BOSTES | BOARD OF STUDIES TEACHING & EDUCATIONAL STANDARDS NSW

2015 HSC Software Design and Development Marking Guidelines

Section I

Multiple-choice Answer Key

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

Section II

Question 21

Criteria	Marks
• Outlines two relevant reasons	2
• Provides one relevant reason	1

Sample answer:

- Correct previously undetected errors
- Implement new requirements due to changes in the client's circumstances

Question 22

Criteria	Marks
• Clearly describes method(s) for testing a completed program before its release	4
• Outlines how a completed program can be tested	3
• Outlines a way in which software can be tested	2
• Identifies a feature of software testing	1

Sample answer:

Test in the user's computing environment, in the form of a beta testing involving a significant number of testers. This could ensure that the program is tested with real live data that is unavailable in the developer's environment.

Load testing could also be used, where multiple users access the program at once to see if the program can handle the amount of requests from each user.

Question 23 (a)

Criteria	Marks
• Provides a comprehensive list of the information required	3
• Lists some information required	2
• Identifies a feature of library routines or documentation	1

Sample answer:

The programmer will need to know what the routine achieves and how it is to be called – this includes its name, and the number, type, order and purpose of parameters passed to it and returned from it.

Question 23 (b)

Criteria	Marks
• Provides a substantially correct algorithm that achieves the purpose	4
• Demonstrates some understanding of the use of CharTally AND at least one of arrays, looping or accumulation	3
• Provides some features of the algorithm	2
• Demonstrates some understanding of the problem	1

Sample answer:

```
BEGIN FindLetter
    total = 0
    FOR loop = 1 TO 20
        CharTally(Students(loop), "f", num)
        total = total + num
    NEXT loop
    display total
END
```

Question 24

Criteria	Marks
• Clearly indicates how the effects of runtime and logic errors are different, and provides an example of each type of error	3
• Shows some understanding of runtime and/or logic errors	2
• Identifies a feature of runtime or logic errors	1

Sample answer:

Runtime errors (such as division by zero) cause a software application to terminate unexpectedly. A logic error (such as an incorrect formula) will not crash the software, but will produce incorrect output.

Question 25

Criteria	Marks
• Clearly explains why free updates are provided	3
• Outlines reason(s) for updates	2
• Identifies a reason for updates	1

Sample answer:

Software developers may provide free updates in order to gain customer loyalty and acquire a larger market share. Also, updates are a way of overcoming identified weaknesses in their software so as not to harm their reputation. This service may also be expected by their customers.

Question 26

Criteria	Marks
• Provides an algorithm that correctly addresses both issues	4
• Demonstrates some understanding of how to use a flag and/or a counter and/or a loop and/or a selection and/or features of equivalent merit to address the issue(s)	2–3
• Demonstrates a basic understanding of how one of the issues can be addressed	1

Sample answer:

```
BEGIN
        Get MemberID
        Found = False
        Counter = 0
        WHILE Found = False AND Counter < 200
                Counter = Counter + 1
                IF MemberID = Members(Counter).ID THEN
                        Found = True
                END IF
        ENDWHILE
        IF Found THEN
                Display Members(Counter).Name
        ELSE
                Display “InvalidID”
        ENDIF
END
```

Question 27

Criteria	Marks
• Outlines effects on software development	3
• Identifies effects on software development OR • Outlines an effect on software development	2
• Identifies one effect on software development	1

Sample answer:

Developers are forced to consider how their interfaces will function on a variety of screen sizes. They will also need to allow for a range of internet connection speeds for the mobile devices. The variety of devices also forces developers to consider the different operating systems they may have to accommodate.

Question 28 (a)

Criteria	Marks
• Provides strengths and/or weaknesses of both development approaches in relation to the system	4
• Provides strengths and/or weaknesses of both development approaches OR • Provides strengths and/or weaknesses of one of the development approaches in relation to the system	3
• Identifies features of prototyping and/or rapid application development	2
• Identifies a feature of prototyping or rapid application development	1

Sample answer:

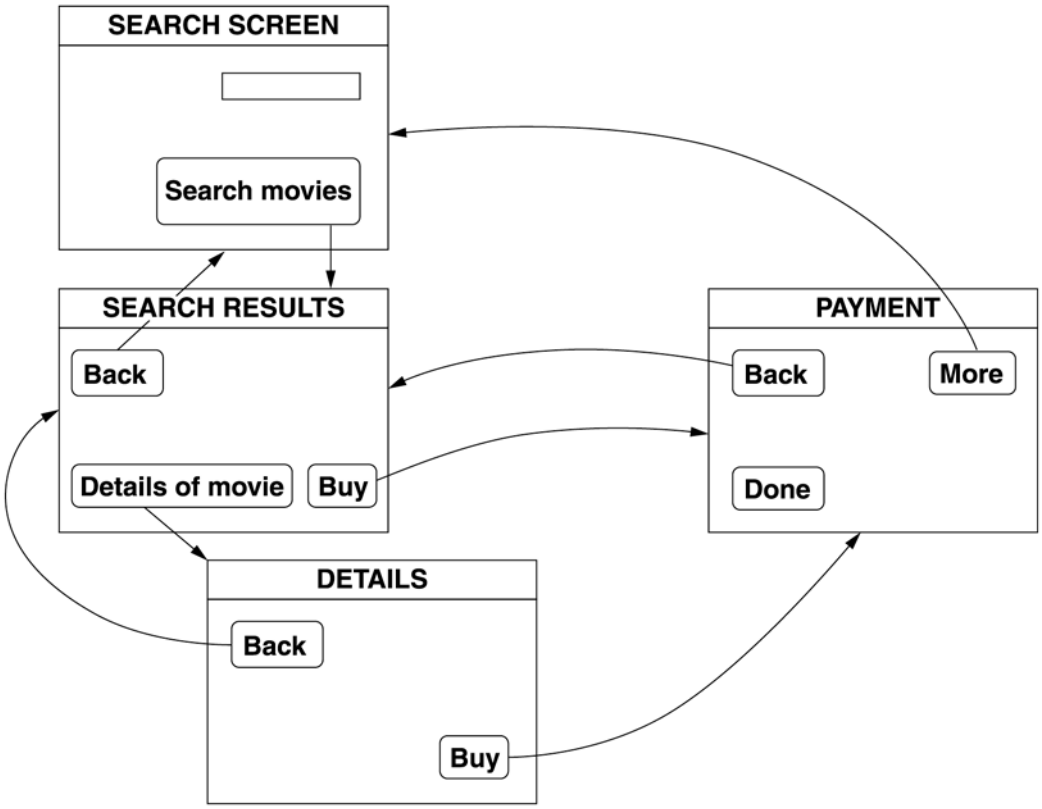
Prototyping is applicable to the USB Kiosk project because it will involve prospective users providing feedback on early working models, particularly with respect to the interface. This has the advantage of ensuring that the final interface design is more likely to meet user needs.

RAD is applicable for the project as many of the processes involved such as credit card transactions, will already exist in similar products and may have been well tested. The reuse of the code will reduce the amount of time needed to be spent on coding and testing.

Question 28 (b)

Criteria	Marks
• Provides a substantially correct storyboard	3
• Provides a storyboard that incorporates some features of the interface	2
• Identifies some features of storyboarding	1

Sample answer:



Question 29 (a)

Criteria	Marks
• Shows a good understanding of the purpose	2
• Shows some understanding of the purpose	1

Sample answer:

The Main module enables the user to see and make choices from the menu and then to access the veg & meat modules repeatedly until deciding to 'finish', whereupon the total is displayed.

Question 29 (b)

Criteria	Marks
• Provides an explanation of the purpose of the stubs in developing the program	3
• Outlines a function of the stubs OR • Identifies some functions of the stubs	2
• Shows some understanding of the problem	1

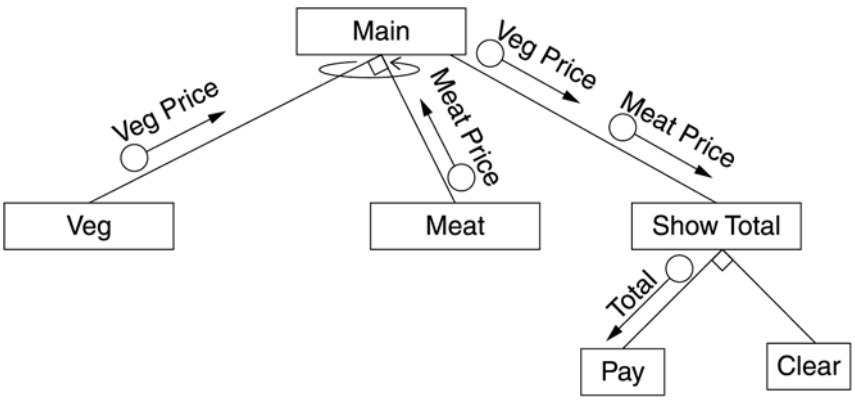
Sample answer:

Stubs provide a way of testing the operation or navigation of a solution without having to fully develop modules. In this algorithm the stub modules enable the testing of the logic in the main module using output statements. Stubs also enable checking of parameter passing. In this algorithm the stubs provide values for passed parameters such as VegPrice & MeatPrice and Total.

Question 29 (c)

Criteria	Marks
• Provides a substantially correct structure chart	4
• Provides a structure chart that addresses the main parts of the algorithm using substantially correct symbols	3
• Provides a diagram that addresses part of the problem using some correct structure chart symbols	2
• Identifies a feature of structure charts OR • Shows some understanding of the problem	1

Sample answer:



Question 30 (a)

Criteria	Marks
• Provides a reason for using arrays of records in the scenario	2
• Identifies a feature of arrays of records	1

Sample answer:

The fields store different types of data about the cars. Each car is stored in a different record (hence array of records). Records are required for the cars because the fields have different data types, for example string for Model and Integer for year of manufacture.

Question 30 (b)

Criteria	Marks
• Provides argument(s) for and/or against sorting the data	3
• Identifies points for and/or against sorting the data OR • Outlines a point for or against sorting the data	2
• Identifies a point for or against sorting the data	1

Sample answer:

Even though searching can be more efficient using a binary search on sorted data, searches in this database will often be on combinations of fields, so sorting on one of them will not speed up the process or provide any benefits. If we were to sort this array of records we would only be able to sort one field. Which one field would be most appropriate? There are none that are more relevant than the others.

Question 31 (a)

Criteria	Marks
• Provides a substantially correct deskcheck	3
• Provides a partially correct deskcheck	2
• Identifies a feature of the deskcheck process	1

Sample answer:

Deskcheck

SerialNumber	Counter	Valid	Display
#1k3#		FALSE	
	2	TRUE	
	3	FALSE	
	4	TRUE	
	5	FALSE	FALSE

Question 31 (b)

Criteria	Marks
• Modifies the algorithm so that it is substantially correct	3
• Makes at least one relevant change to the algorithm	2
• Identifies an error in this algorithm	1

Sample answer:

```
BEGIN
    Input SerialNumber and store in Serial( )
    Valid = True
    IF (Length of SerialNumber is 5)  AND  (Serial(1) is "#")  AND  (Serial(5) is "#")  THEN
        FOR Counter = 2 to 4
            IF Serial(Counter) is NOT a digit THEN
                Valid = False
            ENDIF
        NEXT Counter
    ELSE
        Valid = False
    ENDIF
    Display Valid
END
```

Alternate answer:

```
BEGIN
    Input SerialNumber and store in Serial( )
    Valid = True
    IF (Length of SerialNumber is 5)  AND  (Serial(1) is "#")
        Counter = 2
        WHILE Counter <= 4 AND Valid = True
            IF Serial(Counter) is NOT a digit THEN
                Valid = False
            ENDIF
            Counter = Counter + 1
        END WHILE
    ELSE
        Valid = False
    ENDIF

IF (Serial(5) is not "#")  THEN
    Valid = False
END IF
Display Valid
END
```

Question 31 (c)

Criteria	Marks
• Provides a list of test data that can be used to identify the different possible errors • Includes justification for each item	3
• Provides some relevant test data with some justification	2
• Identifies a feature of test data	1

Sample answer:

Item	Test the case where:
#1234#	the string is too long
@123#	the 1st character is not #
#123@	the 5th character is not #
#a23#	a character is not a digit
#12#	the string is too short

Question 32

Criteria	Marks
• Provides a substantially correct algorithm that uses appropriate control and data structures	4
• Provides an algorithm that addresses some aspects of the problem using appropriate control and data structures	3
• Provides an algorithm that shows some understanding of data structures AND/OR control structures	2
• Shows some understanding of the problem	1

Sample answer:

```
1    BEGIN DealCards
2        FOR A = 0 TO 6
3            Player1(A + 1) = Cards(A * 4 + 1)
4            Player2(A + 1) = Cards(A * 4 + 2)
5            Player3(A + 1) = Cards(A * 4 + 3)
6            Player4(A + 1) = Cards(A * 4 + 4)
7        NEXT A
8    END
```

Section III

Question 33 (a)

Criteria	Marks
• Provides points for and/or against the use of the paradigms in the scenario	4
• Provides points for and/or against the paradigms OR • Provides points for and/or against ONE paradigm with reference to the scenario	3
• Identifies features or reasons for the suitability of the object oriented and/or logic paradigms OR • Shows some understanding of ONE paradigm with reference to the scenario	2
• Identifies a feature or suitability of the object oriented paradigm OR the logic paradigm	1

Sample answer:

The logic paradigm provides facts and rules where the developer is not required to design how they are met. Facts would be suitable in this scenario for stating the prerequisite conditions for the activities and the student's interests, whilst rules could be used to determine whether students have met the prerequisites and suggest further activities.

Object oriented would be appropriate for developing the user interface and the actual activities themselves. The types (or genres) of activities could be set as classes and the different levels of activities within these classes could be developed as subclasses which would then make use of inheritance and polymorphism to make coding faster through reusability of code.

Question 33 (b)

Criteria	Marks
• Outlines when heuristics are appropriate AND provides a relevant example	2
• Identifies a feature of heuristics	1

Sample answer:

Heuristics are criteria or principles for deciding which, among several alternative courses of action, promises to be the most effective in order to achieve some goal.

Heuristics are also called "rules of thumb" which employ fuzzy logic to determine the optimal path to take in order to reach a resolution. This is particularly useful when the algorithm to determine the results is not known and requires the use of probability.

Virus scanning software often use heuristics for detecting viruses and other types of malware. As many new viruses are unknown, heuristic scanning is used to look for tell–tale signs of viruses.

Question 33 (c) (i)

Criteria	Marks
• Correctly extends the code	2
• Shows an understanding of logic paradigm facts	1

Sample answer:

```
salary(ling, 50001).
employer(ling, kim).
```

Question 33 (c) (ii)

Criteria	Marks
• Shows correct result with valid reasoning	2
• Provides some relevant information	1

Sample answer:

An employee is considered valuable if they have an employer and also earn a salary over $50000. Hence the statement `valuable_employee (E )` will evaluate to "al" and "kim".

It evaluates to "al" because he is employed by "jay" and has a salary of $51000.

It evaluates to "kim" because she is employed by "jay" and has a salary of $56000.

If the facts from the above question were included then "ling" would also be evaluated as a valuable employee.

Question 33 (d) (i)

Criteria	Marks
• Correctly explains how polymorphism and inheritance can be used for this scenario	4
• Outlines how polymorphism OR inheritance can be used OR • Outlines features of both concepts	3
• Identifies features of polymorphism and/or inheritance	2
• Provides some relevant information	1

Sample answer:

Inheritance is the ability of objects to take on the characteristics (methods and attributes) of their parent class or classes. This encourages modularity and robust code.

Polymorphism is the ability of methods to appear in many forms. In object oriented programming, this means at runtime a method can process data differently depending on the circumstances. The same command can process objects differently depending on their data type, class or number of parameters.

This means that inheritance and polymorphism both relate to classes and their subclasses. In this scenario a subclass such as the following could be added.

```
class SpecialisedComponent {
        is a Component
}
```

Inheritance would then be able to be used in this software, as this subclass would have each attribute and method of the parent class, Component.

Polymorphism can also be used in the development of this software as some methods in the SpecialisedComponent class would have the same name as the corresponding method in the superclass but would be calculated differently.

Question 33 (d) (ii)

Criteria	Marks
• Provides a substantially correct method	3
• Identifies some features of the cost() method	2
• Identifies a feature of the cost() method	1

Sample answer:

```
class Engine {
        is a Component
        public –
                cost():
                        FOR i = 1 TO number of components
                                IF components[ i ].type() = “standard engine” THEN
                                        components[ i ].cost() = 2000
                                END IF
                        NEXT i
}

class V8Engine {
        is a Engine
        public –
                cost():
                        FOR i = 1 TO number of components
                                IF components[ i ].type() = “V8 engine” THEN
                                        components[ i ].cost() = 2800
                                END IF
                        NEXT i
}
```

Question 33 (e)

Criteria	Marks
• Explains the consequences of the error AND shows how to correct the error	3
• Outlines consequence(s) AND/OR outlines how to correct the error	2
• Identifies the error or a consequence	1

Sample answer:

Encapsulation is the process of including all the attributes and procedures that an object needs within the object itself. It involves hiding an object's data and processes from its environment meaning that only the object can alter its own data.

In this section of code the sort() and swap() methods are in the public section of the SortedArray class, which allows them to be accessed by modules from outside of the class. This compromises the encapsulation of the methods as it means that there is no longer a guarantee that the array will always be in sorted order.

This error can be fixed by moving these two methods into the private section of the method.

Question 34 (a)

Criteria	Marks
• Provides correct working	2
• Identifies a feature of binary multiplication	1

Sample answer:

```
  10111 x
    111
  10111
 101110
1011100
10100001
```

Question 34 (b)

Criteria	Marks
• Outlines relevant benefit(s)	2
• Identifies a feature of 2's complement	1

Sample answer:

The 2's complement representation allows the largest possible range of both negative and positive integers as well as enabling their subtraction by adding complements (negatives).

Question 34 (c) (i)

Criteria	Marks
• Correctly describes what is achieved	3
• Shows some understanding of hexadecimal to binary conversion and provides some interpretation of the data stream	2
• Provides some relevant information	1

Sample answer:

The tool moves to X = 45, Y = 16 without cutting.

Working:

D		6	A	0
1101		0110	1010	0000
11	0	10 1101	010000	0
START	TOOL	X = 45	Y = 16	STOP

Question 34 (c) (ii)

Criteria	Marks
• Provides a substantially correct data stream	3
• Provides a partially correct data stream OR • Shows some understanding of how to work out the data stream	2
• Shows an understanding of the problem	1

Sample answer:

1	1	1	0	0	1	1	1	1	0	0	1	1	1	1	0
1	1	0	0	1	1	1	1	0	0	0	1	1	1	1	0
1	1	1	0	1	1	1	1	0	0	1	1	1	1	0	0

Question 34 (d)

Criteria	Marks
• Identifies all THREE components and outlines how to represent –7.5 in floating point notation	4
• Identifies at least TWO components and shows some steps for how to represent –7.5 in floating point notation	3
• Identifies at least ONE component and shows some understanding of how to represent a decimal fraction in floating point notation OR • Identifies all THREE components	2
• Identifies a feature of floating point notation	1

Sample answer:

The three components are sign bit, exponent and mantissa.

The sign bit is 1 because –7.5 is negative.

$111.1 = 1.111 \times 2^2$

So the exponent is the binary equivalence of 127 + 2 (the index).

The mantissa is the three ones after the point followed by 20 zeros to fill the 23 bits.

Question 34 (e) (i)

Criteria	Marks
• Draws a substantially correct truth table	3
• Draws a partially correct truth table	2
• Shows some understanding of the problem	1

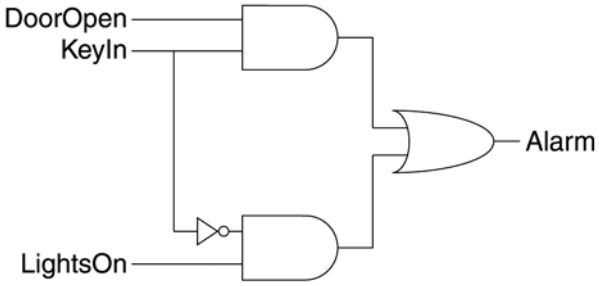
Sample answer:

Key	Door	Lights	Alarm
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Question 34 (e) (ii)

Criteria	Marks
• Draws a correct circuit	3
• Solves part of the problem	2
• Shows some understanding of the problem	1

Sample answer:



2015 HSC Software Design and Development Mapping Grid

Section I

Question	Marks	Content	Syllabus outcomes
1	1	9.2.1 Needs of client	H4.1, H6.1
2	1	9.2.2 Screen elements	H6.4
3	1	9.2.1 Documentation	H5.2
4	1	9.1.1 Intellectual property	H3.1
5	1	9.2.5 Maintenance / documentation	H5.2, H6.1
6	1	9.2.2 Pseudocode output	H4.3
7	1	9.1.1 Inclusivity	H3.2, H6.4
8	1	9.1.2 Implementation methods	H6.3
9	1	9.2.2 Data types / purpose	H1.2
10	1	9.3 Project management	H6.2
11	1	9.2.2 Flowchart control structures	H5.1
12	1	9.2.2 Flowchart output	H4.3
13	1	9.2.3 Fetch/execute cycle	H1.1
14	1	9.2.3 Metalanguages	H2.1
15	1	9.1.1 Reverse engineering	H3.1
16	1	9.2.2 Completing an algorithm	H5.3
17	1	9.2.2 Sort types	H1.2
18	1	9.2.2 Replacing a WHILE loop	H4.2
19	1	9.2.2 Data types in an array	H1.3
20	1	9.2.4 Driver to test module	H5.3

Section II

Question	Marks	Content	Syllabus outcomes
21	2	9.2.5 Reasons for maintenance	H5.1
22	4	9.2.4 Describe the types of testing	H4.2
23 (a)	3	9.2.2 Documentation for library routine	H4.2
23 (b)	4	9.2.2 Use documented subroutines in an algorithm.	H6.1
24	3	9.2.3 Runtime errors	H1.1
25	3	9.1.1 S/W market	H3.1
26	4	9.2.2 Linear search for a unique element	H4.2
27	3	9.1.2 Trends in software development	H2.2
28 (a)	4	9.1.2 Development Approaches	H1.2
28 (b)	3	9.2.1 System modelling tool	H6.4
29 (a)	2	9.2.2 Algorithm module interpretation	H4.1
29 (b)	3	9.2.3 Use of stubs	H5.3
29 (c)	4	9.2.1 Structure Chart	H6.3
30 (a)	2	9.2.1 Array of records	H4.2
30 (b)	3	9.2.2 Planning and designing a solution (sorting)	H4.2, H4.3
31 (a)	3	9.2.2 Desk checking	H4.2
31 (b)	3	9.2.2 Error correction	H4.2
31 (c)	3	9.2.2 Test data	H4.2
32	4	9.2.3 Algorithm design	H4.2

Section III

Question	Marks	Content	Syllabus outcomes
33 (a)	4	9.4.1 Justify paradigms	H1.2, H2.1, H4.1
33 (b)	2	9.4.1 Heuristics	H1.2, H4.1
33 (c) (i)	2	9.4.1 Modify logic fragment	H1.2, H4.2
33 (c) (ii)	2	9.4.1 Interpret logic code	H1.2, H4.2
33 (d) (i)	4	9.4.1 Polymorphism and inheritance	H1.2, H4.2
33 (d) (ii)	3	9.4.1 Modify OOP fragment	H1.2, H4.2
33 (e)	3	9.4.1 Encapsulation	H4.2
34 (a)	2	9.4.2 Binary arithmetic	H1.3
34 (b)	2	9.4.2 2's complement	H1.3
34 (c) (i)	3	9.4.2 Interpretation of data stream	H1.1
34 (c) (ii)	3	9.4.2 Generate data stream	H1.1
34 (d)	4	9.4.2 Floating point representation	H1.3
34 (e) (i)	3	9.4.2 Truth table	H1.1
34 (e) (ii)	3	9.4.2 Circuit design	H1.1