Examiners' commentary 2017–2018

CO1112 Creative Computing I: image, sound and motion – Zone A

General remarks

Overall performance on this paper was good, with the average mark being a high second class. About one fifth of the cohort obtained a mark in the first class range, and only a small handful failed the paper.

What follows is a brief discussion of the individual questions on this paper, with hints and explanations of the answers expected by the examiners.

Comments on specific questions

Question 1

General

This was a straightforward question, and all candidates chose to answer it. Performance on this question was reasonably good, and on average in line with overall performance on the paper. However no candidates obtained more than 20 out of the possible 25 marks available, which is disappointing for a question that is straightforward.

Part (a) of this question required candidates to state for each conjecture, whether it was true or false. In general this was done well, though many thought that Andy Warhol had been involved in the Bauhaus work, which was incorrect. Part (b) saw some incorrect guesses, one particularly off-beat suggestion was Ada Lovelace. Marianne Brandt is one correct example, given by many, but other correct examples were also accepted. Part (c) is a simple matrix multiplication that most candidates were able to do.

For part (d), AND and OR are two operators that were most often given. Other acceptable answers would also be << and >>. However, a number of candidates included NOT which is incorrect. Additionally, some candidates gave three operators, and not two, which was asked for. Giving three in this case can give the impression that the candidate is guessing; examiners would only take the first two given, and these were not always the correct two.

Part (e) required audio compression formats; some candidates gave formats only used for video, and although audio could be seen as included within this, it is not a good approach to answer. However most candidates gave appropriate examples.

Part (f) was weakly answered. Examiners wanted to see that candidates understood that three bits in one generation are input to provide a resulting output of one bit for the next generation. Any format that demonstrated this was acceptable, but the below is a good example.

```
111   110    101    100    011    010    001    000
 0     0      0      1      1      1      1      0
```

Answers for part (g) were usually correct if the candidate understood the concept in part (f). The output should look as below:

```
    1
   111
  11 1
 11 1111
11  1   1
```

Finally, for part (h), most candidates listed genes, development and reproduction as important components of a Biomorph, in the context of evolutionary processes.

Question 2

History and creative thinking

This question was a very unpopular one, with only a small handful of candidates choosing it. The average performance on this question was below a pass mark, which is disappointing as much of the material has been seen in the subject guides.

For part (a), examiners were expecting candidates to give some description of the various symbols used (I, V, X, M, C, L, etc.), and how we represent using combinations. An additive example would be VIII, while subtractive would be IX. In this question, including examples is essential, and valuable marks were lost through the omission of these.

Part (b) required a simple description of an algorithm, in enough detail to be able to convert this to code. This is an exercise that is included in the subject guide, and candidates should have already attempted or thought about an answer.

There are a number of options, all of which obtained marks if they were correct. One could be to convert to decimal, show an algorithm for adding decimal (which is quite straightforward) and then convert back.

Another could be the following:

1. Substitute for any subtractives in both values; that is; "uncompact" the Roman values;
2. Put the two values together, i.e. concatenate them;
3. Sort the symbols in order from left-to-right with the "largest" symbols (i.e. the ones representing the biggest numbers) on the left;
4. Starting with the right end, combine groups of the same symbols that can make a "larger" one and substitute the single larger one;
5. Compact the result by substituting subtractives where possible.

Marks were allocated for being able to come up with a reasonable algorithm; and for correctness, elegance and detail.

Part (c) required both an explanation of what the statements mean, and then a discussion of whether the candidate agrees or disagrees.

An explanation could include some history of how the focus at the Bauhaus was on architecture, and formalising this, moving to design and functional specification. Other aspects include the craft training, including sculpture, metalwork, cabinet-making, etc. A discussion of constructivism would be relevant, as well as the interest in how colour worked and the attempts to create art by mechanical means. Some Bauhaus members were very interested in the design of household artefacts, which also connects with this. Although examiners did not expect all of the above to be present to obtain a good mark, it was essential to include an explicit statement of whether the candidate agrees with the statement, and to justify this. Note that either agreement or disagreeing is acceptable, as long as appropriate justification is given.

Question 3

Colour

About one third of candidates chose this question, with overall performance being reasonable. A couple of marks were below a pass, but there were also some very good responses.

Most candidates were, for part (a), able to correctly name two Bauhaus artists who developed theories of colour, though Klee and Itten were the most popular. It was essential to include what the theories were, to obtain full marks. As an example, Itten was one of the first to formalise any notion of colour scheme, and also developed theory about colour contrasts, while Klee developed a six-part rainbow and work on colour mixing.

Part (b) was also answered well in general. It was essential that the answer was related directly to *Processing*: it implements RGB, and though HSB is available, all colour represented in this way is converted to RGB. *Processing* implements a red, blue, green and alpha channel for each pixel.

For part (c), some candidates did not clearly explain the difference between a colour scheme and colour gradient. A colour scheme is a combination of different colours in a principled manner (for example, a colour scheme may consist of reds and greens; or primary colours; or secondary colours; or even simpler combinations as long as the basis is a principled one) while a gradient shows a gradual movement between two or more colours (for example, a gradient from pale to dark pink). A couple of candidates included the idea that a monochrome image can be a demonstration of the use of a colour gradient.

Part (d) is a question that relates directly to the subject guide. Four different ways include hash notation, hex with explicit Alpha channel, using the `color()` construction and bit-wise operations. Marks were given for knowing these different ways to define colour, and also for correctly showing the coding for red in each.

Question 4

Motion and interaction

This was answered by almost all candidates, with generally good answers.

For part (a), most candidates gave good answers with correct diagrams, but a problem was lack of completeness, where not all of the values asked for were given. A complete answer would include noting the blue background, and red horizontal bar with top-left corner at (0,275) and bottom-right corner at (599,325). The centre of the hole in the bar is at (300,300); the ball is a white circle, with the centre of the circle initially at y=0 and x chosen at random from range (0,599). There is white text for the score, initially showing the number 0 at coordinate (300,40).

Part (b) required understanding that `keyPressed()` is a callback method that *Processing* calls whenever a key is pressed, while the `keyPressed` inbuilt variable is a Boolean variable that indicates whether any key is currently being pressed. The variable would normally be used within the `draw()` method—in which case it would be called once per frame. In both cases, you would need to use the inbuilt variable `key` (and maybe also `keyCode`) to determine whether a specific key has been pressed.

Part (c) was answered less well, and many candidates did not take into account the size of the ball, instead working out a solution for a ball of no size at all. A simple answer would be to insert the following code within the draw method, in an appropriate place:

```
if (ballY > 260 && ballY < 340 &&
(ballX < holeX-10 || ballX > holeX+10))
{
resetPositions();
}
```

Reasonable answers were given for part (d), which is a fairly straightforward question. A simple answer would be to insert the following code after line 36 (although it is possible to place it at other appropriate positions within the draw method):

```
if (ballY >= 600)
{
score++; resetPositions();
}
```

Part (e) was, like part (a), not always answered in a complete manner, and many solutions offered would not have worked properly. It was important to both create a global `PFont` object (e.g. "`PFont f;`" before the `setup()` method), and initialise it in `setup()` using the `loadFont` method, e.g. "`f = loadFont("MyFont-32.vlw");`". After this, also in the `setup()` method, there is a need to specify that this font should be used, using the textFont method, e.g. "`textFont(f);`". Many candidates did not include this part, while others did not specify where the code should be placed.

Question 5

Generative systems

All candidates answered this question, and it was the one that was best answered over- all, with an average of a high upper second and many marks close to the full 25 available.

Almost all candidates correctly said, for part (a), that plants or trees are suitable for modelling using L-systems because bracketed L-systems can produce branching structures. Some also described how they can produce fractals and how repeated shapes can be displayed at different scales (with various techniques being used for scaling).

Part (b) was also answered correctly by almost all candidates, and the first three iterations were shown.

Likewise, part (c)(i) was usually correctly answered, giving the transformations as:

```
line(0,0,25,0);translate(25,0);


rotate(PI/3);
rotate(-PI/3);
pushMatrix();
popMatrix();
```

and the description of the second line as:

```
Move forward length 25 without drawing a line
```

Part (c) (ii) was not always as well answered. Examiners were expecting annotated lengths and angles. The pattern looks something like the below, with the left branch angle at 60 deg, the right branch at 120 deg from the upward vertical, and all lines of length 25:

```
\


\
|\
```

Part (d) expected some reasonable discussion through the various sections. For part (d) (i), examiners expected answers describing user/aesthetic selection; some candidates mentioned fitness functions, which was also acceptable. Aesthetic selection involves a human selecting which individuals should be used as parents for the next generation, according to their own aesthetic criteria.

Variety, for part (d)(ii), is introduced through mutation, by altering the values of the genes that have a low probability. Variety is required as the raw material for selection and evolution; not all candidates included an answer for why it is important.

Part (d)(iii) was sometimes answered well, though this was the weakest part of the question in general. Examiners expected a description of a method with enough detail to begin to code it, and demonstration of understanding of the possible issues. The method should make a small number of changes (usually one, but sometimes more). Mutations could include swapping a symbol in the given rule string for a different symbol, deleting a symbol from any position, or inserting a new symbol (at the beginning, end, or any position within the existing string). The main problems to look out for are preventing the method from returning an empty string, and ensuring that every [symbol is followed by a matching] symbol.

Question 6

3D drawing, animation and object-oriented programming

Question 6 was a less popular question, chosen by only around half of the candidates, but reasonably answered by those that did choose it.

Part (a) required the simple response of the values:

```
ax = 0.0; ay = 0.1; az = 0.0;
```

Almost all candidates correctly stated for part (b) that the code should be:

```
px += vx; py += vy; pz += vz;
```

Again, part (c) was usually answered correctly. The first three co-ordinates are the x, y and z position of the camera; the second three are x, y and z co-ordinates of the centre of screen (focus point), and the final three specify which axis is facing up.

Part (d) was not always answered as completely as required. A good answer would include all of the following. The sketch draws a green ground plane (a box) of 500 units square (and height 1 unit), with a white background. The camera is located at position (0,-500,500) and pointing at the origin (0,0,0). A single particle, represented by a red sphere of diameter 5, starts at the origin and moves upwards in a random direction (i.e. y velocity between -5 and -10, and x and z velocities between -5 and +5). Its upwards velocity is gradually diminished by gravity and it eventually moves downwards with increasing velocity. One candidate also noted that there is no collision detection with the ground.

For part (e), the simplest answer would be to just add two more `Particle` global variables at the start of the code, initialise them in the setup() method the same as p1, and call their draw() and update() methods in the draw() method, the same as p1. More complete answers would suggest using an array or an `ArrayList` or similar data structure.

A naive answer for part (f) might say that it is only necessary to check if `py` > 0.0, and if so, reverse the sign of `vy`:

```
 if (py > 0.0)
{
   vy = -vy;
}
```

However it is also necessary to ensure that `py` returns to a negative position immediately after the collision and that `vy` is only inverted if it is positive to prevent jitter. So a full answer would be something like:

```
if (py > 0.0)
{
   py = -1.0;
   if (vy > 0.0)
   {
      vy = -vy;
   }
}
```

For completeness, examiners expected mention of the coefficient of restitution, and reducing the magnitude of the velocity after the collision.