some solutions attempted to validate rather than generate a password.

Mark points MP1, MP2 and MP4 were the most accessible and were given in many cases. Most candidates recognised the need for a loop structure and in many cases, this was nested correctly.

MP3 was often attempted, using the two modules as described. On many occasions the return values from the functions were unallocated.

MP5 was usually attempted, often in conjunction with MP4 where a conditional loop was implemented which repeated until the length of the group was 4 characters.

MP6 was a straightforward mark and was commonly given provided the character had been identified as being unique. An occasional mistake was the use of the '+' operator rather than the concatenation operator '&'.

MP7 and MP8 were usually given in better solutions. A small number of solutions attempted to output the password rather than return it.

(b) Some good answers were seen, but generally not as good as the solutions seen for **Question 9 (a)**.

Many poor responses were seen, with some making little or no viable attempt at a solution.

For the first mark point (MP1), many solutions did not use the `FindPassword()` module but instead performed their own linear search. If this was correct then the mark was awarded.

Most candidates recognised the need for a loop and MP2 was addressed in many solutions. A common mistake was to attempt to use the `LENGTH()` function to find the number of elements in the array.

A number of solutions attempted to combine MP2 and MP3 into a single conditional loop. These were usually not successful.

Most reasonable solutions addressed MP4, with many of these recognising the need to use the `Encrypt()` function and so also gained MP5.

MP6 proved challenging. Many solutions did not cater for the case when the password was not already in the array, but there was no room to add it.

Several solutions were incorrectly based around the example values given in the question.

(c) Many answers gained one mark for reference to the use of a separator character. Few of these specified that this character was one that did not appear in the domain name (as any character could appear in the encrypted password) and consequently were also vague regarding the order of concatenation.

A small number of solutions were based on calculating the length of one of the strings and appending this value (in string form) to the string formed by concatenating the name and password.

COMPUTER SCIENCE

Paper 9618/31
Advanced Theory

Key messages

Candidates needed to show an in-depth study of the syllabus topics and make good use of the appropriate technical terminology required to answer questions on this paper. Candidates who had studied the theory and had also practised the precise use of these tools and techniques were able to demonstrate successfully how they could be used to solve the problems set on the examination paper.

Candidates need to ensure that they provide the information asked for in the question set. It is not enough to rewrite the information given in the question as an answer.

Candidates need to have practical experience of programming to aid their understanding of programming topics.

General comments

Candidates need to answer examination paper questions carefully, using appropriate terminology and syntax where necessary.

For example, answers to **Question 2(a)** and **2(c)** must use correct declarative language syntax and answers to **Question 9(b)** must use correct assembly language syntax. In **Question 5(a)** Reverse Polish Notation (RPN) is required.

Candidates need to have practical experience of programming to aid their knowledge and understanding of data types for **Question 1**, declarative programming for **Question 2** and Object-Oriented Programming (OOP) for **Question 7**.

Comments on specific questions

Question 1

(a) Most candidates wrote the two required assignment statements. A common error seen was not to include quotation marks around the string value.

(b) (i) Many candidates correctly included the range of values required in their declaration. A common error seen was to declare an array.

(ii) Better responses included an array declaration of the correct size and data type. A common error seen was to not declare an array.

(c) Better responses explained the meaning of a composite data type and included an appropriate example. A common error seen was to give a general answer repeating the words given in the question.

Question 2

(a) Most candidates knew how to write the additional clauses. Better responses were syntactically correct.

(b) Nearly all candidates wrote a correct result.

(c) (i) Most candidates knew how to write the goal. Better responses were syntactically correct.

(ii) Many candidates knew how to write the two parts of the goal. Many responses seen did not include the appropriate joining syntax for the two parts.

Question 3

Most responses included descriptions of both methods of data transmission. Better responses included different advantages and disadvantages as required by the question. A common error seen was to give the reverse of the advantage as a disadvantage.

Question 4

(a) Better responses included a description for each type of processor. A common error seen was to compare the types of processor.

(b) This question part was generally well answered.

Question 5

(a) This question part was generally, well answered.

(b) (i) Many candidates showed the changing contents of the stack accurately. A common error seen was to include the operators on the stack.

(ii) Most candidates could describe the main steps in the evaluation of RPN using a stack. Better responses correctly described the evaluation of the actual expression shown in **part (b)(i)**.

(c) The full range of marks was seen, popular correct answers included recursion and use of procedures.

Question 6

Most candidates successfully identified at least one benefit or limitation of using a virtual machine. Better responses included good descriptions of each benefit and limitation.

Question 7

(a) Most candidates successfully showed at least one property, with many candidates also including the correct method for both classes. Showing inheritance proved challenging for many candidates, common errors included missing lines, missing arrows or incorrect arrows.

(b) This question part required candidates to give clear descriptions of the Object-Oriented Programming (OOP) terms in the question using appropriate OOP terminology. Some excellent responses were seen.

Question 8

(a) (i) Better responses clearly gave reasons that related to the use of cryptography for the electronic transfer of messages. An example of a valid reason is '*to ensure that the message can only be read and understood by the intended receiver of the message*'. Common errors included answers that were too vague such as '*prevents hacking*' or '*others cannot read the message*'.

(ii) This question was generally well answered.

(b) (i) Many candidates gave a possible benefit, few candidates gave two correct benefits. A popular correct answer was '*eavesdropping can be identified as the state will change*'. A common error seen was the inaccurate answer '*prevents eavesdropping*'.

(ii) Some candidates gave a possible drawback. A popular correct answer was '*a limited range for message transmission*'. A common answer seen that was too vague to be creditworthy was '*too expensive*'.

Question 9

(a) Direct and indirect addressing modes were identified by many candidates. Fewer candidates successfully identified immediate addressing. The contents of the accumulator after each instruction were identified by most candidates.

(b) The full range of marks was seen for this question. Most candidates wrote one or more correct assembly language instructions. Good responses included labelled addresses for the storage of the constant and the two variables, `Y` and `Z`.

COMPUTER SCIENCE

Paper 9618/32
Advanced Theory

Key messages

Candidates are required to demonstrate an in-depth study of the syllabus topics and make good use of the appropriate technical terminology required to answer questions on this paper. Those candidates who have thoroughly studied the theory and have also practised the precise use of these tools and techniques, were able to demonstrate successfully how they could be used to solve the problems set on the examination paper.

Candidates should be encouraged to provide the information asked for in the question set. It is not enough to rewrite the information given in the question as an answer.

It is also essential that candidates have practical experience of programming to aid their understanding of programming topics.

General comments

Candidates will benefit from answering questions carefully, using appropriate terminology and syntax where necessary.

For example, in **Question 1(c)(ii)** the answer must use appropriate Object-Oriented Programming (OOP) terminology.
Answers to **Question 2(a)** and **2(c)** must use correct declarative language syntax.
In **Question 4(b)** and **4(c)(ii)** Backus-Naur Form (BNF) is required.

Practical experience of programming is vital, and candidates should use this knowledge to aid their understanding of data types for **Question 1**, declarative programming for **Question 2** and exception handling for **Question 9**.

Comments on specific questions

Question 1

(a) Most candidates wrote the two required assignment statements. A common error seen was not to include quotation marks around the string value.

(b) (i) Many candidates correctly declared the enumerated type required. A common error seen was the incorrect use of `DECLARE` instead of `TYPE`.

(ii) Most candidates wrote the correct declaration.

(iii) Many candidates correctly wrote a revised assignment statement.

(c) (i) Most candidates wrote the correct private declaration.

(ii) Appropriate computer science terminology was necessary for this question. For example, 'To enforce encapsulation, ensuring that the attributes are hidden and can only be accesses by the class's own methods.' A common error seen was to give a general answer concerning data security.

Question 2

(a) Most candidates knew how to write the additional clauses. Correct syntax was important in this question.

(b) Nearly all candidates wrote a correct result.

(c) Performance in this question was varied. Many responses did not include the appropriate variables and were syntactically incorrect.

Question 3

(a) Most candidates correctly identified two protocols. Stronger responses included detailed descriptions of use.

(b) Most candidates correctly identified two other layers. Stronger responses included detailed descriptions of the function of each layer identified.

Question 4

(a) Generally, this question was well answered.

(b) Generally, this question was well answered. A common error seen was to omit the underscore in `<unsigned_integer>`.

(c) (i) Performance in this question was varied. Good understanding of the use of syntax diagrams was important in answering this question.

(ii) Performance in this question was varied. There were several methods of writing a correct response in BNF.

Question 5

Most candidates correctly identified the different categories of computer architecture. Providing acceptable descriptions for each category of computer architecture was more challenging for many candidates.

Question 6

(a) Most candidates successfully completed the truth table for the logic circuit shown.

(b) (i) Nearly all candidates named the logic circuit correctly.

(ii) Most candidates correctly stated the purpose of both outputs.

Question 7

(a) Performance in this question was varied. Stronger responses included clear statements setting out the process of obtaining a digital certificate. For example, '*To obtain a digital certificate an enquiry is made to the Certificate Authority (CA), the enquirer's details are checked and validated by the CA, before issuing the certificate, which contains the enquirer's public key*.' A common error was to describe the Secure Socket Layer authentication process.

(b) (i) Many candidates gave a clear explanation of the process of producing a digital signature. A common error was to mix up the terms digital signature and digital certificate.

(ii) Stronger responses included a clear explanation of how a received digital signature could be checked to prove that the message had not been altered during transmission. Any keys used during the process needed to be identified exactly.

Question 8

(a) This question was well answered. Common errors included setting an incorrect value, often one too large, for `MaxIndex` and including `Index` as part of the output.

(b) (i) The vast majority of candidates correctly identified the necessary condition.

(ii) Many candidates gave an effective description of a binary search. A few candidates incorrectly gave a description of a linear search.

(iii) Successful **c**andidates considered the effect on the time taken to complete a binary search as the number of values in the array increased or decreased.

(c) This question was more challenging, and it was left unanswered by some candidates. Few responses compared the two types of searches in the depth required. Some candidates showed knowledge of Big O notation, but few candidates demonstrated good understanding of the notation.

Question 9

This question proved challenging for most candidates with a small number giving excellent answers showing good, practical understanding of exception handling in programming. A common incorrect answer was to explain interrupt handling.

COMPUTER SCIENCE

Paper 9618/33
Advanced Theory

Key messages

Candidates needed to show an in-depth study of the syllabus topics and make good use of the appropriate technical terminology required to answer questions on this paper. Candidates who had studied the theory and had also practised the precise use of these tools and techniques were able to demonstrate successfully how they could be used to solve the problems set on the examination paper.

Candidates need to ensure that they provide the information asked for in the question set. It is not enough to rewrite the information given in the question as an answer.

Candidates need to have practical experience of programming to aid their understanding of programming topics.

General comments

Candidates need to answer examination paper questions carefully, using appropriate terminology and syntax where necessary.

For example, answers to **Question 2(a)** and **2(c)** must use correct declarative language syntax and answers to **Question 9(b)** must use correct assembly language syntax. In **Question 5(a)** Reverse Polish Notation (RPN) is required.

Candidates need to have practical experience of programming to aid their knowledge and understanding of data types for **Question 1**, declarative programming for **Question 2** and Object-Oriented Programming (OOP) for **Question 7**.

Comments on specific questions

Question 1

(a) Most candidates wrote the two required assignment statements. A common error seen was not to include quotation marks around the string value.

(b) (i) Many candidates correctly included the range of values required in their declaration. A common error seen was to declare an array.

(ii) Better responses included an array declaration of the correct size and data type. A common error seen was to not declare an array.

(c) Better responses explained the meaning of a composite data type and included an appropriate example. A common error seen was to give a general answer repeating the words given in the question.

Question 2

(a) Most candidates knew how to write the additional clauses. Better responses were syntactically correct.

(b) Nearly all candidates wrote a correct result.

(c) (i) Most candidates knew how to write the goal. Better responses were syntactically correct.

(ii) Many candidates knew how to write the two parts of the goal. Many responses seen did not include the appropriate joining syntax for the two parts.

Question 3

Most responses included descriptions of both methods of data transmission. Better responses included different advantages and disadvantages as required by the question. A common error seen was to give the reverse of the advantage as a disadvantage.

Question 4

(a) Better responses included a description for each type of processor. A common error seen was to compare the types of processor.

(b) This question part was generally well answered.

Question 5

(a) This question part was generally, well answered.

(b) (i) Many candidates showed the changing contents of the stack accurately. A common error seen was to include the operators on the stack.

(ii) Most candidates could describe the main steps in the evaluation of RPN using a stack. Better responses correctly described the evaluation of the actual expression shown in **part (b)(i)**.

(c) The full range of marks was seen, popular correct answers included recursion and use of procedures.

Question 6

Most candidates successfully identified at least one benefit or limitation of using a virtual machine. Better responses included good descriptions of each benefit and limitation.

Question 7

(a) Most candidates successfully showed at least one property, with many candidates also including the correct method for both classes. Showing inheritance proved challenging for many candidates, common errors included missing lines, missing arrows or incorrect arrows.

(b) This question part required candidates to give clear descriptions of the Object-Oriented Programming (OOP) terms in the question using appropriate OOP terminology. Some excellent responses were seen.

Question 8

(a) (i) Better responses clearly gave reasons that related to the use of cryptography for the electronic transfer of messages. An example of a valid reason is '*to ensure that the message can only be read and understood by the intended receiver of the message*'. Common errors included answers that were too vague such as '*prevents hacking*' or '*others cannot read the message*'.

(ii) This question was generally well answered.

(b) (i) Many candidates gave a possible benefit, few candidates gave two correct benefits. A popular correct answer was '*eavesdropping can be identified as the state will change*'. A common error seen was the inaccurate answer '*prevents eavesdropping*'.

(ii) Some candidates gave a possible drawback. A popular correct answer was '*a limited range for message transmission*'. A common answer seen that was too vague to be creditworthy was '*too expensive*'.

Question 9

(a) Direct and indirect addressing modes were identified by many candidates. Fewer candidates successfully identified immediate addressing. The contents of the accumulator after each instruction were identified by most candidates.

(b) The full range of marks was seen for this question. Most candidates wrote one or more correct assembly language instructions. Good responses included labelled addresses for the storage of the constant and the two variables, `Y` and `Z`.

COMPUTER SCIENCE

Paper 9618/41
Practical

Key messages

Candidates must make sure their evidence is in the correct space in the evidence document. The code a candidate puts in an answer space for one question is marked, even if they have actually placed their solution to that question part in a different answer space. Candidates should copy their code into the evidence document where possible. If screenshots are used, these should have a white background with black font and must be large enough to be clearly legible. This is the same with screenshots of the output: coloured font on a black background is not always clearly legible and if it cannot be read then it cannot be awarded credit.

Where candidates are unable to provide solutions for earlier parts of a question, they should still attempt the later parts because there are opportunities to gain some marks in most questions without previous working solutions.

When providing evidence of testing, candidates must make sure they include evidence of all data entered and output for that test. This might require multiple screenshots for some responses.

General comments

Candidates need to consider their chosen language and make sure they are converting any pseudocode or algorithms appropriately, making adjustments where needed (for example array access, loop terminators, passing parameters by reference or by value) to allow the solution to work for their languages. Pseudocode and descriptions of algorithms are not written specific to one high-level language and at times VB.NET, Java and Python require different alterations to allow one algorithm to run. This does not mean that candidates should change the functionality.

Comments on specific questions

Question 1

This question required candidates to read data from a file, sort it and store it in a file.

(a) Candidates took a range of approaches to create an appropriate array. Some candidates created a 2D array, some candidates created an array of objects, others an array of records, and some candidates chose to create two separate arrays. Some candidates created the structures but did not define, or identify, the data type of the number of elements in the array.

(b) While many candidates were able to open the text file, fewer also closed the file in an appropriate place. Some candidates made good use of exception handling when opening and reading from the file. There were a range of loops used, including looping twenty times and then calculating which data was being read, reading in two lines ten times, and looping until end of file.

(c) Many candidates were able to access each element within each array independently and output it with the required space between each value. To output this on one line, some candidates assigned the values to a string before outputting and others used appropriate formatting controls or commands to stop a new line being output. Some candidates did not include the required spaces or line breaks, a common error was outputting all the content from the array (or arrays) directly.

(d) (i) This question was often answered well, with candidates accurately calling the two subroutines.

(ii) Many candidates were able to output the required data in the correct format.

(e) (i) Many candidates were able to read in the required data as input, but fewer could fully validate the data. The score was often validated appropriately, but the 3-character name was often checked for whether the length was 3 or less. This meant a 2-character name would be accepted by the algorithm, which did not meet the criteria. When validating the input data, some candidates validated both together. For example, if the name was invalid then both were output as invalid and required re-entry. Although this would meet the criteria for this question, it is important that candidates validate data independently and handle the invalid data suitably as well to make sure the program does not continue to run with invalid data.

(ii) This question required candidates to solve the problem of adding a new high score to the table. Candidates tackled this in a variety of suitable ways including: appending the data and then sorting the new list of eleven items, for example using a bubble sort; using an insertion sort method and moving each item from the end until the position was found and the new data inserted, creating a new list with the data up to the insertion point, inserting the new data and then the remainder. Some candidates did not move all data, especially if they were using two separate arrays, and only one was sorted, missing the name or the score.

(iii) Many candidates were able to call the sorting procedure appropriately, but fewer output the results before and after, often only showing the output after; on some occasions this could have been from candidates not providing all of their code to show where it was called, for example showing the output high score call and missing the previous one. It is important that candidates provide all the code for their solution to each question part. This can include code for other parts as it is preferable for the candidates to include all relevant code than to select small parts that might exclude code required for completeness.

(iv) Many candidates entered the correct data, but fewer produced an output with the data in the correct position. Some candidates did not use the given player name and score; when inputs are provided they need to be used in the output because these are designed to produce a specific output that will demonstrate their algorithms are functional.

(f) Many candidates were able to accurately open the file to write the data, and wrote the data to the file. Some candidates attempted to write eleven values instead of the 10 required for the top ten.

Question 2

This question required candidates to use object-oriented programming to solve the given problem.

(a) Many candidates were able to declare the given class, but fewer used the correct constructor, instead declaring a constructor method and not using their language's constructor method. Candidates were often successful in declaring the attributes as private and with appropriate data types. When using Python, candidates often used appropriate comments to indicate this in their algorithms. When an attribute is assigned a value in the constructor, this should not be included as a parameter. Some candidates passed a new health value as a parameter, which did not meet the requirement as this would allow the value to be different for each object.

(b) Many candidates were able to declare the get method suitably. Some candidates attempted to pass a parameter to the get method and then use this in the method, or they output instead of returning. When using Python, get methods require self to be passed as a parameter.

(c) This question was often answered well, with many candidates adding the parameter value to the attribute. Some candidates replaced the health value with the parameter instead of adding it to the current value. This is a method and therefore in Python self is required to be passed as a parameter. Candidates need to make sure they are meeting the requirements in the question. For example, in this question the value is added and there is no output or return value.

(d) This question was also a method for the class. Many candidates were able to compare the value and return the appropriate value. Some candidates passed a value to the function and then used this value to compare to 0, returning the result of the comparison.

(e) Many candidates were able to enter the defence item and colour appropriately. Fewer were able to instantiate an object of the class using these values, with some candidates hard-coding values into the constructor call. Some candidates successfully declared an instance of the object but did not store this object in a variable.

(f) There was a mix of responses to this question, with some candidates accurately manipulating the balloon object appropriately. Some candidates called methods without reference to the object, instead calling subroutines and passing a balloon object as a parameter. Some candidates accessed the values directly instead of using the methods which were required due to the private attributes. Some candidates did not pass and then return a balloon object. In some answers the parameter was named the same as the class, for example, balloon was passed as an object when balloon was the class identifier, causing a naming violation.

(g) (i) Many candidates were able to call their Defend function, but fewer passed the balloon object as a parameter. Fewer also stored the returned balloon object.

(ii) There were a mix of outputs for this question. Some candidates did not include all of the required outputs, for example missing the output of the defence item, and some candidates did not include the inputs in their evidence to show that the test was using the correct inputs.

Question 3

This question required candidates to create and manipulate a queue data structure using a 1D array.

(a) Candidates were often able to declare the variables appropriately, but some did not initialise them appropriately. The array could allow 10 string items, but some candidates did not include a suitable declaration to show that there were 10 elements in the array. When a list is being used in place of an array it needs to be initialised, or indicated, with the appropriate number of values as per the question requirements.

(b) This question included a pseudocode algorithm for candidates to convert into their chosen language and complete the missing statements. The algorithm included parameters passed by reference. Candidates were required to consider how their chosen language deals with by reference and write the algorithm appropriately. Some solutions were returning multiple values to overwrite those they were called from, while some solutions used global variables. Where global variables are used instead, candidates need to make this clear. Some candidates included the code for these to show where they had used them and some included comments to indicate that they had used global variables. Many candidates copied the code as in the algorithm and did not handle the passing parameters by reference, hence the changes made within the function will not be retained when it returns.

A common error in this algorithm was to store the data in the HeadPointer, or in the NumberOfItems value and not the TailPointer which pointed to the position at the end of the queue.

(c) Candidates found this question challenging. Many candidates followed the provided algorithm for enqueue and accessed the data in the tail pointer, checking this value and changing it, instead of the head pointer. When writing a function that returns a value candidates must make sure that this is the final statement in that part of the algorithm. Some candidates manipulated the head pointer and number of items after a value had been returned.

(d) (i) This question required the main program to read in eleven values from the user. Many candidates completed this, but fewer used each value in the function Enqueue(). Some candidates only input 10 values and did not include the 10^{th}.

(ii) Many candidates produced an output, but few had the correct results. A common error was missing the first dequeue of A, and where candidates had accessed the tail pointer instead of the head pointer K was often output instead. The candidates were required to show all of the inputs to show that they were testing it correctly, some candidates only showed a small number of the inputs (for example J and K) and then the output.

COMPUTER SCIENCE

Paper 9618/42
Practical

Key messages

Candidates must make sure their evidence is in the correct space in the evidence document. The code the candidate puts in an answer space for one question is marked, even if they have actually placed their solution to that question part in a different answer space. Candidates should copy their code into the evidence document where possible. If screenshots are used, these should have a white background with black font and must be large enough to be clearly legible. This is the same with screenshots of the output: coloured font on a black background is not always clearly legible and if it cannot be read then it cannot be awarded credit.

Where candidates are unable to provide solutions for earlier parts of a question, they should still attempt the later parts because there are opportunities to gain some marks in most questions without previous working solutions.

When providing evidence of testing, candidates must make sure they include evidence of all data entered and output for that test. This may sometimes require multiple screenshots.

General comments

Candidates need to consider their chosen language and make sure they are converting any pseudocode or algorithms appropriately, making adjustments where needed (for example, array access, loop terminators, passing parameters by reference or by value) to allow the solution to work for their language. Pseudocode and descriptions of algorithms are not written specific to one high-level language and at times VB.NET, Java and Python would require different alterations to allow one algorithm to run. This does not mean that candidates should change the functionality.

Comments on specific questions

Question 1

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

(a) Candidates were often able to declare an array and a pointer and initialise the pointer to 0.

(b) Some candidates did not provide evidence of declaring a procedure to output the stack elements, instead providing evidence of the algorithm out of context. Many candidates were able to output the stack contents and the stack pointer. Some candidates used the data in the stack to identify where it ended instead of using the pointer. This would not be accurate if the data in the stack is not deleted and instead the pointer is used to indicate how many items are in the stack, and therefore available for output.

(c) Candidates were often able to declare the function with a suitable parameter. Some candidates read a value as input within the function instead of taking the required parameter.

When checking if the stack is full, some candidates checked each element in the array instead of using the stack pointer. This will not work if the data remains when an item is removed, or if the stack is initialised with data.

Candidates need to make sure all code in a function will run, i.e. before a value is returned within the function.

Some candidates did not update the value of the stack pointer in the function.

(d) (i) This question required candidates to write the main program, but some candidates instead, incorrectly, attempted to change their push function. Candidates who wrote the main program often successfully input the required values, although some candidates did not convert this to an integer when required by their chosen language.

(ii) Candidates need to make sure they include all evidence of testing and all results; this includes all inputs and the output messages produced. Some candidates only showed the final stack content and did not show the required data being input.

(e) (i) Candidates needed to work out the value of the stack pointer when the stack is empty. It points to the next space and therefore pointing to index 0 will indicate that the first element is to be used next. Some candidates inaccurately checked –1 and some candidates checked each array element to see whether it contained data which would not work when the data is not removed from the stack or it is initialised to specific values.

When the stack pointer indicates the next empty space, the item to be removed is at the position of stack pointer –1. A common error was accessing the value at stack pointer, or overwriting the data in stack pointer –1 before accessing the data.

(ii) This question required candidates to show the output after the appropriate calls were implemented. When showing the output, candidates must make sure all outputs are displayed to show that all parts of the task have been completed and are working. For example, in this question candidates often missed the inputs of data, so there was no evidence that data had been added to the stack, and only provided a screenshot of the removed data.

Question 2

This question required candidates to use sorting algorithms to manipulate the data in a 2D array.

(a) Some candidates were unable to declare or assign data to a 2D array in their chosen programming language. A common error was declaring a 1D array, or a 2D array with an incorrect number of elements, for example a 2 × 10 array. Some candidates were able to accurately generate a random number in their chosen language.

(b) (i) Candidates were provided with an algorithm in pseudocode that they had to convert into their language. Depending on the language chosen, this involved changing the loop conditions and how the 2D array was accessed. Some candidates were able to accurately convert this code. However, a common error in Python was using incorrect loop ending criteria and attempting to access the 2D array elements using (X,Y) instead of (X)(Y). Candidates should be testing their program after conversion and then making appropriate changes to ensure it works in their language. Candidates should change conditions or elements as required when their chosen language does not follow the same principles as the pseudocode.

(ii) A common error was in the formatting of their output, where candidates displayed all of the data in one line and did not split it into individual rows. Some candidates went further and formatted the output to line-up the data in each column which went beyond the requirements for the question.

(iii) Many candidates produced an output, but fewer had 10 rows and 10 columns of numeric data.

(c) (i) Due to an issue with this question, a discussion took place at the examiners' meeting before marking commenced. The examiners considered the impact on candidates in the light of answers seen. Changes to the marking approach for this question were agreed to ensure that no candidates were disadvantaged by the issue.

Candidates were required to analyse the algorithm, recreate it in their chosen language and complete the gaps. A common error was using the same format for the 2D array as in the pseudocode for the chosen language, for example, in Python not separating the indices or in

VB.NET using square brackets instead of round. When changing a pseudocode algorithm into a language candidates need to consider how the language differs from the pseudocode.

(ii) Please note, that due to the issue with **Question 2 (c)(i)**, full marks were awarded to all candidates for this question to ensure no candidates were disadvantaged.

Question 3

This question focused on object-oriented programming.

(a) Many candidates were able to declare a class in their chosen language. Some candidates attempted to use Constructor instead of their language's built-in constructor. Many candidates were able to set, or indicate, that the attributes were private and of the correct data types.

(b) Some candidates were able to create appropriate get methods. A common error in Python was to miss the required self as a parameter. Some candidates sent a parameter to the get method, while a small number did not then use this attribute. Some candidates incorrectly wrote set methods and assigned a value to the attributes, and another common error was to output the value instead of returning it.

(c) This question required candidates to read in data from a text file and instantiate objects with the data. Many candidates were able to accurately open the text file, and some candidates included appropriate exception handling as well. Fewer candidates also closed the text file.

There were a range of loops used, including looping 60 times and then calculating which data was being read, reading in 2 lines 30 times and looping until end of file. With this loop, many candidates were able to accurately instantiate a Card object and assign the data read from the file.

(d) Candidates had to change their program to be able to store which cards were available and which were chosen. Candidates did this in a range of ways. Some candidates created an array for each card that stored a Boolean value for whether it was used. Some candidates added a third attribute to the Card class to record if it was selected. Some candidates deleted the cards from the array. All of these methods, and others, were appropriate implementations and showed suitable analysis of the problem. Some candidates appeared to be using one of these methods, but did not provide enough evidence to demonstrate how this worked, for example, how an array was initialised.

Some candidates were able to identify that the input between 1 and 30 required 1 subtracted from the value to give an accurate array index.

A common error was returning the card that was selected, instead of the index as per the requirement.

(e) (i) Many candidates were able to declare an array, but fewer declared the array of type Card by declaration, assignment or comment, as appropriate for the chosen language. Many candidates were able to call the function correctly, but fewer stored the return value as an index for a card and then selected this card for the player.

(ii) There were a range of correct outputs depending on how the candidate implemented the chosen card. Some candidates only performed one test and did not show the two different hands being assigned.

COMPUTER SCIENCE

Paper 9618/43
Practical

Key messages

Candidates must make sure their evidence is in the correct space in the evidence document. The code a candidate puts in an answer space for one question is marked, even if they have actually placed their solution to that question part in a different answer space. Candidates should copy their code into the evidence document where possible. If screenshots are used, these should have a white background with black font and must be large enough to be clearly legible. This is the same with screenshots of the output: coloured font on a black background is not always clearly legible and if it cannot be read then it cannot be awarded credit.

Where candidates are unable to provide solutions for earlier parts of a question, they should still attempt the later parts because there are opportunities to gain some marks in most questions without previous working solutions.

When providing evidence of testing, candidates must make sure they include evidence of all data entered and output for that test. This might require multiple screenshots for some responses.

General comments

Candidates need to consider their chosen language and make sure they are converting any pseudocode or algorithms appropriately, making adjustments where needed (for example array access, loop terminators, passing parameters by reference or by value) to allow the solution to work for their languages. Pseudocode and descriptions of algorithms are not written specific to one high-level language and at times VB.NET, Java and Python require different alterations to allow one algorithm to run. This does not mean that candidates should change the functionality.

Comments on specific questions

Question 1

This question required candidates to read data from a file, sort it and store it in a file.

(a) Candidates took a range of approaches to create an appropriate array. Some candidates created a 2D array, some candidates created an array of objects, others an array of records, and some candidates chose to create two separate arrays. Some candidates created the structures but did not define, or identify, the data type of the number of elements in the array.

(b) While many candidates were able to open the text file, fewer also closed the file in an appropriate place. Some candidates made good use of exception handling when opening and reading from the file. There were a range of loops used, including looping twenty times and then calculating which data was being read, reading in two lines ten times, and looping until end of file.

(c) Many candidates were able to access each element within each array independently and output it with the required space between each value. To output this on one line, some candidates assigned the values to a string before outputting and others used appropriate formatting controls or commands to stop a new line being output. Some candidates did not include the required spaces or line breaks, a common error was outputting all the content from the array (or arrays) directly.

(d) (i) This question was often answered well, with candidates accurately calling the two subroutines.

(ii) Many candidates were able to output the required data in the correct format.

(e) (i) Many candidates were able to read in the required data as input, but fewer could fully validate the data. The score was often validated appropriately, but the 3-character name was often checked for whether the length was 3 or less. This meant a 2-character name would be accepted by the algorithm, which did not meet the criteria. When validating the input data, some candidates validated both together. For example, if the name was invalid then both were output as invalid and required re-entry. Although this would meet the criteria for this question, it is important that candidates validate data independently and handle the invalid data suitably as well to make sure the program does not continue to run with invalid data.

(ii) This question required candidates to solve the problem of adding a new high score to the table. Candidates tackled this in a variety of suitable ways including: appending the data and then sorting the new list of eleven items, for example using a bubble sort; using an insertion sort method and moving each item from the end until the position was found and the new data inserted, creating a new list with the data up to the insertion point, inserting the new data and then the remainder. Some candidates did not move all data, especially if they were using two separate arrays, and only one was sorted, missing the name or the score.

(iii) Many candidates were able to call the sorting procedure appropriately, but fewer output the results before and after, often only showing the output after; on some occasions this could have been from candidates not providing all of their code to show where it was called, for example showing the output high score call and missing the previous one. It is important that candidates provide all the code for their solution to each question part. This can include code for other parts as it is preferable for the candidates to include all relevant code than to select small parts that might exclude code required for completeness.

(iv) Many candidates entered the correct data, but fewer produced an output with the data in the correct position. Some candidates did not use the given player name and score; when inputs are provided they need to be used in the output because these are designed to produce a specific output that will demonstrate their algorithms are functional.

(f) Many candidates were able to accurately open the file to write the data, and wrote the data to the file. Some candidates attempted to write eleven values instead of the 10 required for the top ten.

Question 2

This question required candidates to use object-oriented programming to solve the given problem.

(a) Many candidates were able to declare the given class, but fewer used the correct constructor, instead declaring a constructor method and not using their language's constructor method. Candidates were often successful in declaring the attributes as private and with appropriate data types. When using Python, candidates often used appropriate comments to indicate this in their algorithms. When an attribute is assigned a value in the constructor, this should not be included as a parameter. Some candidates passed a new health value as a parameter, which did not meet the requirement as this would allow the value to be different for each object.

(b) Many candidates were able to declare the get method suitably. Some candidates attempted to pass a parameter to the get method and then use this in the method, or they output instead of returning. When using Python, get methods require self to be passed as a parameter.

(c) This question was often answered well, with many candidates adding the parameter value to the attribute. Some candidates replaced the health value with the parameter instead of adding it to the current value. This is a method and therefore in Python self is required to be passed as a parameter. Candidates need to make sure they are meeting the requirements in the question. For example, in this question the value is added and there is no output or return value.

(d) This question was also a method for the class. Many candidates were able to compare the value and return the appropriate value. Some candidates passed a value to the function and then used this value to compare to 0, returning the result of the comparison.

(e) Many candidates were able to enter the defence item and colour appropriately. Fewer were able to instantiate an object of the class using these values, with some candidates hard-coding values into the constructor call. Some candidates successfully declared an instance of the object but did not store this object in a variable.

(f) There was a mix of responses to this question, with some candidates accurately manipulating the balloon object appropriately. Some candidates called methods without reference to the object, instead calling subroutines and passing a balloon object as a parameter. Some candidates accessed the values directly instead of using the methods which were required due to the private attributes. Some candidates did not pass and then return a balloon object. In some answers the parameter was named the same as the class, for example, balloon was passed as an object when balloon was the class identifier, causing a naming violation.

(g) (i) Many candidates were able to call their Defend function, but fewer passed the balloon object as a parameter. Fewer also stored the returned balloon object.

(ii) There were a mix of outputs for this question. Some candidates did not include all of the required outputs, for example missing the output of the defence item, and some candidates did not include the inputs in their evidence to show that the test was using the correct inputs.

Question 3

This question required candidates to create and manipulate a queue data structure using a 1D array.

(a) Candidates were often able to declare the variables appropriately, but some did not initialise them appropriately. The array could allow 10 string items, but some candidates did not include a suitable declaration to show that there were 10 elements in the array. When a list is being used in place of an array it needs to be initialised, or indicated, with the appropriate number of values as per the question requirements.

(b) This question included a pseudocode algorithm for candidates to convert into their chosen language and complete the missing statements. The algorithm included parameters passed by reference. Candidates were required to consider how their chosen language deals with by reference and write the algorithm appropriately. Some solutions were returning multiple values to overwrite those they were called from, while some solutions used global variables. Where global variables are used instead, candidates need to make this clear. Some candidates included the code for these to show where they had used them and some included comments to indicate that they had used global variables. Many candidates copied the code as in the algorithm and did not handle the passing parameters by reference, hence the changes made within the function will not be retained when it returns.

A common error in this algorithm was to store the data in the HeadPointer, or in the NumberOfItems value and not the TailPointer which pointed to the position at the end of the queue.

(c) Candidates found this question challenging. Many candidates followed the provided algorithm for enqueue and accessed the data in the tail pointer, checking this value and changing it, instead of the head pointer. When writing a function that returns a value candidates must make sure that this is the final statement in that part of the algorithm. Some candidates manipulated the head pointer and number of items after a value had been returned.

(d) (i) This question required the main program to read in eleven values from the user. Many candidates completed this, but fewer used each value in the function Enqueue(). Some candidates only input 10 values and did not include the 10^{th}.

(ii) Many candidates produced an output, but few had the correct results. A common error was missing the first dequeue of A, and where candidates had accessed the tail pointer instead of the head pointer K was often output instead. The candidates were required to show all of the inputs to show that they were testing it correctly, some candidates only showed a small number of the inputs (for example J and K) and then the output.