2018 HSC Software Design and Development Marking Guidelines

Section I

Multiple-choice Answer Key

Question	Answer
1	D
2	C
3	B
4	B
5	B
6	C
7	A
8	B
9	D
10	D
11	B
12	A
13	C
14	C
15	D
16	C
17	B
18	A
19	A
20	C

Section II

Question 21 (a)

Criteria	Marks
• Outlines social and ethical issues with reference to this system	3
• Outlines one social ethical issue with reference to this system	2
• Identifies a social or ethical issue	1

Sample answer:

It is possible to store details of every trip made by each user. This could be used by those wanting to track the routes taken eg employers, stalkers, potential advertisers.

Those not wanting to provide bank account details or without a smartphone or connectivity, can't hire the cars.

Answers could include:

Bank account details need to be stored securely.

Question 21 (b)

Criteria	Marks
• Identifies an appropriate development approach and provides thorough justification	3
• Identifies an appropriate development approach with some justification	2
• Identifies a development approach	1

Sample answer:

The flexible agile approach is best suited as it allows for new versions to be delivered quickly, enabling the incorporation of new features in a short timeframe, based on user feedback. Because the car share application is relatively small and online, users will easily accept new features as they are identified and implemented. The users will appreciate the responsiveness of the app developers to their feedback.

Question 21 (c)

Criteria	Marks
• Identifies a suitable installation method and provides thorough justification	3
• Identifies a suitable installation method and provides some justification	2
• Identifies an installation method	1

Sample answer:

Pilot.
The developers could select a limited area eg a suburb, with representative car hirers to test the system before rolling it out over the whole country/city. Feedback from the users can be used to identify issues and allow appropriate modifications to be made. In this way the impact of any issues can be minimised and resolved before the larger rollout.

Question 22 (a)

Criteria	Marks
• Provides a substantially correct desk check that shows the infinite loop	3
• Provides a substantially correct desk check	2
• Shows some understanding of a desk check	1

Sample answer:

len	name	finish	result	start
3	dog	4	" "	2
			0	
				3
			og	4
				1
			ogD	2
			ogDo	3
			ogDog	4
			ogDogD	1
				2

The process never ends.

Question 22 (b)

Criteria	Marks
• Explains the use of breakpoints and single line stepping for this scenario	3
• Explains the use of breakpoints OR single line stepping	2
• Shows some understanding of breakpoints OR single line stepping	1

Sample answer:

Breakpoints are useful when infinite loops occur. The execution of a loop, such as in lines 40–100, can be interrupted so that variables can be inspected. This would show that the value of start never reaches the value of finish.

Single line stepping can show how the contents of variables change with each line of code, demonstrating the unintended values of start.

Question 22 (c)

Criteria	Marks
• Provides correct modifications	3
• Provides partially correct modifications	2
• Shows some understanding of the error	1

Sample answer:

Changes:

25: Set numChars to (finish – start + 1)

40: WHILE numChars > 0

95: Subtract 1 from numChars

Question 23 (a)

Criteria	Marks
• Describes project management techniques and their use in this scenario	3
• Outlines relevant project management techniques	2
• Identifies a relevant project management technique	1

Sample answer

They can use Gantt charts to document required subtasks, estimate timeframes and allocate programmers to tasks.

The chart is updated to show the completion of tasks and to see the impact on the project, dependent tasks and other projects.

They can insist programmers keep regular logs to document their progress. Logs are brought to meetings, which are held regularly between the project manager and their programmers, to check progress and see if support is required.

Question 23 (b)

Criteria	Marks
• Provides differences between the purpose and content of the documentation types	3
• Provides differences between the purpose OR content of the documentation types OR • Shows good understanding of the purpose and content of one of the documentation types	2
• Shows some understanding of one of the documentation types	1

Sample answer:

A user manual helps users learn how to interact with the software. Each section deals with a separate area of functionality. It should have a table of contents, an index and annotated screenshots to illustrate required actions.

A reference manual is for experienced users who wish to know the syntax or specific use of a particular function. It is organised alphabetically and requires an index to allow fast searching for specific terms.

Question 23 (c)

Criteria	Marks
• Describes ways in which the company can ensure better user satisfaction	3
• Outlines a way in which the company can ensure better user satisfaction	2
• Identifies a way in which the company can ensure better user satisfaction	1

Sample answer:

The company should implement a set of quality assurance procedures that clearly define the standards to which the software, interfaces and documentation should adhere. They can introduce regular structured walk-throughs, to identify progress of individual team members. If issues arise the project leader can discuss with management how best to proceed. They should survey users to determine the reasons for dissatisfaction, so they can be addressed.

Question 24 (a)

Criteria	Marks
• Provides substantially correct contents of the array	2
• Shows some understanding of the purpose of the algorithm	1

Sample answer:

The result of the logic in the algorithm is to transpose the array, so it now looks like this.

5	2		
8	6		
1	4		

Question 24 (b)

Criteria	Marks
• Provides substantially correct modifications	3
• Provides partially correct modifications	2
• Shows some understanding of the required logic	1

Sample answer:

Between lines 100 and 110 insert:

```
Number_of_columns = 4
Number_of_rows = 4
FOR col = 1 TO Number_of_columns
    largest = Table (1, col)
    FOR row = 2 TO Number_of_rows – 1
        IF Table (row, col) > largest THEN
            largest = Table (row, col)
        ENDIF
    NEXT
    Table (Number _ of _ rows, col) = largest
NEXT
```

Question 25

Criteria	Marks
• Explains benefits of client/developer communication	3
• Provides some benefits OR explains one benefit of client/developer communication	2
• Provides a benefit of client/developer communication	1

Sample answer:

The developer must talk to the client to understand the problem.

They can work together to refine a prototype to clarify their needs.

The developer must be aware of the client's environment to pre-empt compatibility issues.

After installation the client should evaluate the product to ensure that their needs have been met. Any issues should be communicated to the developer to allow appropriate maintenance.

Question 26

Criteria	Marks
• Describes relevant testing techniques that could have helped prevent the issue	3
• Outlines one relevant testing technique that could have helped prevent the issue	2
• Shows some understanding of the problem	1

Sample answer:

The developer should have predicted the maximum load on the system at peak times, and tested their system with a large volume of data. This should include a mix of transaction types (such as searching, browsing, purchasing items and amending orders) and the use of large files of data for products and orders.

This would have alerted the developer to potential issues with response times and allocated file space before the system went live.

Question 27

Criteria	Marks
• Identifies the syntax errors and explains the illegal syntax	4
• Identifies some syntax errors AND explains some of the illegal syntax	3
• Identifies some syntax errors OR explains the illegal syntax of one error	2
• Shows some understanding of syntax rules	1

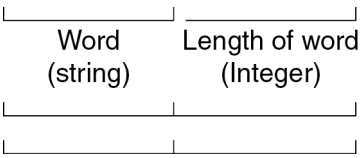
Sample answer:

1. The identifier A is invalid in lines 10 and 70, as it should be terminated by an integer (definition for Identifier)
2. The identifier F1 is invalid, as F is not a valid letter (definition for Letter)
3. The ENDLOOP statement should be followed by a #, as each statement (including repetition) is to be followed by a # (definition for Program)
4. Print is not a valid verb. It should be 'Display'.

Question 28 (a)

Criteria	Marks
• Describes a suitable structure for this scenario	2
• Shows some understanding of the problem	1

Sample answer:

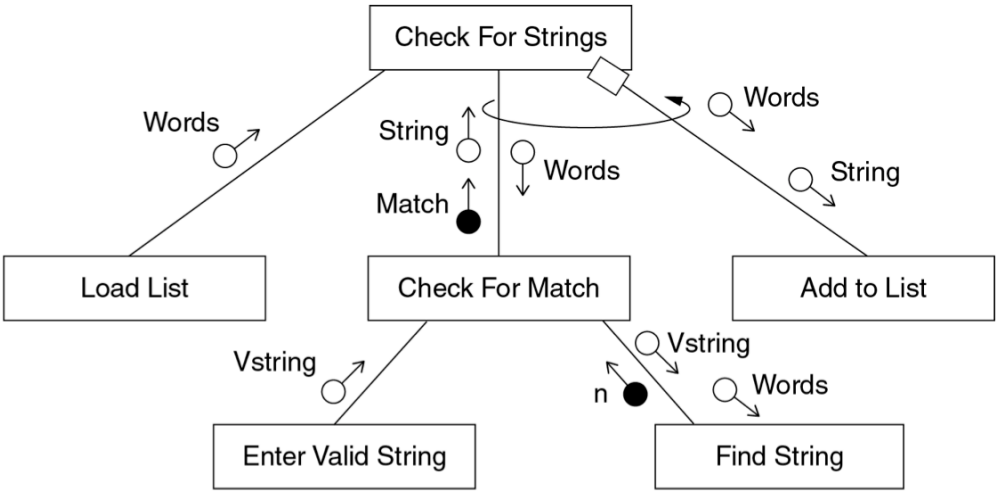


An array of records, with each record consisting of two fields.

Question 28 (b)

Criteria	Marks
• Provides a correct structure chart for this scenario, showing correct parameter passing, a loop, decisions and the relationship of the modules	4
• Provides a substantially correct structure chart for this scenario	3
• Provides some correct elements of a structure chart for this scenario	2
• Provides a modelling tool showing some understanding of the scenario	1

Sample answer:



Question 28 (c)

Criteria	Marks
• Provides a substantially correct algorithm that addresses the following features: – Checks for all letters – Checks for the length – Displays an appropriate error message – Loops until a valid string is provided	4
• Provides a substantially correct algorithm that addresses some of the features	3
• Provides an algorithm that attempts to address some of the features	2
• Shows some understanding of the problem	1

Sample answer:

```
BEGIN EnterValidString
    valid = 0
    REPEAT
        Ask for string
        IF length of string >= 2 THEN
            Checkletters (string, valid)
            IF valid = 0 THEN
                Display “Must be all letters”
            END IF
        ELSE
            Display “Must be correct length”
        END IF
    UNTIL valid = 1
END EnterValidString

BEGIN Checkletters (string, valid)
    valid = 1
    FOR i = 1 to length of string
        Extract ith letter into nextChar
        IF nextChar is NOT a letter THEN
            valid = 0
        END IF
    NEXT i
END Checkletters
```

Question 29

Criteria	Marks
• Describes how the statement is translated with reference to the three steps	3
• Outlines some steps in the translation process	2
• Shows a basic understanding of one of the steps	1

Sample answer:

Lexical analysis: The statement of the source code is scanned and the lexemes Total, =, number1, +, number2 are identified and each allocated a unique token.

Syntactical analysis: This tokenised stream is now scanned against the relevant syntax rules.

Because the statement begins with a variable, it must be an assignment statement, for which a valid rule is:

<Variable> = < variable > < operator> < variable>

It then matches each of the tokens to this structure, with total, number1, number2 being variables, and + being an operator.

Code generation: The machine code statements are generated by converting the tokenised stream to equivalent machine code statements, making use of the available operations in the instruction set of the computer.

Question 30

Criteria	Marks
• Provides a substantially correct algorithm that incorporates the following features: – Generates 10 passwords – Each password – has four characters – starts with a different digit – contains three random uppercase letters – Includes appropriate logic for GetLetter	5
• Provides an algorithm which addresses most of the main features of the algorithm	4
• Provides an algorithm which addresses some of the main features of the algorithm	3
• Provides an algorithm with an attempt to address some aspect of the problem	2
• Shows some understanding of the problem	1

Sample answer:

```
BEGIN GeneratePasscodes
        ‘ REM This builds each passcode
        FOR j = 0 TO 9
            NewCode = STR(j)
            FOR n = 1 TO 3
                Ch = GetLetter(‘A’ , ‘Z’)
                NewCode = NewCode + Ch
            NEXT n
            PassCode(j) = NewCode
        NEXT j
END GeneratePasscodes

BEGIN GetLetter (Start, End)
      Lower = ASC(Start)
      Upper = ASC(End)
      Letter = CHAR(RAND(Lower, Upper))
      Return Letter
END GetLetter
```

Section III

Question 31 (a)

Criteria	Marks
• Correctly describes benefits of encapsulation in OOP	3
• Correctly describes a benefit of encapsulation in OOP OR identifies some benefits of encapsulation in OOP	2
• Shows some understanding of encapsulation in OOP	1

Sample answer:

Encapsulation allows the combining of data and functions/methods into a single object that can easily be used without needing to know its detail.

This allows the definition to be changed without affecting code that uses it.

Question 31 (b) (i)

Criteria	Marks
• Provides the correct fact and rule	3
• Provides a substantially correct extension to the code	2
• Shows an understanding of the logic paradigm syntax	1

Sample answer:

team(volleyball, jim)
same_coach(X, Y, Z) :- coach(A, X), coach(B, X), team(A, Y), team(B, Z), Y ≠ Z

OR

team(volleyball, jim)
same_coach (X, Y, Z) :- is_coached_by (X, Y), is_coached_by (X, Z), Y ≠ Z

Question 31 (b) (ii)

Criteria	Marks
• Shows a good understanding of how this query would be evaluated using forward chaining, with correct reference to all relevant facts and rules	3
• Shows some understanding of how this query would be evaluated with reference to relevant facts and rules	2
• Shows some understanding of the problem	1

Sample answer:

To evaluate is_coached_by(fred, X), it scans the given facts and rules and identifies from the coach facts provided that:

```
coach(soccer, fred)
coach(netball, fred)
```

It then uses the is_coached_by rule:

```
is_coached_by(X, Y) :- coach(A, X), team(A, Y)
```

with fred to find that fred is the coach of jane as jane is in the soccer team due to the fact:

```
team(soccer, jane)
```

It would then find that fred is the coach of ivy and jim as they are members of the netball team due to the facts

```
team(netball, ivy)
team(netball, jim)
```

Through this process, it finds that jim, ivy and jane are coached by fred, which is outputted as the result of the query.

Question 31 (c)

Criteria	Marks
• Provides reasons for the development of TWO paradigms, after the imperative paradigm, with reference to relevant concepts of both	3
• Provides a reason for the development of ONE paradigm, after the imperative paradigm, with reference to a relevant concept	2
• Shows some understanding of programming paradigms	1

Sample answer:

The imperative paradigm was developed first, but did not lend itself to effectively solving all problems, and often required tedious coding. It required multiple definitions of the same property or code each time it was required.

The logic paradigm was developed to solve problems that can be expressed as a set of facts and rules such as in an expert system.

Object Oriented Programming allows the reusability of common objects through encapsulation. Inheritance allows programmers to define the overall characteristics once for a class and only the differences for each sub-class.

Question 31 (d)

Criteria	Marks
• Demonstrates understanding of why heuristics are used in artificial intelligence applications	2
• Shows some understanding of heuristics or artificial intelligence	1

Sample answer:

Heuristics are generic, intuitive rules used to find relevant solutions to problems where a single, clear answer is either too difficult or too time consuming to achieve, or the output generated needs to more closely mimic a human response.

Artificial intelligence systems are often used to solve challenging problems, such as conversing with humans, which do not have one answer that is clearly superior to other answers.

Question 31 (e) (i)

Criteria	Marks
• Effectively demonstrates how the concept of inheritance can be used to improve this code	3
• Shows some understanding of how the code can be improved, with reference to inheritance	2
• Shows some understanding of inheritance	1

Sample answer:

The common sections can be incorporated into a superclass called Ball. The existing classes would then be subclasses of Ball and take its attributes and methods through inheritance.

Modification

```
class Ball {
      private –
            containsAir: Boolean
            cost: float
            setCost( )
      public –
            colour: string
            setBrand ( )
}
```

```
class CricketBall is a ball {
      private –
            seamHeight: float
      public –
            setPieces ( )
}
```

Question 31 (e) (ii)

Criteria	Marks
• Correctly describes the use of polymorphism with reference to the context of the code fragment	3
• Outlines the use of polymorphism with reference to the context of the code fragment	2
• Shows some understanding of polymorphism	1

Sample answer:

OOP allows objects to be processed differently depending on their type or class, even though the method used has the same name.

In the code fragment provided, it could be that the logic required for SetPieces() is different, depending on whether the class is a cricket ball, a golf ball or a basketball. Depending on which sub-class is relevant at run time, the appropriate method for SetPieces() will be used.

Question 32 (a)

Criteria	Marks
• Explains two limitations of ASCII	3
• Provides two relevant limitations OR • Explains one limitation of ASCII	2
• Shows some understanding of the use of ASCII	1

Sample answer:

7-bit ASCII is suitable for representing a limited number (up to 127) of standard keyboard characters (eg digits, letters) and therefore cannot also represent a range of other scripts (Thai, Greek etc).

ASCII represents characters, not numbers such as integers or floating point, and thus is not useful for calculations.

Question 32 (b)

Criteria	Marks
• Provides substantially correct calculations and a valid explanation of the results obtained	3
• Provides substantially correct calculations AND shows some understanding of 8-bit 2's complement data representation	2
• Shows some relevant understanding of binary arithmetic	1

Sample answer:

32 = 00100000 –32 = 11011111 + 1 = 11100000
96 = 01100000 –96 = 10011111 + 1 = 10100000

64 + 64: –32 –96 = –32+(–96):

```
  01000000        11100000
+ 01000000      + 10100000
  10000000        10000000
```

This is –128 This is also –128

In 8-bit 2's complement, the largest positive number that can be stored is 127, the largest negative is –128. 64 + 64 = 128 which is one more than 127, which cannot be represented as an 8-bit 2's complement integer.

Question 32 (c)

Criteria	Marks
• Clearly demonstrates that the two solutions are logically the same	3
• Shows understanding of the use of relevant truth tables OR Boolean algebra	2
• Shows some understanding of the problem	1

Sample answer:

By truth table

			Boolean	Circuit	
A	B	C	OUT	B AND C	A OR (B AND C)
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	1	1	1
1	0	0	1	0	1
1	0	1	1	0	1
1	1	0	1	0	1
1	1	1	1	1	1

These are the same

Or alternatively

By Boolean Algebra

Solution by Circuit is (B AND C) OR A

Boolean expression is $\overline{A}BC + A\overline{B}\,\overline{C} + A\overline{B}\,C + AB\overline{C} + ABC$

$= \overline{A}BC + A\overline{B}\,\overline{C} + A\overline{B}\,C + AB\overline{C} + ABC + ABC$

$= BC(\overline{A} + A) + A\overline{B}\,(\overline{C} + C) + AB(\overline{C} + C)$

$= BC + A\overline{B} + AB$

$= BC + A(\overline{B} + B)$

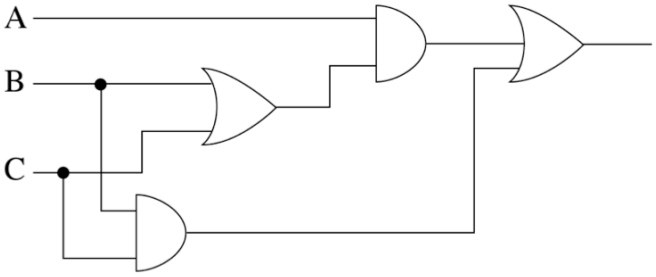
$= BC + A$

= (B AND C) OR A

Question 32 (d)

Criteria	Marks
• Provides a circuit that meets the requirements	3
• Provides a substantially correct circuit OR a correct truth table for the requirements	2
• Shows some understanding of the problem	1

Sample answer:



Question 32 (e) (i)

Criteria	Marks
• Provides two correct data streams with justification	3
• Provides substantially correct data streams	2
• Shows some correct analysis of the data streams provided	1

Sample answer:

Vessel 1 has a temperature of 300 degrees with the heater on and a pressure of 120 kPa with the valve closed. Data stream back should turn off the heater and leave the valve as is.

Vessel 2 has a temperature of 80 degrees with the heater off and a pressure of 240 kPa with the valve open. Data stream sent back should turn on the heater and leave the valve as is.

Therefore the data streams are:

1 01 01 00 1 and 1 10 10 00 1.

Question 32 (e) (ii)

Criteria	Marks
• Provides the necessary changes to the data streams	2
• Shows some understanding of the problem	1

Sample answer:

One extra bit is needed for Vessel ID for both streams and one extra bit is needed for Temperature in the data stream from the vessels.

Question 32 (e) (iii)

Criteria	Marks
• Provides a substantially correct algorithm that addresses the following features: input a stream, isolate the relevant bits, convert to a decimal number, display the pressure	3
• Provides a substantially correct algorithm that attempts to address most of the following features: input a stream, isolate the relevant bits, convert to a decimal number, display the pressure	2
• Provides an algorithm that shows some understanding of the problem	1

Sample answer:

Solution:

```
BEGIN
        get next stream
        pressure = 0
        FOR loop = 12 to 9 STEP – 1
            index = 12 – loop
            pressure = pressure + value of loop^th character of stream x 2^index
        NEXT
        Display 'Pressure =' pressure
END
```

2018 HSC Software Design and Development Mapping Grid

Section I

Question	Marks	Content	Syllabus outcomes
1	1	9.2.2 Output from code fragment	H4.2
2	1	9.2.4 Test data	H4.2
3	1	9.2.3 Error types	H4.2
4	1	9.2.5 Maintenance	H4.3
5	1	9.1.1 Code of conduct	H3.1
6	1	9.1.1 Software licence	H3.2
7	1	9.2.1 Data types	H4.2
8	1	9.2.1 Documentation (DFD)	H1.3, H6.2
9	1	9.2.2 Interface design elements	H6.4
10	1	9.2.3 Stubs	H5.1
11	1	9.2.2, 9.2.4 Benchmarking	H4.2
12	1	9.2.3 Language syntax	H4.2
13	1	9.2.2 Searches	H4.1, H4.2
14	1	9.2.2 Algorithm (flowchart to pseudocode)	H4.2
15	1	9.2.4 Testing	H4.2, H4.3
16	1	9.2.3 Fetch / Execute cycle	H1.3
17	1	9.2.3 Control Structure / interpreting algorithm	H4.2
18	1	9.2.2 Sorts	H4.1, H4.2
19	1	9.2.2 Algorithm (element deletion)	H4.2
20	1	9.2.2 Drivers	H4.3

Section II

Question	Marks	Content	Syllabus outcomes
21 (a)	3	9.1.1 Relevant social/ethical issues	H3.1
21 (b)	3	9.1.2 Development approach	H4.2
21 (c)	3	9.1.2 Installation method	H4.2, H1.2
22 (a)	3	9.2.2 Deskchecking	H4.3
22 (b)	3	9.2.3 Debugging techniques other than deskcheck	H5.3
22 (c)	3	9.2.2 Error correction	H4.2
23 (a)	3	9.3 Project management	H5.1, H6.3
23 (b)	3	9.2.3 User documentation	H5.2, H6.2
23 (c)	3	9.2.1, 9.2.4 Testing / quality assurance	H4.3
24 (a)	2	9.2.2 Purpose of a particular piece of code	H4.2
24 (b)	3	9.2.2 Modify an algorithm	H4.2

Question	Marks	Content	Syllabus outcomes
25	3	9.2.1 Communication between developers and clients	H5.1, H6.2
26	3	9.2.4 Importance of live and volume testing	H5.1
27	4	9.2.3 Syntax diagram: explain why and where code will result in syntax error	H1.2, H4.2
28 (a)	2	9.2.2 Design appropriate data structure	H1.3
28 (b)	4	9.2.2 Produce system documentation	H5.2
28 (c)	4	9.2.2 Define data validation procedure	H4.2
29	3	9.2.3 Translation process	H1.1
30	5	9.2.2 Design an algorithm	H4.2

Section III

Question	Marks	Content	Syllabus outcomes
31 (a)	3	9.4.1 OOP encapsulation	H2.1, H4.2
31 (b) (i)	3	9.4.1 Logic Paradigm – expanding code	H4.2
31 (b) (ii)	3	9.4.1 Logic Paradigm – evaluating query	H4.2
31 (c)	3	9.4.1 Reasons for paradigm development	H2.1, H1.2
31 (d)	2	9.4.1 Heuristics	H1.2
31 (e) (i)	3	9.4.1 OOP modifying code – inheritance	H4.2
31 (e) (ii)	3	9.4.1 OOP polymorphism	H4.2
32 (a)	3	9.4.2 Data representation	H1.3
32 (b)	3	9.4.2 Binary arithmetic	H1.3
32 (c)	3	9.4.2 Logic circuit – comparing solutions	H1.1
32 (d)	3	9.4.2 Logic circuit – design	H1.3
32 (e) (i)	3	9.4.2 Data Stream – interpret and design stream	H1.1
32 (e) (ii)	2	9.4.2 Data Stream – modify structure	H1.1
32 (e) (iii)	3	9.4.2 Algorithm for isolating / interpreting specific bits	H4.2