Question 1

Students performed quite well in this section of the paper with very few “t”s being given. Most students could answer at least most of Question 1, some of 2 and at least set up the character initially in 3. It is evident that students need to read questions more carefully before writing an answer (I am aware of the amount of reading required in this section).

Answered very well in general. Some students did not show the change of Feet2.

- **Initially**
  - yard1 = 0
  - feet1 = 0
  - yard2 = 0
  - feet2 = 0

- **When a number is entered into “Number of yards 1” TextField**
  - Set yard1 to value in “Number of yards 1” TextField

- **When a number is entered into “Number of feet 1” TextField**
  - Set feet1 to value in “Number of feet 1” TextField
  - if feet1 >= 3
    - Set feet1 = 2
    - Display “feet1 is” feet1

- **When a number is entered into “Number of yards 2” TextField**
  - Set yard2 to value in “Number of yards 2” TextField

- **When a number is entered into “Number of feet 2” TextField**
  - Set feet2 to value in “Number of feet 2” TextField
  - if feet2 >= 3
    - Set feet2 = 2
    - Display “feet1 is” feet2

- **When the “Addition” button is pressed**
  - Set yard2 = yard2 + yard1
  - Set feet2 = feet2 + feet1
  - if (feet2 >= 3)
    - Set feet2 = feet2 - 3
    - Set yard2 = yard2 + 1
  - Set yard1 = 0
  - Set feet1 = 0
  - Display “total yards” yard2
  - Display “total feet” feet2
Question 2

(a) A good number of students recognised that the purpose of the variable *used* is to stop the user pressing the **Set advance** button until a favourable number is obtained.

(b) In part (i) students used equality to 20 rather than \( \geq 20 \) (recognising that they number advanced could vary) in many instances and in part (ii) lots of students didn’t realise that the position, once the capture had been made, decreases towards the base. Some students created a variable to indicate capture and used this to help determine win.

**Initially**

\begin{align*}
\text{advance} &= 0 \\
\text{piece} &= 0 \\
\text{count} &= 0 \\
\text{direction} &= 1 \\
\text{used} &= \text{true}
\end{align*}

Set enemy to a random value between 1 and 18

Display “enemy position” enemy

When the “Set advance” button is pressed

if used equals true

Set advance to a random value between 1 and 5
Set used = false
Set count = count + 1
Display “Next Advance” advance

When the “Move” button is pressed

Set used = true

Set piece = piece + advance * direction
Set advance = 0
if piece \( \geq 20 \)

\begin{align*}
\text{piece} &= 0 \\
\end{align*}

if piece equals enemy

set direction = direction * -1
Display “piece position” piece

if direction equals -1 and piece <= 0

Display “Finished in “ count “ turns”

Question 3

Students were able to set up the initial position depending on the character selected but the defining part was being able to deal with the addition/subtraction of characteristics values in conjunction with the limitations that were given. Many students had plus and minus buttons on the characteristics which gives greater scope/difficulty in matching the requirements.

**Initially**

\begin{align*}
\text{strength} &= 3 \\
\text{dexterity} &= 2 \\
\text{intelligence} &= 5 \\
\text{incStrength} &= 0 \\
\text{incDexterity} &= 0 \\
\text{incIntelligence} &= 0 \\
\text{race} &= \text{'H'} \\
\text{points} &= 20
\end{align*}

When a number is entered into “Increase Strength” TextField

Set temp to value in “Increase Strength” TextField
if \( \text{temp + incDexterity + incIntelligence} \leq 20 \) and temp \( \leq 10 \)
Set incStrength = temp;
Set points = 20 - (incStrength + incDexterity + incIntelligence)
Display “Points ” points
Display “Increase Strength ” incStrength

**When a number is entered into “Increase Dexterity” TextField**
Set temp to value in “Increase Dexterity” TextField
if ( incStrength + temp + incIntelligence) <= 20 and temp <= 10
  Set incDexterity = temp;
  Set points = 20 - (incStrength + incDexterity + incIntelligence)
  Display “Points ” points
Display “Increase Dexterity ” incDexterity

**When a number is entered into “Increase Intelligence” TextField**
Set temp to value in “Increase Intelligence” TextField
if (incStrength + incDexterity + temp) <= 20 and temp <= 10
  Set incIntelligence = temp;
  Set points = 20 - (incStrength + incDexterity + incIntelligence)
  Display “Points ” points
Display “Increase Intelligence ” incIntelligence

**When a text is entered into “Race” TextField**
Set race to value in “Race” TextField
if race equals ‘H’
  Set strength = 3
  Set dexterity = 2
  Set intelligence = 5
  Set points = 20
if race equals ‘O’
  Set strength = 9
  Set dexterity = 1
  Set intelligence = 0
  Set points = 20
if race equals ‘E’
  Set strength = 1
  Set dexterity = 5
  Set intelligence = 4
  Set points = 20
Display “Race ” race
Display “Strength ” strength
Display “Dexterity ” dexterity
Display “Intelligence ” intelligence
Display “Points ” points

**When the “Accept” button is pressed**
Set strength = strength + incStrength
Set dexterity = dexterity + incDexterity
Set intelligence = intelligence + incIntelligence
Set incIntelligence = 0
Set incStrength = 0
Set incDexterity = 0
Set points = 0
Display “Strength ” strength
Display “Dexterity ” dexterity
Display “Intelligence ” intelligence
Display “Points ” points
Display “Increase Strength ” incStrength
Display “Increase Dexterity ” incDexterity
Display “Increase Intelligence” inclIntelligence

Question 4

(i) Well answered although a small number of students erroneously thought that the left to right rule took precedence over BODMAS.

(ii) Well answered. The marker expected an answer of 13.0 to demonstrate that the result of the calculation was a double. Students are reminded to use the information booklet when uncertain about a mathematical function and that double will not multiply by 2!

(a) (i) \( a = 25 \times (6+20-1) \)
(ii) \( b = 13.0 \times (5.0+8.0) \)

Examiner Comment:
(i) Well answered.
(ii) Well answered.

(b) (i) \( c = 5 \times (2+3) \)
(ii) \( d = 12 \times (2*6) \)

Examiner Comment:
Well answered by about half the students but the operation of the % function is not well understood by many candidates. Students should bear in mind that the trace table always provides more lines than are required and they should not be fooled into using every space.

(c) \( e = 2 \)

Question 5

(i) Well answered.

(ii) Poorly answered with most students giving an answer of -1 after erroneously thinking that \((\text{int})('3') = 3\) rather than its ASCII value of 51. Fortunately most students were wise enough to show their working and received credit for rounding 3.6 to 4.

(a) (i) \( f = 6 \)
(ii) \( h = 47 \times (51-4) \)

Generally well answered but the marker found it difficult to give marks when the student gave just an incorrect value for \( k \) without any indication of how it was obtained. The trace table could have been more helpful. Most students realised the \( z[-1] \) is not a valid array location.

(b) (i) \( k = 15 \times (2+(2+5)+(5+1)) \)
(ii) if the statement \( k = k + z[i]+ z[i-1] \); was executed when \( i == 0 \) then it would attempt to access the array with an index of -1 which would produce an “Array index out of bounds” error.
Students appear to need more practice with tracing code as many were unable to cope with a nested loop and could not list the sequence of values of i and j.

(c) \( m = 8 \)

**Question 6**

This question proved inaccessible to the majority of candidates. Once again, more practice with tracing code with array's is required if students wish to access the A question.

(a)

<table>
<thead>
<tr>
<th>colony</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6 7 8</td>
<td>0 1 2 3 4 5 6 7 8</td>
</tr>
<tr>
<td>D D L L D D D D D D D D K=1</td>
<td>D D D D D D D D D D D D K=2</td>
</tr>
<tr>
<td>D D D D D D D D D D D D D D D D D D K=3</td>
<td></td>
</tr>
</tbody>
</table>

Or …. D becomes L if it has and L both to its left and right. L remains L if it has exactly one L to its left or to its right (both sides changes L to D)

(ii)

```
D L D L D D L L L
D D L D D D L L L
D D D D D D L L L
```

Once again, few students attempted (b) and (c) which was disappointing as they could be answered without understanding the code. Students do not have a clear understanding of parameters and the nature of passing objects by reference.

(b) It would not affect the operation of the applet as the arrays are called as parameters to the other methods and since they are objects not simple data types the reference to the parameters will change the data in the original arrays declared in the paint method.

(c) The first code segment will store into the locations of the array colony the corresponding elements in the array status. The second code segment will point the variable array variable name colony at the array elements of the array status. This means there will be one array of data with two names resulting in both being full of 'D’s.'

**Question 7**

(a)

aField: Green
bField: Red
lockField: Locked

(b)

aField: Green
bField: Red
lockField: Locked
(c)
    aField: Red
    bField: Green
lockField: Unlocked

Answered well.

**Question 8**

(a)

String2: I get
String2: I get dessert

Generally answered well.

Two most common mistakes, both related to substring:

1. Substring takes two parameters, the first indicates where the substring starts, the second indicates where the substring is to finish. This second parameter indicates “go up to, but do not include, this character”
2. Numeric answers were provided, but strings of this nature will not be numeric.

(b) (i)

<table>
<thead>
<tr>
<th></th>
<th>D2</th>
<th>D1</th>
<th></th>
<th>D3</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>T</td>
<td>W</td>
<td>T</td>
<td>F</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
</tr>
<tr>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
</tr>
<tr>
<td>29</td>
<td>30</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>31</td>
</tr>
</tbody>
</table>

D1 and D2 were consistently answered correctly, but a number of students had more difficulty with D3. The most common incorrect date provided for D3 was 8/10, which is 4 weeks from D2 rather than 4 weeks from D1.

(ii) Write Java code to create a CalendarDay and then change the CalanderDay to the 7th November using **only** the addWeeks and addDays class methods.

```java
d1 = new CalendarDay();
d1.addWeeks(5);
d1.addDays(2);
```

To ensure differentiation between students this was marked very strictly to only allow the use of the CalendarDay constructor, addWeeks and addDays methods – the question did phrase this specifically. It was important to be aware that the constructor created a new Calendar Day using the date 1/10. Most common issues were assuming the date would be the first of January, and the use of copyDate and setDate.

**Question 9**

(a)

```java
Hero hero1 = new Hero();
```

Generally answered well. The most common error was that student forgot the declaration, before the instantiation. Eg. `hero1 = new Hero();` → you must specify first that `hero1` will be a Hero object.
(b)  
hero1.setRace('E');  
hero1.addStrength(6);

Most students did well on this, but some forgot to use the object they created eg. setRace('E') doesn’t apply to anything, you must first specify which object you are applying the statement to eg. hero1.setRace('E'); It was also important to indicate that the character would use single quotes ‘E’. Most students realised that an Elf had a strength of 1, therefore 6 needed to be added to get to a strength of 7.

(c) (i)  
public int health()  
{  
    return (200 + 10 * strength);  
}

This was fairly poorly answered, I suspect due to students not reading the question in enough detail.  
Common Issues:  
1. Forgetting the return type: public void ... rather than public int ...  
2. Not specifying the data type where a local variable was used within the method.  
   eg. health = 200+10*strength; return health; BUT health must be declared as an int  
3. Taking strength as a parameter rather than using the strength variable already provided in the Hero class  
4. A number of students wrote the method as setter method, where it was supposed to be a getter method

(ii) int heroHealth = hero1.health();

There were a number of issues with this, most common problems were the use of the health method without applying it to the current object eg. health() instead of hero1.health(); and many students did not declare heroHealth as an integer.

Question 10

(a)  
(i)  
<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>~B</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Done fairly well across most papers.

(ii)  
F = A v ~(B & ~C) v (~A & B & C)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>~C</th>
<th>~A</th>
<th>B &amp; ~C</th>
<th>~(B &amp; ~C)</th>
<th>~A &amp; B &amp; C</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Most of the sections were grouped well but only a few actually came up with the correct answer. Most of the errors were in the section where A and B and C were all 0.

(b) There were plenty of errors made in this section with the most common being the incorrect bracketing of the final answers. Errors were more common in ii than i. Also there was a tendency to leave the H and G completely out of the equation which while minor was looked on as an error.
(a) 

(i) \[ G = \neg(P \land Q) \]

(ii) \[ H = P \lor \neg((Q \land \neg R) \lor (\neg P \lor R)) \]

(c) Java code is compiled to bytecode rather than machine code. Machine code is unique to each computer system and so a different compiler is required for each system. Bytecode is run on a virtual machine (The Java Virtual Machine). This is not dependent on the computer system as can run as a plug in to the internet browser. This means it can run on any computer system without need for recompiling.

This question was exceptionally poorly answered. Answers included that Java was not a high level programming language, it pre-dated compiling or was so simple that it did not need compiling. All of these obviously are completely wrong and it appears that the students simply were not aware of how or why Java operates in this fashion.

**Question 11**

Most people were able to come up with the patterns in i but many students omitted entering 0’s into the Karnaugh map that meant it was incomplete. The student was looked favourably on if the 1’s were in a correct pattern but if a student completed it with 0’s they were preferred over those that had not. Question ii was done very poorly with very few students able to simplify it down to the most simple stated defined in the answers below. A lot of this was the focus on extraneous variables for example where it should have been simplified to \((\neg C \land \neg D)\) many actually contained \((\neg B \land \neg C \land \neg D)\) which still gives the correct answer but not in the most simplified form. Some students used the logic laws to take the \(\neg\) from the brackets and where this was done they did not affect the operators correctly in most cases and ended up getting it wrong. For iii most logic diagrams were well done and represented what answer they had given in ii.

(a) 

(i) 

(ii) & (iii) \[ J = (\neg C \land \neg D) \lor (B \land \neg C) \lor (\neg A \land \neg B \land \neg D) \]

(b) 

(i) 

if \((a == 1) \&\& ((b == 2) || (c == 3))) \quad \text{[logic Law 8]}

\[ c = 1; \]

else

\[ c = 4; \]

This was done very well in the most part.

(ii) \[ W = (D \land K) \lor (\neg S \land D \land L) \]
Very few students answered this incorrectly. 
(iii) The multiplication is done by repeated use of the addition instruction in a loop. In this example it would be 
\[ x = x + 5 \] three times. 
Many students focused on left shifting in their answer. The numbers were not such that left shifting was an effective 
use and so the above answer was the most efficient.

Question 12

(a) Most students of all capability were able to clearly define the differences in the code for question i. For question ii 
though the students seem to be troubled by the fact that the TOY code was actually the machine version of the Java 
code. The students were less confident being able to explain that connection in their answers and therefore it was 
poorly answered in the most part.

(i) In Java code 1 the if statement has a condition of \( a > 3 \) and the TOY code 1 has the line \( R[C] \leftarrow R[A] - R[1] \). In Java code 2 the if statement has a condition of \( a < 3 \) and the TOY code 1 has the 
line \( R[C] \leftarrow R[1] - R[A] \).

(ii) In TOY code 1 the Branch instruction has a branching condition equivalent to \( a - 3 > 0 \) which is 
the same as the Java code 1 if statement condition of \( a > 3 \). In TOY code 2 the Branch instruction 
has a branching condition equivalent to \( 3 - a > 0 \) which is the same as the Java code 2 if statement 
condition of \( a < 3 \).

(b)

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Contents</th>
<th>Pseudocode</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0000</td>
<td>data</td>
<td>The variable ( a )</td>
</tr>
<tr>
<td>01</td>
<td>0000</td>
<td>data</td>
<td>The variable ( b )</td>
</tr>
<tr>
<td>10</td>
<td>8A00</td>
<td>( R[A] \leftarrow \text{mem}[00] )</td>
<td>Set Register A to the contents of location 00</td>
</tr>
<tr>
<td>11</td>
<td>7103</td>
<td>( R[1] \leftarrow 03 )</td>
<td>Set Register 1 to the number 03</td>
</tr>
<tr>
<td>12</td>
<td>2C1A</td>
<td>( R[C] \leftarrow R[1] - R[A] )</td>
<td>Register ( C = \text{Register 1} - \text{Register A} )</td>
</tr>
<tr>
<td>13</td>
<td>DC19</td>
<td>if ( R[C] &gt; 0 ) pc ( \leftarrow 19 )</td>
<td>Branch to address 19 if Register ( C &gt; 0 )</td>
</tr>
<tr>
<td>14</td>
<td>7105</td>
<td>( R[1] \leftarrow 05 )</td>
<td>Set Register 1 to the number 05</td>
</tr>
<tr>
<td>15</td>
<td>2CA1</td>
<td>( R[C] \leftarrow R[A] - R[1] )</td>
<td>Register ( C = \text{Register A} - \text{Register 1} )</td>
</tr>
<tr>
<td>16</td>
<td>DC19</td>
<td>if ( R[C] &gt; 0 ) pc ( \leftarrow 19 )</td>
<td>Branch to address 19 if Register ( C &gt; 0 )</td>
</tr>
<tr>
<td>17</td>
<td>DC19</td>
<td>if ( R[C] == 0 ) pc ( \leftarrow 19 )</td>
<td>Branch to address 19 if Register ( C == 0 )</td>
</tr>
<tr>
<td>18</td>
<td>C01B</td>
<td>if ( R[0] == 0 ) pc ( \leftarrow 18 )</td>
<td>Branch to address 18</td>
</tr>
<tr>
<td>19</td>
<td>7B02</td>
<td>( R[B] \leftarrow 02 )</td>
<td>Set Register B to the number 02</td>
</tr>
<tr>
<td>1A</td>
<td>9401</td>
<td>( \text{mem}[01] \leftarrow R[B] )</td>
<td>Store Register B in location 01</td>
</tr>
<tr>
<td>1B</td>
<td>0000</td>
<td>exit</td>
<td>End of program.</td>
</tr>
</tbody>
</table>

Many students attempted this question and all showed that they had understanding of TOY and how it operated. 
Unfortunately most students did not seem to understand the java expression as the expression was an OR statement. 
Most answers checked the first logic expression and if it were true then checked the second logic expression rather 
than simply conducting the operation required. In fact the setting of the variable only occurred if both were true thus 
turning the expression into an AND statement rather than an OR statement. Other minor errors also included setting 
the variable to 2 regardless of both expressions being false as they did not lead to an exit status in that case and 
continued through the code.
Question 13

(a) 

\[
\begin{array}{c}
1 & 0 & 1 & 1 \\
\hline
+ & 1 & 0 & 1 & 1 & 1 \\
\hline
1 & 0 & 1 & 1 & 0 & 1
\end{array}
\]

Generally answered well.

(b)  

(i) \(A9_{16} = 10 \times 16 + 9 = 169_{10}\)

More than half of the students had trouble with this question. A very common answer was 10+9 = 19. A number of students converted to binary instead of hexadecimal.

(ii) 14 = 001110 so –14 = 110001 + 1 = 110010. The left most bit is the Most Significant Bit.

More than half of the students had trouble with this question.

(iii) 0 to 11111 = 0 to 31. The range of values represented would increase from 0 - 15 to 0 - 31.

Generally answered well.

(c) 

\[
10011 01001 10100 00100
\]

19 9 20 4
S I T D

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Generally answered well.

Question 14

(a)  

(i) Programs will have been written using the ASCII codes. If a different set of codes was used all these programs would need to rewritten.

Answered reasonably well by most students.

(ii) The two bytes could be set up as a 16-bit word. For the existing ASCII character codes the most upper byte will be set to all zeros and the lower byte will contain the ASCII code numbers.

Students who attempted this did well, but most students apparently missed that this answer could be taken almost directly from the Information Booklet.

(b) 

\[
\begin{array}{cccccccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0
\end{array}
\]

A left shift increases the value of each bit to the next higher power of two. This is the same as multiplying the value stored by 2. E.g. the original value is 10 and the value after the left shift is 20.
Generally answered well

(c) \( \text{If } (\text{Math.abs(result1 } - \text{ result2)} < 0.0001 ) \)

Very few students were able to answer this in terms of the limitations of floating-point representation, as specified by the question.

**Question 15**

(a)

(i) File size = sample rate x sample size x time spent sampling
\[= 8000 \times 16 \times 60 \text{ bits.} \]

Generally answered well.

(ii) Compressing the file makes it smaller. This means it takes up less storage space and to faster to transfer. When editing a compressed file must first be uncompressed which takes time slowing down the editing process.

Most students gained partial marks for this part, answering purely in terms of why a compressed file would be better for storage or transfer. Relatively few considered why WAV would be suitable for editing. Of these, most seemed to regard “compressed” as being equivalent to “lossy”.

(b)  
\[
\text{char[ ] [ ] a = new char[2][4]}
\]

Most students were able to provide a declaration for a 2D array in response to this part, but a very common mistake was to declare it as an array of integers or Strings. E.g. \( \text{int[ ] [ ] a = new int[2][4] } \)

(c) (i) \( \text{int[ ] b = new int[5];} \)
(ii) \( \text{b[4] contains 7} \)

This was a particularly challenging part for most students.