COMPUTER SCIENCE

Paper 9618/11
Theory Fundamentals

Key messages

There is now an increased requirement for application of knowledge, rather than just recall. Candidates should be aware that when a context is set in the stem of a question, that context holds good for the whole question and the subsequent parts of the question are then inter-related and follow on. For example, **Question 1** was all about images. Many answers to **part 1(b)** and **part 1(d)** did not take this into consideration.

Questions that required calculations or the understanding of code were answered well by most candidates. Candidates found questions that began with the command words 'describe' or 'explain' more challenging. If a question asks for a description answers need to include more detail, a single, brief statement is not enough for full credit.

There is a tendency to repeat information given in the question. For example, in **Question 4(d)**.

This is a subject that requires precision and exactness in answers. For example, data dependency and program-data dependency are two different things. In **Question 7(a)** many candidates wrote that data dependency was a drawback of a file-based approach to storing data when they meant program-data dependency.

Candidates should avoid using brand names for software.

General comments

Candidates must read the question carefully. If the question asks for a description of **one** benefit, marks will only be awarded for the first answer.

Some candidates did not follow the instructions in the question where a tick box was provided, and in questions where lines needed to be drawn between boxes. They put the wrong number of ticks or lines.

The binary arithmetic and low-level language questions were answered very well. Candidates found the questions about embedded systems and networks more challenging.

Comments on specific questions

Question 1

(a) (i) The stem of **Question 1(a)** indicates that the question is about a bitmapped image, therefore, the descriptions of the terms in this part question should refer to a bitmapped image. Simply stating that a pixel is a picture element is not enough at this level of study. There needs to be a description of what is meant by a picture element. While the inclusion of examples of the contents of a file header is commendable, candidates must be aware that when asked for a description a list is not enough and some further explanation is needed which should refer to the context given, in this case, a bitmap.

(ii) There were many good answers to this part question. The method for calculating the file size of an image was usually correct. Some candidates multiplied the number of pixels by 8 to calculate the number of bits required to store the image, but then did not divide by 8 to convert the bits to bytes.

(b) Most candidates correctly identified a suitable method of lossless compression. The most popular choice was Run-Length Encoding (RLE). Descriptions of RLE did not always make it clear that the identical pixels were adjacent to each other. The stem of the question stated that the **image** was compressed. A common incorrect response was to describe the compression of text by looking for repeated characters rather than the compression of the image by considering repeated pixels.

(c) (i) There were many correct answers for this question. Some candidates did not read the question carefully enough and gave the answer in binary instead of denary.

(ii) There were many correct answers to this calculation. Some candidates did not read the question carefully enough and did not follow the instruction to perform the addition in binary. Candidates should be aware that when asked to show their working, a brief explanation of each step of their calculation is required and any carry bits should be clearly visible.

(iii) The binary subtraction was more challenging. The question stated that the denary value 10 was to be subtracted from the hexadecimal value 23, that is $23_{16} – 10_{10}$. A common error was to perform the subtraction the wrong way round and take the two's complement of hexadecimal 23 instead of the two's complement of denary 10.

(d) Some candidates found describing copyright challenging. There was confusion between what was meant by copyright and what was involved with the breaking of copyright, with incorrect answers such as 'copyright means using something without permission'. Many answers referred to software rather than the image.

Question 2

(a) Many candidates correctly matched the disk formatter and back-up software to the appropriate descriptions. Some candidates did not read the descriptions carefully enough and matched the disk repair software to the first box in the list which described scanning **software**, rather than matching it to the fifth box which described scanning a disk.

(b) Most candidates were able to correctly identify two or three operating system management tasks. Giving four different tasks proved to be more challenging. A common incorrect response was to repeat one of the tasks already listed such as hardware management as well as device management.

Question 3

(a) There were some excellent completely correct answers to this part question. Some candidates need to be aware that if they expand the acronyms the expansions must be correct.

(b) Many candidates were able to correctly trace this code. A very common incorrect response was the inclusion of quotation marks around the output.

(c) (i) Almost all candidates were able to give the correct contents of the Accumulator.

(ii) Many candidates found this part question challenging and answers often repeated the exact wording given in the table of opcodes, whereas the question asked for the equivalent mathematical operation.

Question 4

(a) The most popular correct response to this question was the lack of a central controlling device and that all devices were of equal status. Many candidates found it more challenging to then give a second different point. There was considerable confusion between a peer-to-peer network and a mesh topology with a general misconception that a peer-to-peer network could not operate over a wireless connection.

(b) Many candidates realised that weaker security could be a drawback of this type of network, but a response that states just, 'weak security' is not enough for a description. There needs to be a reason why the security is weak. Some of the drawbacks described could apply to any network, not specifically to a peer-to-peer network.

(c) (i) There were some excellent answers to this question. Some candidates need to improve their understanding of the role of a router.

(ii) There were some very good answers to this question, with many candidates fully justifying their choice. Some candidates need to understand that imprecise statements such as, 'it is more convenient' are not enough. Answers should include why it is more convenient.

(d) Almost all candidates were able to correctly identify that both the internet and the World Wide Web were being used. Many candidates found it challenging to justify the use of both. Some candidates should understand that paraphrasing the question is not enough. A common statement was 'because her email account is accessed through a website', which simply repeats the question.

Question 5

(a) This question required the application of knowledge and many candidates found it challenging both to describe what is meant by an embedded system and then to relate it to a washing machine. Many candidates need to improve their understanding of the application of embedded systems.

(b) This question also required application of knowledge to the washing machine. Responses were frequently generic with no application to the given context.

(c) There were some excellent answers to this question which discussed feedback and changes to the environment. Some of the explanations included only information that had already been given in the question, a common statement was, 'because it turns off the cooling if the temperature is too low'.

Question 6

(a) The majority of candidates correctly identified this as a range check.

(b) There were many correct answers to this question. A common incorrect response was an explanation of the code rather than the naming of a validation check.

(c) Many candidates found it challenging to identity this validation check. A common incorrect answer was a type check.

Question 7

(a) There were some excellent answers to this question. Some candidates need to improve their understanding of what is meant by program-data dependence.

(b) (i) Many candidates were able to correctly identify a one-to-one relationship and a many-to-many relationship. The one-to-many relationship was more challenging. A common incorrect one-to-many-relationship was customer to product.

(ii) Almost all candidates correctly identified that the relationship was many-to-many.

(iii) Many candidates gave a correct DDL statement. Some candidates need to improve their understanding of the difference between a `CREATE TABLE` command and a `CREATE DATABASE` command.

(c) There were many correct answers to this question. A common incorrect answer was to include both field name and attribute.

Question 8

Candidates were told to tick **one** box in each **row**. Some responses included more than one tick in each row, especially for the last statement.

COMPUTER SCIENCE

Paper 9618/12
Theory Fundamentals

Key messages

There is now an increased requirement for application of knowledge, rather than just recall. Candidates should be aware that when a context is set in the stem of a question, that context applies to the whole question and the subsequent parts of the question are then inter-related and follow on. For example, in **Question 1**, the `DepositPaid` field had been given the data type `BOOLEAN` in **part 1(c)(i)** yet many candidates treated it as a text data type in their `WHERE` clause in **part 1(c)(ii)**.

Questions that required calculations or the understanding of code were answered well by most candidates. Candidates found questions that began with the command words 'describe' or 'explain' more challenging. If a question asks for a description answers need to include more detail, a single, brief statement is not enough for full credit.

At this level, this subject requires precision and exactness in answers. For example, in **Question 5(c)** vague statements that apply equally to either local storage or cloud storage will not gain credit.

Many candidates would benefit from greater familiarity with writing SQL statements involving more than one table.

Candidates should avoid using brand names for software.

General comments

Candidates must read the question carefully. If the question asks for a description of **one** benefit, marks will only be awarded for the first answer.

For questions where a tick box was provided, and questions where lines needed to be drawn between boxes, some candidates did not follow the instructions in the question and put the wrong number of ticks or lines.

The logic and low-level language questions were answered very well. Candidates found the questions about the components of a computer and Integrated Development Environments (IDEs) more challenging.

Comments on specific questions

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

Question 1

(a) Many of the candidates were able to give correct examples for each term taken from the database given in the question. Defining the terms proved to be more challenging. There was considerable confusion between the use of 'table' and 'database', with many responses using them inter-changeably. Some candidates need to improve their understanding of multi-table databases. A common incorrect definition of a foreign key did not mention the linking of the tables. An entity and a table are two different things. Many candidates described and gave an example of a table rather than an entity.

(b) This question also required application of knowledge to the given database. Many candidates correctly identified that the database was in 3NF, though often the justification included generic statements about the requirements of 3NF without any application to the given tables.

(c) (i) There were many correct answers to this question. Many candidates correctly completed the DDL statement.

(ii) There were some excellent DML scripts written for this question. Some candidates need to improve their understanding of using more than one table in an SQL query. A common error was the inclusion of quotation marks around the `No` in the `WHERE` clause when the `DepositPaid` field had been set as Boolean in the DDL script of the previous part question.

Question 2

(a) There were some very good answers to this question. Some candidates need to understand that they need to provide two distinct points for two marks. Some answers about Aisha acting ethically towards her colleagues were the same point written in two different ways.

(b) This question required the application of knowledge and there were some interesting and imaginative answers. Some candidates need to improve their understanding of the application of Artificial Intelligence to playing a board game. Many responses included generic statements about machine learning that were not applied to the context given.

(c) Most candidates correctly identified the licence type that each statement described.

Question 3

(a) Most candidates drew a correct logic circuit. There were some very neat solutions that made use of a single three-input AND gate. Some candidates need to improve their understanding of the difference between the symbols for an XOR gate and a NOR gate.

(b) Most candidates correctly completed the truth table.

Question 4

(a) Many candidates were able to correctly trace the program code. Some candidates need to improve their understanding of the difference between the `CMP <address>` instruction given in this table and the `CMP #n` instruction.

(b) (i) Almost all candidates correctly converted the unsigned binary value to denary.

(ii) There were many correct answers. Some candidates need to improve their understanding of the difference between the AND and OR logic operations.

(iii) This question was more challenging. A common incorrect response was `AND #6`.

Question 5

(a) There were some excellent completely correct answers to this part question. Some candidates need to be aware that if they expand the acronyms, the expansions must be correct.

(b) Many candidates found this question challenging and need to improve their understanding of the effect of changing the number of cores or the clock speed. Vague statements such as, ‘more cores means that the computer can process instructions faster’ are not enough. A better answer is, ‘more cores means that the computer can process more instructions simultaneously’.

(c) (i) There were some very good answers to this question. Many candidates understood the benefits of storing data using cloud computing.

(ii) Identifying drawbacks of using cloud computing was more challenging. Many of the drawbacks given applied equally to other methods of storing data and were not specific to cloud computing.

Some responses such as, 'it can only be accessed with a network connection' are not precise enough. It needs to be clear that it is a connection to the Internet that is required.

(d) There were many completely correct answers to this question.

Question 6

(a) There were a few completely correct answers to this question. A common incorrect answer for the number of characters in the extended ASCII character set was 255, the highest denary value, rather than 256.

(b) Many candidates found this question challenging. The two main points that were required were uniqueness and ordering. For example, statements such as, 'each character has a code' are not precise enough. It needs to be stated that the code for each character is unique.

(c) (i) There were many correct answers to this question. Some candidates gave the answer in binary when the question asked for hexadecimal.

(ii) There were many correct answers to this question. Some candidates gave the answer in binary when the question asked for denary.

Question 7

(a) This was an example of a situation where candidates needed to read the question carefully. The question asked **how** a program library could be used. Many responses listed the benefits of program libraries without describing how the library would be used.

(b) (i) This part question also required the application of knowledge rather than straightforward recall. Many candidates found it challenging to describe the ways that a compiler and interpreter can be used in program development and instead described the functions of the compiler and/or interpreter.

(ii) There were a few excellent answers to this question. Some candidates need to improve their understanding of the different types of tools provided by a typical IDE. Popular incorrect answers were context-sensitive prompts and pretty printing which are tools for coding rather than debugging.

Question 8

(a) There were many very good answers to this question.

(b) The majority of candidates were able to correctly identify one other software measure for restricting access to the computer.

(c) The majority of candidates were able to correctly identify two threats posed to data by networks and the Internet.

COMPUTER SCIENCE

Paper 9618/13
Theory Fundamentals

Key messages

There is now an increased requirement for application of knowledge, rather than just recall. Candidates should be aware that when a context is set in the stem of a question, that context holds good for the whole question and the subsequent parts of the question are then inter-related and follow on. For example, **Question 1** was all about images. Many answers to **part 1(b)** and **part 1(d)** did not take this into consideration.

Questions that required calculations or the understanding of code were answered well by most candidates. Candidates found questions that began with the command words 'describe' or 'explain' more challenging. If a question asks for a description answers need to include more detail, a single, brief statement is not enough for full credit.

There is a tendency to repeat information given in the question. For example, in **Question 4(d)**.

This is a subject that requires precision and exactness in answers. For example, data dependency and program-data dependency are two different things. In **Question 7(a)** many candidates wrote that data dependency was a drawback of a file-based approach to storing data when they meant program-data dependency.

Candidates should avoid using brand names for software.

General comments

Candidates must read the question carefully. If the question asks for a description of **one** benefit, marks will only be awarded for the first answer.

Some candidates did not follow the instructions in the question where a tick box was provided, and in questions where lines needed to be drawn between boxes. They put the wrong number of ticks or lines.

The binary arithmetic and low-level language questions were answered very well. Candidates found the questions about embedded systems and networks more challenging.

Comments on specific questions

Question 1

(a) (i) The stem of **Question 1(a)** indicates that the question is about a bitmapped image, therefore, the descriptions of the terms in this part question should refer to a bitmapped image. Simply stating that a pixel is a picture element is not enough at this level of study. There needs to be a description of what is meant by a picture element. While the inclusion of examples of the contents of a file header is commendable, candidates must be aware that when asked for a description a list is not enough and some further explanation is needed which should refer to the context given, in this case, a bitmap.

(ii) There were many good answers to this part question. The method for calculating the file size of an image was usually correct. Some candidates multiplied the number of pixels by 8 to calculate the number of bits required to store the image, but then did not divide by 8 to convert the bits to bytes.

(b) Most candidates correctly identified a suitable method of lossless compression. The most popular choice was Run-Length Encoding (RLE). Descriptions of RLE did not always make it clear that the identical pixels were adjacent to each other. The stem of the question stated that the **image** was compressed. A common incorrect response was to describe the compression of text by looking for repeated characters rather than the compression of the image by considering repeated pixels.

(c) (i) There were many correct answers for this question. Some candidates did not read the question carefully enough and gave the answer in binary instead of denary.

(ii) There were many correct answers to this calculation. Some candidates did not read the question carefully enough and did not follow the instruction to perform the addition in binary. Candidates should be aware that when asked to show their working, a brief explanation of each step of their calculation is required and any carry bits should be clearly visible.

(iii) The binary subtraction was more challenging. The question stated that the denary value 10 was to be subtracted from the hexadecimal value 23, that is $23_{16} – 10_{10}$. A common error was to perform the subtraction the wrong way round and take the two's complement of hexadecimal 23 instead of the two's complement of denary 10.

(d) Some candidates found describing copyright challenging. There was confusion between what was meant by copyright and what was involved with the breaking of copyright, with incorrect answers such as 'copyright means using something without permission'. Many answers referred to software rather than the image.

Question 2

(a) Many candidates correctly matched the disk formatter and back-up software to the appropriate descriptions. Some candidates did not read the descriptions carefully enough and matched the disk repair software to the first box in the list which described scanning **software**, rather than matching it to the fifth box which described scanning a disk.

(b) Most candidates were able to correctly identify two or three operating system management tasks. Giving four different tasks proved to be more challenging. A common incorrect response was to repeat one of the tasks already listed such as hardware management as well as device management.

Question 3

(a) There were some excellent completely correct answers to this part question. Some candidates need to be aware that if they expand the acronyms the expansions must be correct.

(b) Many candidates were able to correctly trace this code. A very common incorrect response was the inclusion of quotation marks around the output.

(c) (i) Almost all candidates were able to give the correct contents of the Accumulator.

(ii) Many candidates found this part question challenging and answers often repeated the exact wording given in the table of opcodes, whereas the question asked for the equivalent mathematical operation.

Question 4

(a) The most popular correct response to this question was the lack of a central controlling device and that all devices were of equal status. Many candidates found it more challenging to then give a second different point. There was considerable confusion between a peer-to-peer network and a mesh topology with a general misconception that a peer-to-peer network could not operate over a wireless connection.

(b) Many candidates realised that weaker security could be a drawback of this type of network, but a response that states just, 'weak security' is not enough for a description. There needs to be a reason why the security is weak. Some of the drawbacks described could apply to any network, not specifically to a peer-to-peer network.

(c) (i) There were some excellent answers to this question. Some candidates need to improve their understanding of the role of a router.

(ii) There were some very good answers to this question, with many candidates fully justifying their choice. Some candidates need to understand that imprecise statements such as, 'it is more convenient' are not enough. Answers should include why it is more convenient.

(d) Almost all candidates were able to correctly identify that both the internet and the World Wide Web were being used. Many candidates found it challenging to justify the use of both. Some candidates should understand that paraphrasing the question is not enough. A common statement was 'because her email account is accessed through a website', which simply repeats the question.

Question 5

(a) This question required the application of knowledge and many candidates found it challenging both to describe what is meant by an embedded system and then to relate it to a washing machine. Many candidates need to improve their understanding of the application of embedded systems.

(b) This question also required application of knowledge to the washing machine. Responses were frequently generic with no application to the given context.

(c) There were some excellent answers to this question which discussed feedback and changes to the environment. Some of the explanations included only information that had already been given in the question, a common statement was, 'because it turns off the cooling if the temperature is too low'.

Question 6

(a) The majority of candidates correctly identified this as a range check.

(b) There were many correct answers to this question. A common incorrect response was an explanation of the code rather than the naming of a validation check.

(c) Many candidates found it challenging to identity this validation check. A common incorrect answer was a type check.

Question 7

(a) There were some excellent answers to this question. Some candidates need to improve their understanding of what is meant by program-data dependence.

(b) (i) Many candidates were able to correctly identify a one-to-one relationship and a many-to-many relationship. The one-to-many relationship was more challenging. A common incorrect one-to-many-relationship was customer to product.

(ii) Almost all candidates correctly identified that the relationship was many-to-many.

(iii) Many candidates gave a correct DDL statement. Some candidates need to improve their understanding of the difference between a `CREATE TABLE` command and a `CREATE DATABASE` command.

(c) There were many correct answers to this question. A common incorrect answer was to include both field name and attribute.

Question 8

Candidates were told to tick **one** box in each **row**. Some responses included more than one tick in each row, especially for the last statement.

COMPUTER SCIENCE

Paper 9618/21
Problem-Solving and Programming

Key messages

This paper involves the application of practical skills. These skills involve analysing and understanding the requirement as well as designing and presenting a solution. Requirements are often presented using a scenario description. 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. 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 should also be advised that they should not answer a question by simply repeating phrases from the question.

It is important that candidates are familiar with the fundamental programming concepts. Lack of understanding is often illustrated by the confusion between a literal value and an identifier, or the misuse of `OUTPUT` in place of `RETURN`. Many candidates appear not to be able to use parameters; often replacing parameters to a procedure or function with a series of prompts and inputs within the body of the subroutine.

The concept of reserved keywords is not well understood. Please see the example under **Question 7(c)** in the next section.

The functions and operators that are available for use in pseudocode answers are described in the Insert which accompanies the paper. Candidates should be advised that they should not use language-specific functions or methods that do not appear in the insert.

Candidates need to read each question carefully before attempting to answer it. Questions may address topics in many ways, and it is often necessary to apply knowledge in a specific way if marks are to be gained. It should not be assumed that simply because a question contains some recognised terms that it is the same question that has appeared in past papers from the previous syllabus.

General comments

If answers are crossed out, the new answers must be written clearly so that the text may be read.

Many candidates make use of blank pages for rough work when preparing their final answer. In these cases, it is extremely important that this text is crossed out.

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

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 question part was well answered by most candidates.

Variable names such as 'flag' for row 2 and 'Date' for row three were too imprecise. 'Date' is also the name of a data type and therefore a keyword.

A very small number of candidates gave answers for the variable name column that were not names. On occasion these appeared to be alternative data types and sometimes the column was left blank.

(b) Some descriptions were not precise enough. For example, the answer '+ is used in calculations' for the last row does not adequately describe the error as it does not refer to the operands.

(c) A minority identified the use of a record and gave as a justification the fact that a record could hold different data types. The justification that the values could be stored under a single identifier was rarely seen.

Several answers suggested the use of two 1D arrays and so accessed these mark points.

While some programming languages support arrays containing multiple data types these are not supported by the pseudocode definitions that apply to this section of the syllabus.

Question 2

(a) This part question was generally answered well.

Common mistakes included the omission of the selection diamond (MP2) and occasional confusion over whether the circle on a parameter arrow should be shaded or unshaded.

A small number of answers suggested that the candidates concerned had not studied this topic.

(b) This part question was answered correctly by around half the candidates.

A common mistake was to present the subroutine as a function. Of those that correctly identified the subroutine as a procedure, most correctly identified the need for the `BYREF` keyword.

(c) A wide range of answers was seen. A small number of solutions gained full marks. Some answers suggested little familiarity with the topic.

The most common mistake was for the initialisation of the `Max` variable (MP1). Many candidates initialised this to zero or a large negative number, but the question gave no indication as to the range of values input and so the only correct solution was to take the first value input and to use this as the initial `Max` value. Solutions that did implement this either used a separate `INPUT` statement before the loop, or contained a condition within the loop to assign the first value input to `Max`.

MP2 was gained by most candidates, usually by assigning the value zero, and MP5 was commonly awarded.

Most solutions addressed MP3 although in some cases the number of iterations was not correct. Some candidates unsuccessfully attempted to implement the loop via a single flowchart symbol containing a description such as 'Repeat while Index < 51'.

MP6 was often missed, as solutions did not subtract the Max value and/or divided by 50 rather than 49.

Several solutions attempted to store the input values into an array. In many of these cases the attempt to find the largest value was incorrectly based on comparing two adjacent elements.

Question 3

(a) (i) This question presented a scenario and asked for an explanation of how files could be used to speed up the process. Many candidates made little or no reference to the scenario and simply described how arrays represented temporary storage while file storage was permanent.

Of those that did refer to the scenario, the most common mark given was for the third point – the avoidance of the need to re-enter data manually at the start of each day.

This was a question where candidates should ensure they give complete answers and relate them to the given scenario. In this case, for example, by stating that data could be 'read from the text file' and 'written back to the array' at the 'start of each day'.

(ii) Many candidates successfully answered this part question, either directly or via a follow-through from the previous question.

(iii) Many correct answers described the storing of items on a single line of the text file and the use of a special separator character. The method of storing one data item per line of the file was less common.

There were few references to the need to convert all data items to strings.

As for **Question 3 (a) (i)**, many candidates did not pay attention to the scenario and simply wrote 'the data is written to the file'

Many answers attempted to store the data as a record in the file, although the question specified a text file was to be used.

(b) Many solutions started with a correct procedure header, but in many cases, there was no procedure end (MP1). Some candidates attempted to pass a parameter with a name such as `ThisFile.txt`

Many solutions correctly opened the file in read mode but in many cases, the corresponding `CLOSEFILE` was missing (MP2). A common mistake was to treat the parameter name as a literal string by enclosing it in quotation marks when used in the `OPEN` and subsequent file operations.

A common mistake (for MP3) was to test the filename itself, rather than the file contents. Another mistake was to compare `EOF(filename)` with an empty string rather than a Boolean value. Many solutions read a line from the file and then checked whether the line was an empty string. Nothing in the question suggested that the file could *not* contain blank lines, so this approach to detect an empty file was not successful.

Many candidates recognised the need for a loop, but the fact that this would need to repeat for either five lines or until the end of the file was reached proved too complicated for many (MP4). Several unsuccessful attempts at nested loops were seen, such as a `FOR` loop for five iterations contained within a `WHILE` loop which tested for `EOF()`.

Many solutions gained MP5 for reading and then outputting a line within a loop. Some lost this mark as they chose to skip blank lines, which was not in the question.

Question 4

(a) A high percentage of correct answers was seen, with many gaining 4 or 5 marks.

Common errors were the absence of quotation marks around the data representing `NewName` or `ThisChar`, and the omission of the final value of 11 for identifier `Index`.

(b) Many varied loop structures were suggested. Where a count-controlled loop was identified, often the justification was rather convoluted and lacked the clarity of the answer 'because the number of iterations is known'.

(c) (i) The majority of candidates offered the correct modification and identified the line:

```
NewName ← NewName & LCASE(ThisChar)
```

A significant number made no attempt at this question.

(ii) A challenging question that was only answered correctly by a minority. A larger number correctly identified the line that should be moved but specified the wrong destination.

Question 5

A bubble sort with a slight variation in that the requirement was to perform a sort on a 2D array.

A small number of excellent solutions were seen which addressed all mark points including those concerned with the efficiency of the algorithm.

An inner and outer loop were present in most solutions (MP1 and MP2). The outer loop could be a straightforward count-controlled loops for MP1 but would need to be conditional loop for later mark points.

Often the range implemented by the inner loop meant that the final element comparison would have generated an out-of-bounds index value.

Most solutions correctly addressed the comparison statement for MP3. A very small number had reversed the comparison operator and so would have sorted the array into descending order.

MP4 was normally addressed successfully, but fewer solution identified the need to also swap the 'other' column value (MP5). Some solutions lost this mark by declaring the temporary swap variable as a string rather than an integer.

Many solutions addressed the 'efficiency' marks, often successfully. Common mistakes included:

- Setting a flag to TRUE when elements were swapped but also setting it to FALSE if they were not.
- Reducing the boundary counter inside the inner loop, rather than in the outer loop.
- Failing to rest the flag in the outer loop before the start of the inner loop.

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

Question 6

(a) Not well-answered.

Where marks were awarded, they were usually given for MP4 and MP5. These were very general points and could have been expected to be gained by someone without any specific knowledge of the ADT.

The operation of the free list was rarely mentioned (MP1, MP3 and MP6)

The explanation of how to find the correct insertion point (MP2), where given, was often limited to checking that the value was 'greater than the previous node', without taking account of the value in the next node in the list.

'Push' and 'pop' operations were referenced occasionally.

(b) Very poorly answered.

Many answers failed to refer to arrays and variables, even though this was asked for in the question.

A small number of solutions successfully referred to the use of an array to store a user-defined data type comprising a data element and a pointer.

The use of variables to store the start pointer and next free pointer was very rarely mentioned.

Question 7

(a) A wide range of marks was seen, with several candidates gaining full marks for very good solutions. A large number gained three or less marks.

Several candidates made little or no attempt at this question.

MP1 was achieved by many candidates. Some omitted the `ENDFUNCTION` statement and so did not gain this mark.

The need for iteration was identified by many, who gained MP2 for either a conditional or count-controlled loop.

Many candidates achieved MP3, for a familiar statement making use of the `MID()` function. The majority of these went on to compare the extracted character with a space character, and if equal to increment a count variable (MP4).

MP5 was more challenging for many candidates. Often the count variable had not been initialised, and the comparison with the parameter was often in error by one (For example, when searching for the second word it is only necessary to count one space)

MP6 was addressed usually by assigning the value of the current character position plus one (assuming the current character was a space character) to a variable, or by executing an immediate `RETURN` of the value. It was usually necessary for the variable to have been initialised to a rogue value such as –1 before the loop, so that the code after the loop could work correctly.

MP7 was often missed. In many cases a value was returned either if the word was found, or if it was not found, but not in both cases. Solutions that had implemented an immediate `RETURN` within the loop simply had to then return –1 after the loop to gain the mark. Solutions that have assigned a value to a variable under MP6 often tested this value before returning it, although an initial value of –1 would have simplified this part of the algorithm.

(b) Not well answered.

Few answers presented a coherent description of the process. Only a small number of candidates gave what could be considered as a 'detailed description of the algorithm' as asked for. To assist centres, the appendix at the end of this report illustrates the level of answer that was expected.

General descriptive phrases were common, such as 'Check if the array is full'. Often these phrases were copies of those given in the question.

Although absent from many answers, the reference to a loop was the most common mark awarded (MP2)

Answers that described testing individual array elements usually scored at least three marks.

(c) Not well answered. Several candidates made little or no attempt at this question.

A straightforward linear search, from a given start point until a given character was found or the end of the string was reached.

A small number of solutions contained conditional loops where the condition checked for a space character and for the end of `FNString`. These solutions often addressed MP1, MP2 and MP4.

Simpler count-controlled loops iterated from the parameter value to the length of `FNString` and contained code that attempted to build up the return string until a space character was encountered (MP1 and MP3).

Many conditional loops that simply checked for a space character would have generated an error when using the `MID()` function to extract a non-existent character from `FNString`.

It was common for no type of loop to be present.

Within the loop, the statement to build up the return string was attempted by many, often correctly (MP3). A common mistake was to omit the concatenation and simply assign each extracted character in turn to the return string. In some cases, the identifier name chosen for the return string was the same as the keyword:

```
DECLARE String : STRING
```

Most solutions addressed MP5, returning the created string after the loop. It was common for the string initialisation to have been omitted.

Appendix

For questions that require candidates to present a detailed description of an algorithm, it may be helpful for candidates to imagine they are describing the algorithm to someone who is just starting to learn about programming and so need a precise description of the steps required.

The following texts are based on actual answers to **Question 7 (b)**. Please note that these are NOT model answers but are intended to illustrate of the level of detail that should be given to achieve high marks.

The function AddWord loops through the 1D array IgnoreList using a count-controlled loop that loops 10 times. In the loop it checks each element of the array to see if it is unused or if it already contains the word.

If an unused element is found and the number is not already stored, then the word that was passed as a parameter is stored in the element.

Before the loop, a Boolean flag variable is initialised to FALSE. IF an unused element is found it is set to TRUE. Outside the loop, the Boolean flag variable is returned to the main program.

The count-controlled loop is carried out twice. In the first loop it checks whether or not the word is already stored in the array using a selection statement and sets a Boolean flag variable to FALSE.

The following example uses structured English:

1. Loop through each element of the array IgnoreList
2. Try to match the parameter word with the element of the array
3. If match is found, return FALSE
4. Loop through array again until empty element is found
5. Set element to value of word and return TRUE
6. If no empty element found then return FALSE

COMPUTER SCIENCE

Paper 9618/22
Problem-Solving and Programming

Key messages

This paper involves the application of practical skills. These skills involve analysing and understanding the requirement as well as designing and presenting a solution. Requirements are often presented using a scenario description. 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. 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 should also be advised that they should not answer a question by simply repeating phrases from the question.

It is important that candidates are familiar with the fundamental programming concepts. Lack of understanding is often illustrated by the confusion between a literal value and an identifier, or the misuse of `OUTPUT` in place of `RETURN`. Many candidates appear not to be able to use parameters; often replacing parameters to a procedure or function with a series of prompts and inputs within the body of the subroutine.

The concept of reserved keywords is not well understood. Please see the example under **Question 6** in the next section.

The functions and operators that are available for use in pseudocode answers are described in the Insert which accompanies the paper. Candidates should be advised that they should not use language-specific functions or methods that do not appear in the insert.

Candidates need to read each question carefully before attempting to answer it. Questions may address topics in many ways, and it is often necessary to apply knowledge in a specific way if marks are to be gained. It should not be assumed that simply because a question contains some recognised terms that it is the same question that has appeared in past papers from the previous syllabus.

General comments

If answers are crossed out, the new answers must be written clearly so that the text may be read.

Many candidates make use of blank pages for rough work when preparing their final answer. In these cases, it is extremely important that this text is crossed out.

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

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) (i) This question part was well answered by most candidates. Occasional errors included suggesting real for `Index` and string for `Holiday`.

(ii) This question part was well answered by most candidates. The most common error was the omission of quotation marks to indicate the string value `"Cat"`. A few candidates retained the decimal part of the number for row three and some suggested 'error' or zero for row 1.

(b) There were many correct answers to this part question. Some answers gave more than one tick for a row.

Question 2

(a) (i) Many candidates gained the mark for the last row of the table. Various answers were seen for the first three rows, with the first two rows appearing to provide the greatest challenge.

Common incorrect answers seen for row three were 'Input' or some scenario-specific expressions such as 'Water level'.

(ii) Many candidates gained both marks. Various incorrect answers for the number of outputs were seen. Candidates who gave an incorrect number of outputs often correctly identified the current state for the second mark point.

(b) (i) There were a wide range of responses to this part question. 'Date of loan' was the most common correct answer. Few answers included three correct items to gain both marks.

Many answers offered data items that were considered too ambiguous, such as 'Book name' or 'Borrower name' which could well be duplicated within the system. Where 'ID' was used as a suffix the mark was given.

(ii) Many correct answers were seen. Popular answers included 'Name of Author' or 'Title of Book'.

This question asked for 'an item of data' which would be required but many answered with values which could reasonably be expected to be calculated from other data such as 'Number of books on loan'

(iii) Many candidates took advantage of the example in the question and adapted this to provide a correct answer such as 'the **names** of all the people with overdue books'. Most answers focused on the production of a list of some sort, although a few directly addressed the loan and return process.

Some answers did not represent operations that would manipulate data. Others offered general computing terms, seen such as 'Procedure' and 'Breakpoint'.

Question 3

(a) Many identified the structure as a linked list. Other ADT names and general terms such as 'array' were frequently seen.

(b) Many candidates simply suggested 'Start' (as in a flowchart), or 'Start node'. 'Input' was commonly seen. A significant number did not attempt this question.

(c) Few candidates gave the correct term. Many just gave 'Pointer' as the answer.

Many answers to the 'meaning' were heading in the right direction but lacked the clarity that was required, such as in the phrase 'it points to nothing'.

References to other ADTs were common, for example suggesting that the null pointer pointed to 'the last thing in the queue'.

(d) Most candidates completed the diagram correctly.

Candidates need to ensure that the link arrow starts from the pointer section of the (previous) node. Others added what might have been array index values to the pointers of each mode but omitted the arrows altogether.

Question 4

Few answers presented a coherent narrative starting at the beginning of the process and progressing to the end. Only a small number of candidates gave what could be considered as a 'detailed description of the algorithm' as required. To assist centres, the appendix at the end of this report illustrates the level of answer that was expected.

Several answers took the form of numbered steps and often these gained high marks. More structured answers sometimes included fragments of pseudocode to illustrate the description.

Most did not mention the requirement of any sort of loop. Some responses provided detail about the thresholds and gave their own levels to compare marks, often with very long `IF` statements.

Many solutions included features that were not required, such as storing all names and test results in an array, converting each test mark into a percentage, or sorting the marks into ascending order.

A common error seen was referring to 'marks' as 'grades'. Statements such as 'input each grade' or 'add up all the grades' were frequently used.

Question 5

(a) (i) The array declaration (MP1) was successfully addressed by many candidates. Some of the responses used square brackets in the declaration but then ordinary brackets in the assignment statement.

Many candidates implemented a working `FOR` loop (MP2) and gained MP3 for correct array syntax.

Most attempts to generate a random number in the required range (MP4) were unsuccessful, despite a definition of the `RAND()` function being given in the insert. Some gave the `RAND()` parameter as 201 and many did not convert the generated value to an integer.

Some responses incorrectly used the > = comparison as a single symbol.

Most solutions ended with the output of a count variable, but in many cases, this had not been initialised before the loop.

(ii) Another question which required candidates to describe an algorithm, this time in terms of the changes that were required. Few candidates achieved more than one mark.

Many stated that another random value must be generated if not unique but did not go into detail as to how to find whether it was unique. They provided little more than the text from the question.

Those that attempted to describe the method of checking, often presented vague descriptions along the lines of 'compare with previous numbers'.

Many solutions did not mention the need for any sort of loop, and those that did often suggested a count-controlled loop rather than a conditional loop that would search the array until the value was found or the end of the array was reached.

(b) (i) A wide range of marks was seen, with many candidates gaining 3 or 4 marks. The first and third rows were most often correct; there did not appear to be any pattern to the incorrect answers.

(ii) Correctly answered by few candidates. Those that gained a mark using the `AND` operator often did not receive the second mark by not correctly changing the comparison statement, stating > 'a' rather than > = 'a'

Many candidates simply repeated the original line.

Question 6

Most solutions addressed MP1 successfully. The concept of reserved keywords is not well understood, with many procedure headings containing the sort of error illustrated:

```
PROCEDURE CountVowels(String : STRING)
```

Very few solutions initialised the global `CharCount` array (MP2). Many solutions introduced their own set of counter variables.

Many candidates identified that a loop for the length of the string was required (MP4) and many correctly extracted a single character within the loop (MP5).

A `CASE` statement was seen quite often but many did not assign the count to the array, or just kept an overall total of all the vowels. Some correct solutions used a variable to store the count of each vowel then assigned these totals to the array after the loop.

Most did not correctly calculate the total of the consonants (MP7). By just using `ELSE` or `OTHERWISE`, their solutions would select all other alphanumeric characters, rather than just the consonants. Some overcame this by checking that a character was in the rage 'a' to 'z' before addressing MP6 and MP7.

Most solutions contained an attempt to output the count values at the end. Solutions that used a comma-separated list were usually the most successful; those that attempted concatenation often used the incorrect operator ('`+`' in place of '`&`') and did not convert the integer count values into strings.

Question 7

(a) The full range of marks was seen, with many candidates gaining full marks. Some lost a mark for a single space between all words and/or the lack of an upper-case character.

Most candidates followed the directive to use the symbol to represent a space character.

Some answers given were complete lines of pseudocode which included the call to the procedure with the test string.

(b) Not well answered. Marks were usually gained for an answer that mapped to the first mark point. Other mark points that were seen less frequently included the addition of output statements and the use of breakpoints and single stepping. A small number of answers described the use of different test values in an attempt see which ones would fail.

Many general terms relating to testing were given.

Question 8

(a) Some recognised the need for a loop (MP1) but tried to use a `FOR` loop to `LENGTH(IgnoreList)`. As described in the Insert, function `LENGTH()` can only be used on strings.

Few solutions converted the strings to the same case (MP2).

Many solutions included a correct comparison (MP3), after which a flag would be set to `TRUE` (MP4). A common error was to include an `ELSE` statement that incorrectly set the flag to `FALSE` before continuing the loop.

(b) A small number of excellent solutions were seen that achieved full marks for this challenging question. Many good attempts were seen with some achieving close to full marks.

Although the module description points to a solution using the three user-defined functions `GetStart()`, `GetWord()` and `IgnoreWord()`, many candidates attempted to implement this module using only the functions given in the insert. This was generally not successful.

Some neat solutions were seen which nested the user-defined functions.

The use of the user-defined functions proved too challenging for many. They were unable to incorporate them into their own pseudocode.

Candidates who made a reasonable attempt at a solution usually achieved MP1.

A significant number of solutions did not include any loop (MP2). Of those that did, many chose to simply iterate through the length of `FNString` rather than to use the defined functions given in the question. Whether or not a simple count-controlled loop was successful was determined later in their solution.

Candidates who understood the purpose of the user-defined functions were able to incorporate them into their solution, giving straightforward flow from MP3 through to MP7.

Regardless of the use of the user-defined functions, many candidates addressed MP7 and finally MP9.

Appendix

For questions that require candidates to present a detailed description of an algorithm, it may be helpful for candidates to imagine they are describing the algorithm to someone who is just starting to learn about programming and so need a precise description of the steps required.

The following text is an amalgam of several actual answers to **Question 4**. Please note that this text is NOT a model answer but is an illustration of the level of detail that should be given to achieve high marks.

The algorithm will have to start by asking the teacher to input the marks using a post-condition loop. After each input, the teacher will be asked if she wants to continue, if she does the program exits the loop. In this loop each time a mark is input it will be added to a variable which after the loop ends will be divided by the number of inputs to get the average mark.

Each mark she inputs will be stored in an array and the number of inputs as well.

With the number of inputs, a FOR loop will be done in which each item in the array of marks will be compared each time around the loop with a series of IF statements the mark will be assigned a grade within the pre-defined ranges and stored in a new string for grades in the same index value as the mark.

At the end the program can either OUTPUT the array or output each grade one by one in a FOR loop. In addition, it will output the average mark with a suitable message.

COMPUTER SCIENCE

Paper 9618/23
Problem-Solving and Programming

Key messages

This paper involves the application of practical skills. These skills involve analysing and understanding the requirement as well as designing and presenting a solution. Requirements are often presented using a scenario description. 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. 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 should also be advised that they should not answer a question by simply repeating phrases from the question.

It is important that candidates are familiar with the fundamental programming concepts. Lack of understanding is often illustrated by the confusion between a literal value and an identifier, or the misuse of `OUTPUT` in place of `RETURN`. Many candidates appear not to be able to use parameters; often replacing parameters to a procedure or function with a series of prompts and inputs within the body of the subroutine.

The concept of reserved keywords is not well understood. Please see the example under **Question 7(c)** in the next section.

The functions and operators that are available for use in pseudocode answers are described in the Insert which accompanies the paper. Candidates should be advised that they should not use language-specific functions or methods that do not appear in the insert.

Candidates need to read each question carefully before attempting to answer it. Questions may address topics in many ways, and it is often necessary to apply knowledge in a specific way if marks are to be gained. It should not be assumed that simply because a question contains some recognised terms that it is the same question that has appeared in past papers from the previous syllabus.

General comments

If answers are crossed out, the new answers must be written clearly so that the text may be read.

Many candidates make use of blank pages for rough work when preparing their final answer. In these cases, it is extremely important that this text is crossed out.

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

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 question part was well answered by most candidates.

Variable names such as 'flag' for row 2 and 'Date' for row three were too imprecise. 'Date' is also the name of a data type and therefore a keyword.

A very small number of candidates gave answers for the variable name column that were not names. On occasion these appeared to be alternative data types and sometimes the column was left blank.

(b) Some descriptions were not precise enough. For example, the answer '+ is used in calculations' for the last row does not adequately describe the error as it does not refer to the operands.

(c) A minority identified the use of a record and gave as a justification the fact that a record could hold different data types. The justification that the values could be stored under a single identifier was rarely seen.

Several answers suggested the use of two 1D arrays and so accessed these mark points.

While some programming languages support arrays containing multiple data types these are not supported by the pseudocode definitions that apply to this section of the syllabus.

Question 2

(a) This part question was generally answered well.

Common mistakes included the omission of the selection diamond (MP2) and occasional confusion over whether the circle on a parameter arrow should be shaded or unshaded.

A small number of answers suggested that the candidates concerned had not studied this topic.

(b) This part question was answered correctly by around half the candidates.

A common mistake was to present the subroutine as a function. Of those that correctly identified the subroutine as a procedure, most correctly identified the need for the `BYREF` keyword.

(c) A wide range of answers was seen. A small number of solutions gained full marks. Some answers suggested little familiarity with the topic.

The most common mistake was for the initialisation of the `Max` variable (MP1). Many candidates initialised this to zero or a large negative number, but the question gave no indication as to the range of values input and so the only correct solution was to take the first value input and to use this as the initial `Max` value. Solutions that did implement this either used a separate `INPUT` statement before the loop, or contained a condition within the loop to assign the first value input to `Max`.

MP2 was gained by most candidates, usually by assigning the value zero, and MP5 was commonly awarded.

Most solutions addressed MP3 although in some cases the number of iterations was not correct. Some candidates unsuccessfully attempted to implement the loop via a single flowchart symbol containing a description such as 'Repeat while Index < 51'.

MP6 was often missed, as solutions did not subtract the Max value and/or divided by 50 rather than 49.

Several solutions attempted to store the input values into an array. In many of these cases the attempt to find the largest value was incorrectly based on comparing two adjacent elements.

Question 3

(a) (i) This question presented a scenario and asked for an explanation of how files could be used to speed up the process. Many candidates made little or no reference to the scenario and simply described how arrays represented temporary storage while file storage was permanent.

Of those that did refer to the scenario, the most common mark given was for the third point – the avoidance of the need to re-enter data manually at the start of each day.

This was a question where candidates should ensure they give complete answers and relate them to the given scenario. In this case, for example, by stating that data could be 'read from the text file' and 'written back to the array' at the 'start of each day'.

(ii) Many candidates successfully answered this part question, either directly or via a follow-through from the previous question.

(iii) Many correct answers described the storing of items on a single line of the text file and the use of a special separator character. The method of storing one data item per line of the file was less common.

There were few references to the need to convert all data items to strings.

As for **Question 3 (a) (i)**, many candidates did not pay attention to the scenario and simply wrote 'the data is written to the file'

Many answers attempted to store the data as a record in the file, although the question specified a text file was to be used.

(b) Many solutions started with a correct procedure header, but in many cases, there was no procedure end (MP1). Some candidates attempted to pass a parameter with a name such as `ThisFile.txt`

Many solutions correctly opened the file in read mode but in many cases, the corresponding `CLOSEFILE` was missing (MP2). A common mistake was to treat the parameter name as a literal string by enclosing it in quotation marks when used in the `OPEN` and subsequent file operations.

A common mistake (for MP3) was to test the filename itself, rather than the file contents. Another mistake was to compare `EOF(filename)` with an empty string rather than a Boolean value. Many solutions read a line from the file and then checked whether the line was an empty string. Nothing in the question suggested that the file could *not* contain blank lines, so this approach to detect an empty file was not successful.

Many candidates recognised the need for a loop, but the fact that this would need to repeat for either five lines or until the end of the file was reached proved too complicated for many (MP4). Several unsuccessful attempts at nested loops were seen, such as a `FOR` loop for five iterations contained within a `WHILE` loop which tested for `EOF()`.

Many solutions gained MP5 for reading and then outputting a line within a loop. Some lost this mark as they chose to skip blank lines, which was not in the question.

Question 4

(a) A high percentage of correct answers was seen, with many gaining 4 or 5 marks.

Common errors were the absence of quotation marks around the data representing `NewName` or `ThisChar`, and the omission of the final value of 11 for identifier `Index`.

(b) Many varied loop structures were suggested. Where a count-controlled loop was identified, often the justification was rather convoluted and lacked the clarity of the answer 'because the number of iterations is known'.

(c) (i) The majority of candidates offered the correct modification and identified the line:

```
NewName ← NewName & LCASE(ThisChar)
```

A significant number made no attempt at this question.

(ii) A challenging question that was only answered correctly by a minority. A larger number correctly identified the line that should be moved but specified the wrong destination.

Question 5

A bubble sort with a slight variation in that the requirement was to perform a sort on a 2D array.

A small number of excellent solutions were seen which addressed all mark points including those concerned with the efficiency of the algorithm.

An inner and outer loop were present in most solutions (MP1 and MP2). The outer loop could be a straightforward count-controlled loops for MP1 but would need to be conditional loop for later mark points.

Often the range implemented by the inner loop meant that the final element comparison would have generated an out-of-bounds index value.

Most solutions correctly addressed the comparison statement for MP3. A very small number had reversed the comparison operator and so would have sorted the array into descending order.

MP4 was normally addressed successfully, but fewer solution identified the need to also swap the 'other' column value (MP5). Some solutions lost this mark by declaring the temporary swap variable as a string rather than an integer.

Many solutions addressed the 'efficiency' marks, often successfully. Common mistakes included:

- Setting a flag to TRUE when elements were swapped but also setting it to FALSE if they were not.
- Reducing the boundary counter inside the inner loop, rather than in the outer loop.
- Failing to rest the flag in the outer loop before the start of the inner loop.

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

Question 6

(a) Not well-answered.

Where marks were awarded, they were usually given for MP4 and MP5. These were very general points and could have been expected to be gained by someone without any specific knowledge of the ADT.

The operation of the free list was rarely mentioned (MP1, MP3 and MP6)

The explanation of how to find the correct insertion point (MP2), where given, was often limited to checking that the value was 'greater than the previous node', without taking account of the value in the next node in the list.

'Push' and 'pop' operations were referenced occasionally.

(b) Very poorly answered.

Many answers failed to refer to arrays and variables, even though this was asked for in the question.

A small number of solutions successfully referred to the use of an array to store a user-defined data type comprising a data element and a pointer.

The use of variables to store the start pointer and next free pointer was very rarely mentioned.

Question 7

(a) A wide range of marks was seen, with several candidates gaining full marks for very good solutions. A large number gained three or less marks.

Several candidates made little or no attempt at this question.

MP1 was achieved by many candidates. Some omitted the `ENDFUNCTION` statement and so did not gain this mark.

The need for iteration was identified by many, who gained MP2 for either a conditional or count-controlled loop.

Many candidates achieved MP3, for a familiar statement making use of the `MID()` function. The majority of these went on to compare the extracted character with a space character, and if equal to increment a count variable (MP4).

MP5 was more challenging for many candidates. Often the count variable had not been initialised, and the comparison with the parameter was often in error by one (For example, when searching for the second word it is only necessary to count one space)

MP6 was addressed usually by assigning the value of the current character position plus one (assuming the current character was a space character) to a variable, or by executing an immediate `RETURN` of the value. It was usually necessary for the variable to have been initialised to a rogue value such as –1 before the loop, so that the code after the loop could work correctly.

MP7 was often missed. In many cases a value was returned either if the word was found, or if it was not found, but not in both cases. Solutions that had implemented an immediate `RETURN` within the loop simply had to then return –1 after the loop to gain the mark. Solutions that have assigned a value to a variable under MP6 often tested this value before returning it, although an initial value of –1 would have simplified this part of the algorithm.

(b) Not well answered.

Few answers presented a coherent description of the process. Only a small number of candidates gave what could be considered as a 'detailed description of the algorithm' as asked for. To assist centres, the appendix at the end of this report illustrates the level of answer that was expected.

General descriptive phrases were common, such as 'Check if the array is full'. Often these phrases were copies of those given in the question.

Although absent from many answers, the reference to a loop was the most common mark awarded (MP2)

Answers that described testing individual array elements usually scored at least three marks.

(c) Not well answered. Several candidates made little or no attempt at this question.

A straightforward linear search, from a given start point until a given character was found or the end of the string was reached.

A small number of solutions contained conditional loops where the condition checked for a space character and for the end of `FNString`. These solutions often addressed MP1, MP2 and MP4.

Simpler count-controlled loops iterated from the parameter value to the length of `FNString` and contained code that attempted to build up the return string until a space character was encountered (MP1 and MP3).

Many conditional loops that simply checked for a space character would have generated an error when using the `MID()` function to extract a non-existent character from `FNString`.

It was common for no type of loop to be present.

Within the loop, the statement to build up the return string was attempted by many, often correctly (MP3). A common mistake was to omit the concatenation and simply assign each extracted character in turn to the return string. In some cases, the identifier name chosen for the return string was the same as the keyword:

```
DECLARE String : STRING
```

Most solutions addressed MP5, returning the created string after the loop. It was common for the string initialisation to have been omitted.

Appendix

For questions that require candidates to present a detailed description of an algorithm, it may be helpful for candidates to imagine they are describing the algorithm to someone who is just starting to learn about programming and so need a precise description of the steps required.

The following texts are based on actual answers to **Question 7 (b)**. Please note that these are NOT model answers but are intended to illustrate of the level of detail that should be given to achieve high marks.

The function AddWord loops through the 1D array IgnoreList using a count-controlled loop that loops 10 times. In the loop it checks each element of the array to see if it is unused or if it already contains the word.

If an unused element is found and the number is not already stored, then the word that was passed as a parameter is stored in the element.

Before the loop, a Boolean flag variable is initialised to FALSE. IF an unused element is found it is set to TRUE. Outside the loop, the Boolean flag variable is returned to the main program.

The count-controlled loop is carried out twice. In the first loop it checks whether or not the word is already stored in the array using a selection statement and sets a Boolean flag variable to FALSE.

The following example uses structured English:

1. Loop through each element of the array IgnoreList
2. Try to match the parameter word with the element of the array
3. If match is found, return FALSE
4. Loop through array again until empty element is found
5. Set element to value of word and return TRUE
6. If no empty element found then return FALSE

COMPUTER SCIENCE

Paper 9618/31
Advanced Theory

There were too few candidates for a meaningful report to be produced.

COMPUTER SCIENCE

Paper 9618/32
Advanced Theory

Key messages

Candidates are required to demonstrate a detailed study of the topics covered by the syllabus, ensuring they make good use of appropriate technical terminology as appropriate for this advanced theory paper.

Candidates must demonstrate good examination technique and respond to each question after carefully reading it to make sure they understand what is required. Questions must answer the question in the appropriate manner determined by the command word of the question, for example, a question beginning with 'explain' requires more detail than a question beginning with 'identify'. If a question asks for working to be shown, candidates must ensure that they do this, to gain high marks.

Candidates are always advised to answer questions in the context of the question they are answering rather than in generic terms.

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

General comments

Many candidates demonstrated knowledge of techniques associated with data storage. Some candidates must ensure that they provide a detailed response.

Candidates generally performed well when answering the Artificial Intelligence (AI) questions, even though this is a topic that was not present in the legacy syllabus 9608.

Candidates demonstrated they were able to complete a truth table for a more complex logic circuit.

Comments on specific questions

Question 1

(a) Many candidates demonstrated their ability to represent a negative denary number in binary, and then as a floating-point number. High marks were achieved by most candidates. Some candidates did not show sufficient working or made errors in the final answer.

(b) A high proportion of candidates were able to demonstrate how to convert a binary floating-point number to its denary equivalent, with many candidates achieving high marks. Some candidates did give the final answer as a positive number instead of correctly identifying it as negative number.

(c) The majority of candidates were able to demonstrate the normalisation of a non-normalised floating-point binary number, along with the appropriate working. Some candidates made errors in their calculation of the exponent.

(d) (i) Most candidates recognised that the given number could not be stored accurately in the system described in the question as there were insufficient bits available in the mantissa, with some candidates going on to explain that an overflow error would occur. Alternatively, some candidates were able to demonstrate what was needed for a system to be able to store the given number accurately, however, very few candidates were able to state all three of these points.

(ii) The majority of candidates correctly stated that the number of bits in the mantissa of the given system would need to be increased to store the number accurately. A small number of candidates went further and suggested how the bits may be divided between the mantissa and exponent in a modified system.

Question 2

(a) A significant number of candidates were able to give a partial description of the purpose of a user-defined data type, with only a small number giving a fuller response leading to both marks. Many candidates stated a use of the data type, rather than its purpose.

(b) (i) Many candidates did not use the correct pseudocode technique to define an enumerated data type, which would begin with the keyword `TYPE`.

(ii) As with **part 2(b)(i)**, many candidates did not use the correct pseudocode technique to define an enumerated data type, which would begin with the keyword `TYPE`.

(c) Candidates demonstrated a better understanding of how to define a composite data type, with most candidates achieving some marks, and some candidates going on to achieve full marks.

Question 3

(a) The majority of candidates demonstrated good knowledge of the various terms associated with the operating system, with many high scoring responses.

(b) Candidates demonstrated some knowledge of the role of an interpreter and how it executes a program without producing a complete translated version of it. They usually achieved this by describing the role of the interpreter in working on each program statement, one at a time, and stopping the program if errors are found. Some candidates applied descriptions more appropriate to the role of a compiler to this question, and others wrote long answers that did not address the whole question.

Question 4

(a) (i) Some candidates were able to explain why Reverse Polish Notation (RPN) is used to carry out expressions, for example, in terms of its unambiguous representation, or that expressions are read from left to right. Other candidates incorrectly applied the question to the use of stacks and gave an answer more appropriate for **Question 4(a)(ii)**.

(ii) A large proportion of candidates identified that a stack is a suitable data structure to be used to evaluate an expression in RPN, with many of these candidates going on to show understanding of how a stack operates in relation to the RPN method. A binary tree was an alternative correct answer that was rarely seen.

(b) Most candidates successfully converted the infix expression to RPN.

(c) Most candidates successfully converted the RPN expression into an infix expression.

(d) A high proportion of candidates were able to evaluate a given RPN expression using a variety of techniques, including conversion to infix followed using algebra, or alternatively, using stack diagrams.

Question 5

(a) A high proportion of candidates were able to state the correct answers to this question, which was to calculate the shortest distance between the base and several other towns, using Dijkstra's algorithm. Most of these candidates also demonstrated their working, which showed how they arrived at the shortest distances, and in most cases, showed multiple routes between the base and the towns. A small number of candidates demonstrated more detailed use of Dijkstra's algorithm, by showing other evidence, such as initially setting the Base to 0, the other towns to ∞, or clearly

identifying 'visited nodes'. The full range of marks was seen, with most candidates achieving at least some of these marks.

(b) A wide range of marks was achieved for this question with most candidates gaining some of the marks. Many different answers were seen, for example, candidates recognised the use of graphs in Artificial Intelligence for Artificial Neural Networks, relationships, or connections between nodes, or in a range of methods such as back propagation of errors.

Question 6

Candidates who gave benefits of packet switching, such as 'missing packets can be easily detected and a resend request sent', and/or drawbacks, such as 'time delays to correct errors may occur', were rewarded by gaining marks. Others simply stated features of packet switching, or differences between packet switching and circuit switching. Some candidates also gave two versions of what was either the same benefit or the same drawback.

Question 7

(a) Most candidates were able to achieve at least some of the marks for completing the truth table for the given logic circuit, with many of these candidates achieving full marks.

(b) Most candidates who answered this question recognised that the diagram represented a full adder logic circuit.

(c) The majority of candidates achieved some of the marks for this question. For the most part, the candidates recognised the purpose of the two outputs in the logic circuit. Many candidates did not achieve the marks for the Boolean algebra representations of the outputs as sum-of-products. Many incorrect answers featured simplified Boolean algebra expressions, which the question did not ask for.

Question 8

(a) The majority of candidates recognised at least one factor that could affect the performance of a sorting algorithm, with many of these candidates achieving both marks.

(b) The full range of marks was seen for this question, which required the writing of an algorithm to perform an insertion sort based on the given scenario and example. Few candidates provided completely correct solutions. There were many partially correct solutions, mostly related to the temporary storage and insertion parts of the algorithm.

Question 9

(a) A large proportion of candidates were able to describe at least one aspect of an imperative (procedural) programming language, with some of these candidates able to expand their answers and achieve both marks. Some candidates gave answers related to declarative programming languages.

(b) A large proportion of candidates were able to describe, at least in part, what is meant by a declarative language, with some of these candidates able to expand their answers and achieve both marks. Some candidates gave answers related to imperative (procedural) programming languages.

(c) The full range of marks was seen for this question, with candidates recognising some, or all, of the examples of programming paradigms represented by the given samples of program code. A few candidates incorrectly described what the code was doing rather than stating the programming paradigm it represented.

COMPUTER SCIENCE

Paper 9618/33
Advanced Theory

Key messages

Candidates are required to demonstrate a detailed study of the topics covered by the syllabus, ensuring they make good use of appropriate technical terminology as appropriate for this advanced theory paper.

Candidates must demonstrate good examination technique and respond to each question after carefully reading it to make sure they understand what is required. Questions must answer the question in the appropriate manner determined by the command word of the question, for example, a question beginning with 'explain' requires more detail than a question beginning with 'identify'. If a question asks for working to be shown, candidates must ensure that they do this, to gain high marks.

Candidates are always advised to answer questions in the context of the question they are answering rather than in generic terms.

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

General comments

Many candidates demonstrated knowledge of techniques associated with data storage. Some candidates must ensure that they provide a detailed response.

Candidates generally performed well when answering the Artificial Intelligence (AI) questions, even though this is a topic that was not present in the legacy syllabus 9608.

Candidates demonstrated they were able to complete a truth table for a more complex logic circuit.

Comments on specific questions

Question 1

(a) Many candidates demonstrated their ability to represent a negative denary number in binary, and then as a floating-point number. High marks were achieved by most candidates. Some candidates did not show sufficient working or made errors in the final answer.

(b) A high proportion of candidates were able to demonstrate how to convert a binary floating-point number to its denary equivalent, with many candidates achieving high marks. Some candidates did give the final answer as a positive number instead of correctly identifying it as negative number.

(c) The majority of candidates were able to demonstrate the normalisation of a non-normalised floating-point binary number, along with the appropriate working. Some candidates made errors in their calculation of the exponent.

(d) (i) Most candidates recognised that the given number could not be stored accurately in the system described in the question as there were insufficient bits available in the mantissa, with some candidates going on to explain that an overflow error would occur. Alternatively, some candidates were able to demonstrate what was needed for a system to be able to store the given number accurately, however, very few candidates were able to state all three of these points.

(ii) The majority of candidates correctly stated that the number of bits in the mantissa of the given system would need to be increased to store the number accurately. A small number of candidates went further and suggested how the bits may be divided between the mantissa and exponent in a modified system.

Question 2

(a) A significant number of candidates were able to give a partial description of the purpose of a user-defined data type, with only a small number giving a fuller response leading to both marks. Many candidates stated a use of the data type, rather than its purpose.

(b) (i) Many candidates did not use the correct pseudocode technique to define an enumerated data type, which would begin with the keyword `TYPE`.

(ii) As with **part 2(b)(i)**, many candidates did not use the correct pseudocode technique to define an enumerated data type, which would begin with the keyword `TYPE`.

(c) Candidates demonstrated a better understanding of how to define a composite data type, with most candidates achieving some marks, and some candidates going on to achieve full marks.

Question 3

(a) The majority of candidates demonstrated good knowledge of the various terms associated with the operating system, with many high scoring responses.

(b) Candidates demonstrated some knowledge of the role of an interpreter and how it executes a program without producing a complete translated version of it. They usually achieved this by describing the role of the interpreter in working on each program statement, one at a time, and stopping the program if errors are found. Some candidates applied descriptions more appropriate to the role of a compiler to this question, and others wrote long answers that did not address the whole question.

Question 4

(a) (i) Some candidates were able to explain why Reverse Polish Notation (RPN) is used to carry out expressions, for example, in terms of its unambiguous representation, or that expressions are read from left to right. Other candidates incorrectly applied the question to the use of stacks and gave an answer more appropriate for **Question 4(a)(ii)**.

(ii) A large proportion of candidates identified that a stack is a suitable data structure to be used to evaluate an expression in RPN, with many of these candidates going on to show understanding of how a stack operates in relation to the RPN method. A binary tree was an alternative correct answer that was rarely seen.

(b) Most candidates successfully converted the infix expression to RPN.

(c) Most candidates successfully converted the RPN expression into an infix expression.

(d) A high proportion of candidates were able to evaluate a given RPN expression using a variety of techniques, including conversion to infix followed using algebra, or alternatively, using stack diagrams.

Question 5

(a) A high proportion of candidates were able to state the correct answers to this question, which was to calculate the shortest distance between the base and several other towns, using Dijkstra's algorithm. Most of these candidates also demonstrated their working, which showed how they arrived at the shortest distances, and in most cases, showed multiple routes between the base and the towns. A small number of candidates demonstrated more detailed use of Dijkstra's algorithm, by showing other evidence, such as initially setting the Base to 0, the other towns to ∞, or clearly

identifying 'visited nodes'. The full range of marks was seen, with most candidates achieving at least some of these marks.

(b) A wide range of marks was achieved for this question with most candidates gaining some of the marks. Many different answers were seen, for example, candidates recognised the use of graphs in Artificial Intelligence for Artificial Neural Networks, relationships, or connections between nodes, or in a range of methods such as back propagation of errors.

Question 6

Candidates who gave benefits of packet switching, such as 'missing packets can be easily detected and a resend request sent', and/or drawbacks, such as 'time delays to correct errors may occur', were rewarded by gaining marks. Others simply stated features of packet switching, or differences between packet switching and circuit switching. Some candidates also gave two versions of what was either the same benefit or the same drawback.

Question 7

(a) Most candidates were able to achieve at least some of the marks for completing the truth table for the given logic circuit, with many of these candidates achieving full marks.

(b) Most candidates who answered this question recognised that the diagram represented a full adder logic circuit.

(c) The majority of candidates achieved some of the marks for this question. For the most part, the candidates recognised the purpose of the two outputs in the logic circuit. Many candidates did not achieve the marks for the Boolean algebra representations of the outputs as sum-of-products. Many incorrect answers featured simplified Boolean algebra expressions, which the question did not ask for.

Question 8

(a) The majority of candidates recognised at least one factor that could affect the performance of a sorting algorithm, with many of these candidates achieving both marks.

(b) The full range of marks was seen for this question, which required the writing of an algorithm to perform an insertion sort based on the given scenario and example. Few candidates provided completely correct solutions. There were many partially correct solutions, mostly related to the temporary storage and insertion parts of the algorithm.

Question 9

(a) A large proportion of candidates were able to describe at least one aspect of an imperative (procedural) programming language, with some of these candidates able to expand their answers and achieve both marks. Some candidates gave answers related to declarative programming languages.

(b) A large proportion of candidates were able to describe, at least in part, what is meant by a declarative language, with some of these candidates able to expand their answers and achieve both marks. Some candidates gave answers related to imperative (procedural) programming languages.

(c) The full range of marks was seen for this question, with candidates recognising some, or all, of the examples of programming paradigms represented by the given samples of program code. A few candidates incorrectly described what the code was doing rather than stating the programming paradigm it represented.

COMPUTER SCIENCE

Paper 9618/41
Practical

There were too few candidates for a meaningful report to be produced.

COMPUTER SCIENCE

Paper 9618/42
Practical

Key messages

This was the first series for the new practical examination. There were many responses written in Python, some in VB.NET but very few used Java.

candidates need to make sure they copy the code into the correct space in the evidence document. Some candidates found it easier to copy all their code into each answer space which is acceptable provided the required code for that question is also included. When taking screenshots candidates need to make sure that these are cropped appropriately, to make sure that all inputs and outputs are visible, but also that the results are clearly legible.

When writing Python candidates need to make sure that declarations given is in appropriate format, such as a comment to identify the data type, or to identify that an attribute is declared when no value is assigned.

General comments

There were a range of approaches taken to many questions, showing a variety of programming skills and methods of solving the given problems. Where candidates were not able to complete one question, they often continued appropriately, gaining marks in later questions even though the solutions may not be fully functional.

Candidates were able to copy their code into the appropriate evidence document, although they do need to check that these are copied into the correct spaces. When candidates are given data in a question, they need to double check that they have entered the correct values into their program to avoid minor errors.

Candidates found the manipulation of classes challenging at times, for example the requirement to use get methods to access private attributes.

Comments on specific questions

Question 1

(a) Many candidates were able to answer this question well. They were able to declare a suitable structure for the record node and the two fields. When candidates are using Python and field declarations are required, they need to make sure they include these in an appropriate format, such as a comment to indicate the data type of each field.

Candidates took a range of approaches to this question. Some wrote structures whilst others wrote classes or dictionaries. Where a class is being used, candidates will need to declare more than the class outline to make sure it is functional, for example, by including the constructor with the required fields.

(b) Where candidates had declared a suitable structure for **part a**, they were often able to use this effectively to assign the data and nodes. Some candidates attempted to declare these in separate arrays, one for the data and one for the node, which did not meet the requirements of the question having a single array of type node.

When using Python and a data type is required, candidates need to make sure this is included in their answer, for example through a comment, or in some cases candidates assigned a series of elements of type node where they had declared a class which indicated the data type.

When candidates are provided with data, they need to make sure they are using the correct data. Some candidates made errors in the values assigned.

Some candidates did not declare the required pointers, or they did not use the data provided in the stem of the task to assign the pointers with the correct values.

(c) (i) Some candidates found this part challenging. They output the array values sequentially from the index 0 to index 9, instead of following the pointers to output the linked list in the correct order. Where candidates did access and use the pointers, they were often able to do this accurately, accessing the next node value using an appropriate method of direct access.

Some candidates when looping, attempted to check if the node data was empty as opposed to checking whether the next node value was empty.

(ii) Many candidates provided a clear screenshot of the results. Some candidates had cropped too much off the screenshot or had made it too small. Candidates need to make sure their screenshot includes all the output data, and that the values can be clearly viewed.

(d) (i) Many candidates made suitable attempts at this question, although their solution was not fully functional. Only a small number of candidates were able to write a fully functional solution.

Most candidates were able to write the function declaration. Some candidates did not follow the question instruction for the linked list and pointers as parameters, with many excluding the pointers.

Most candidates were able to take the data as input. Some candidates attempted to pass this value as a parameter instead of the function taking it as input.

Many candidates did not make use of the empty list pointer. This points to the first empty node, i.e., the position to place the next node. Instead, candidates checked each array element in turn to find one without any data, which would insert the data into an empty node, but this may not be the next free empty node to be used.

Fewer candidates followed the pointers to find the last element in the linked list and were not able to update the pointer of the last element to the new element.

Some candidates were able to update the empty list pointer appropriately, and many candidates returned true after successful addition.

(ii) This part was answered well by many candidates who were able to call the functions they had defined. Some candidates called `addNode()` without making use of the return result to output a message as required by the question.

(iii) Not all candidates provided a response to this question. Those that did were often given follow through marks due to repeating the errors they made previously. Many of those did have the required data inserted into the array, although these were not all in the appropriate location.

Question 2

(a) Most candidates were able to declare the array with the appropriate number of elements. Most candidates were also able to assign the appropriate values. Candidates need to make sure they copy values accurately. Some candidates did not copy all the values correctly.

(b) (i) This part was answered well, with many candidates were able to write a linear search to check through each array value from the first to the last. A few candidates output ‘True’ and ‘False’, or another message, instead of the returning the required values.

(ii) Many candidates were also able to answer this question well. Candidates were able to take a value as input. Some candidates did valid or cast the value as an integer before using it which was

appropriate. Some candidates called the required function but did not store or make use of the return value to output a suitable message.

(iii) Many candidates were able to provide suitable screenshots in response. The question required two screenshots, but some candidates only provided one, e.g., finding a value in the array, and not one that was not found.

(c) This part required candidates to interpret the pseudocode solution, identify the missing elements and then recreate the algorithm in their chosen language. When doing this, candidates need to make sure they are using the algorithm provided, and not producing a different solution that does not follow the given algorithm, for example, the use of a variable `temp` to store the data whilst it is being moved. A common error was candidates not identifying that the algorithm sorts into descending order, with many using the incorrect comparison operator.

Question 3

(a) Many candidates were able to declare the class identified. When declaring attributes, some candidates missed the requirement for a private attribute. In Python, this can be done by using an comment or using the standard self.__ identifier starter. Candidates writing in Python must make sure they include comments for the data types of the attributes as stated in the question.

(b) Candidates found elements of this question challenging. Candidates were required to access the data from the text file, and many candidates were able to do this accurately. A common error included not closing the file within the procedure. Some candidates made appropriate use of exception handling when accessing the data in the file.

Some candidates read in the data but were unable to create an instance of treasure chest with each value read in and add this to the array.

(c) (i) Candidates aften demonstrated a good understanding of get methods and were able to write an appropriate function to return the attributes.

(ii) This part was often answered well, with candidates writing an appropriate function and comparing the correct values. Some candidates did not make use of the attribute in the comparison, for example, comparing a parameter to a read in value.

(iii) Most candidate were able to write the outline for the selection statement, but many did not perform the correct calculation for integer division. Some candidates used standard division without then casting this as an integer for return or declaring the return value would be an integer. Some candidates attempted to use modular division.

(iv) Fewer candidates attempted this question. Those that did often called the correct procedure and input the required value. Fewer were able to use the appropriate methods to access the question and answer from the array, instead attempting to directly access the values. Some output the actual question(s) required for the test in **part (c)(v)**.

(v) This part was not attempted by many candidates. Those that did often had accurate results that showed the appropriate inputs and outputs. When showing these screenshots, candidates must make sure that all data is visible, which may need to be done in two screenshots if they will not clearly fit on the page.

Some candidates only provided one screenshot for the first test and omitted the second test.

COMPUTER SCIENCE

Paper 9618/43
Practical

Key messages

This was the first series for the new practical examination. There were many responses written in Python, some in VB.NET but very few used Java.

candidates need to make sure they copy the code into the correct space in the evidence document. Some candidates found it easier to copy all their code into each answer space which is acceptable provided the required code for that question is also included. When taking screenshots candidates need to make sure that these are cropped appropriately, to make sure that all inputs and outputs are visible, but also that the results are clearly legible.

When writing Python candidates need to make sure that declarations given is in appropriate format, such as a comment to identify the data type, or to identify that an attribute is declared when no value is assigned.

General comments

There were a range of approaches taken to many questions, showing a variety of programming skills and methods of solving the given problems. Where candidates were not able to complete one question, they often continued appropriately, gaining marks in later questions even though the solutions may not be fully functional.

Candidates were able to copy their code into the appropriate evidence document, although they do need to check that these are copied into the correct spaces. When candidates are given data in a question, they need to double check that they have entered the correct values into their program to avoid minor errors.

Candidates found the manipulation of classes challenging at times, for example the requirement to use get methods to access private attributes.

Comments on specific questions

Question 1

(a) Many candidates were able to answer this question well. They were able to declare a suitable structure for the record node and the two fields. When candidates are using Python and field declarations are required, they need to make sure they include these in an appropriate format, such as a comment to indicate the data type of each field.

Candidates took a range of approaches to this question. Some wrote structures whilst others wrote classes or dictionaries. Where a class is being used, candidates will need to declare more than the class outline to make sure it is functional, for example, by including the constructor with the required fields.

(b) Where candidates had declared a suitable structure for **part a**, they were often able to use this effectively to assign the data and nodes. Some candidates attempted to declare these in separate arrays, one for the data and one for the node, which did not meet the requirements of the question having a single array of type node.

When using Python and a data type is required, candidates need to make sure this is included in their answer, for example through a comment, or in some cases candidates assigned a series of elements of type node where they had declared a class which indicated the data type.

When candidates are provided with data, they need to make sure they are using the correct data. Some candidates made errors in the values assigned.

Some candidates did not declare the required pointers, or they did not use the data provided in the stem of the task to assign the pointers with the correct values.

(c) (i) Some candidates found this part challenging. They output the array values sequentially from the index 0 to index 9, instead of following the pointers to output the linked list in the correct order. Where candidates did access and use the pointers, they were often able to do this accurately, accessing the next node value using an appropriate method of direct access.

Some candidates when looping, attempted to check if the node data was empty as opposed to checking whether the next node value was empty.

(ii) Many candidates provided a clear screenshot of the results. Some candidates had cropped too much off the screenshot or had made it too small. Candidates need to make sure their screenshot includes all the output data, and that the values can be clearly viewed.

(d) (i) Many candidates made suitable attempts at this question, although their solution was not fully functional. Only a small number of candidates were able to write a fully functional solution.

Most candidates were able to write the function declaration. Some candidates did not follow the question instruction for the linked list and pointers as parameters, with many excluding the pointers.

Most candidates were able to take the data as input. Some candidates attempted to pass this value as a parameter instead of the function taking it as input.

Many candidates did not make use of the empty list pointer. This points to the first empty node, i.e., the position to place the next node. Instead, candidates checked each array element in turn to find one without any data, which would insert the data into an empty node, but this may not be the next free empty node to be used.

Fewer candidates followed the pointers to find the last element in the linked list and were not able to update the pointer of the last element to the new element.

Some candidates were able to update the empty list pointer appropriately, and many candidates returned true after successful addition.

(ii) This part was answered well by many candidates who were able to call the functions they had defined. Some candidates called `addNode()` without making use of the return result to output a message as required by the question.

(iii) Not all candidates provided a response to this question. Those that did were often given follow through marks due to repeating the errors they made previously. Many of those did have the required data inserted into the array, although these were not all in the appropriate location.

Question 2

(a) Most candidates were able to declare the array with the appropriate number of elements. Most candidates were also able to assign the appropriate values. Candidates need to make sure they copy values accurately. Some candidates did not copy all the values correctly.

(b) (i) This part was answered well, with many candidates were able to write a linear search to check through each array value from the first to the last. A few candidates output ‘True’ and ‘False’, or another message, instead of the returning the required values.

(ii) Many candidates were also able to answer this question well. Candidates were able to take a value as input. Some candidates did valid or cast the value as an integer before using it which was

appropriate. Some candidates called the required function but did not store or make use of the return value to output a suitable message.

(iii) Many candidates were able to provide suitable screenshots in response. The question required two screenshots, but some candidates only provided one, e.g., finding a value in the array, and not one that was not found.

(c) This part required candidates to interpret the pseudocode solution, identify the missing elements and then recreate the algorithm in their chosen language. When doing this, candidates need to make sure they are using the algorithm provided, and not producing a different solution that does not follow the given algorithm, for example, the use of a variable `temp` to store the data whilst it is being moved. A common error was candidates not identifying that the algorithm sorts into descending order, with many using the incorrect comparison operator.

Question 3

(a) Many candidates were able to declare the class identified. When declaring attributes, some candidates missed the requirement for a private attribute. In Python, this can be done by using an comment or using the standard self.__ identifier starter. Candidates writing in Python must make sure they include comments for the data types of the attributes as stated in the question.

(b) Candidates found elements of this question challenging. Candidates were required to access the data from the text file, and many candidates were able to do this accurately. A common error included not closing the file within the procedure. Some candidates made appropriate use of exception handling when accessing the data in the file.

Some candidates read in the data but were unable to create an instance of treasure chest with each value read in and add this to the array.

(c) (i) Candidates aften demonstrated a good understanding of get methods and were able to write an appropriate function to return the attributes.

(ii) This part was often answered well, with candidates writing an appropriate function and comparing the correct values. Some candidates did not make use of the attribute in the comparison, for example, comparing a parameter to a read in value.

(iii) Most candidate were able to write the outline for the selection statement, but many did not perform the correct calculation for integer division. Some candidates used standard division without then casting this as an integer for return or declaring the return value would be an integer. Some candidates attempted to use modular division.

(iv) Fewer candidates attempted this question. Those that did often called the correct procedure and input the required value. Fewer were able to use the appropriate methods to access the question and answer from the array, instead attempting to directly access the values. Some output the actual question(s) required for the test in **part (c)(v)**.

(v) This part was not attempted by many candidates. Those that did often had accurate results that showed the appropriate inputs and outputs. When showing these screenshots, candidates must make sure that all data is visible, which may need to be done in two screenshots if they will not clearly fit on the page.

Some candidates only provided one screenshot for the first test and omitted the second test.