UNIVERSITY OF LONDON **CO1112 ZB**

BSc, CertHE and Diploma Examination

CREATIVE COMPUTING AND COMBINED DEGREE SCHEME

Creative Computing 1: Image, Sound and Motion

Thursday 17 May 2018: 14.30 – 17.30

Time allowed: 3 hours

There are **SIX** questions on this paper. Candidates should answer **FOUR** questions. All questions carry equal marks, and full marks can be obtained for complete answers to a total of **FOUR** questions. The marks for each part of a question are indicated at the end of the part in [.] brackets.

Only your first **FOUR** answers, in the order that they appear in your answer book, will be marked.

There are 100 marks available on this paper.

A handheld calculator may be used when answering questions on this paper but it must not be pre-programmed or able to display graphics, text or algebraic equations. The make and type of machine must be stated clearly on the front cover of the answer book.

Question 1 General

(a) For the following questions, give an answer of **T** (if the statement is true), or **F** (if the statement is false). You need only write down the question number, and your answer of **T** or **F**. Each correct answer is worth 1 mark. NOTE: there is no negative marking in this question; an incorrect answer will simply receive a mark of zero; other marks already obtained will not be affected.

i. Pythagoras investigated how to produce different musical notes by hitting bells of different sizes.

ii. Brunelleschi worked out the principles of perspective, through the invention of the perspective machine.

iii. Jackson Pollock was involved in the work at the Bauhaus.

iv. The Sierpinski triangle is an example of a fractal.

v. The loudness of a sound is related to the amplitude via the transformation of a logarithm.

vi. Pulse Code Modulation is a type of compression process.

vii. The highest frequency component that we can measure for a sampling rate of SR is $SR/2$.

viii. 44100 Hz is a standard sample rate for CD quality audio.

ix. `popMatrix()` is used to save the current coordinate system transformation in *Processing*.

x. The Blind Watchmaker algorithm is an example of a user-guided genetic algorithm.

xi. An L-system is a string rewriting system, where rules are applied in parallel in any one iteration.

[11]

(b) Name **ONE** woman who was part of the Bauhaus. [1]

(c) What is the output of the following calculation?

$$\begin{pmatrix} 1 & 8 & 3 \\ 0 & 2 & 5 \\ 6 & 5 & 7 \end{pmatrix} \cdot \begin{pmatrix} 8 & 3 \\ 4 & 1 \\ 2 & 2 \end{pmatrix}$$

[2]

(d) Name **TWO** bitwise logical operators provided by *Processing*. [1]

(e) Name **TWO** different audio compression formats. [1]

(f) Define Rule 60 of a 3-bit 1-D cellular automaton. [3]

(g) Starting with a single bit set to 1, in the centre, show the output of the above automaton after four generations. [3]

(h) A biomorph, in the context of Dawkins' visualistion of evolutionary processes, has **THREE** important processes or components associated with it. What are these? [3]

Question 2 History and Creative Thinking

(a) The Bauhaus theory of design includes the identification of basic forms from which all design can flow. What are the basic forms from Bauhaus design theory, and how does the *Processing* language act as a workbench to implement the idea that all design can flow from these forms? [6]

(b) What is a visualisation, in the context of design? Give **TWO** examples of visualisations that are effective, and **TWO** examples of visualisations that are not, explaining why in each case. [10]

(c) The choreographer Merce Cunningham often included a mathematical or algorithmic component in his work. Explain what this means, and discuss in general and through the use of examples how an algorithmic aspect could be used in choreography.

[9]

Question 3 Colour

(a) Name **TWO** Bauhaus members or other artists who developed theories of colour, and briefly describe their theories. [7]

(b) Discuss **THREE** differences between the RGB and CMYK colour schemes. [6]

(c) Describe **FOUR** different ways in which it is possible to define colour in *Processing*. For each one, show how the colour green would be coded. [8]

(d) Describe what a secondary colour is. How would you construct the three secondary colours using *Processing*'s bitwise operators? [4]

Question 4 Motion and Interaction

Study the *Processing* sketch below, then answer the questions that follow.

```
int holeX;
int ballX,ballY;
int ballInc=2;
int score=0;

void setup()
{
  size(500,500);
  frameRate(30);
  noStroke();
  textSize(32);
  resetPositions();
}

void draw()
{
  background(0,255,0);

  fill(255,255,255);
  text(score,250,40);

  // draw bar
  fill(0,0,255);
  rectMode(CORNER);
  rect(0,230,500,40);

  // draw hole
  fill(0,255,0);
  rectMode(CENTER);
  rect(holeX,250,40,40);

  // draw ball
  fill(255,0,0);
  ellipse(ballX,ballY,20,20);

  ballY += ballInc;

  if (keyPressed)
  {
    if (key == ’X’ || key == ’x’)
    {
      holeX += 5;
    }
    else if (key == ’Z’ || key == ’z’)
```

```
45      {
46        holeX -= 5;
47      }
48    }
49  }
50
51  void resetPositions()
52  {
53    holeX=250;
54    ballX=(int)random(500);
55    ballY=0;
56  }
```

(a) When run, the code draws a horizontal bar with a hole in it, and a ball that drops from the top of the screen. Draw a sketch of the output produced by this code in the first frame after it is run. State the colour of the background and any objects drawn, give the top-left and bottom-right coordinates of the horizontal bar, the coordinates of the centre of the hole in the bar, the position of any text drawn, and indicate the range of possible starting positions of the ball. [7]

(b) The code handles keyboard interaction in the `draw()` method (lines 38–48) by testing the inbuilt boolean variable `keyPressed` to see if a key is currently being pressed. *Processing* also provides an alternative method for keyboard interaction, by implementing the `keyPressed()` method. Describe the differences and similarities in how these two approaches are used. [5]

(c) The code shown forms the basis of a simple game. The idea is that the user moves the hole in the central bar left or right (using the keyboard) to allow the ball to drop through the hole without hitting the bar. Add code to the `draw()` method to implement collision detection between the ball and the bar. If a collision is detected, your code should call the `resetPositions()` method. You do not need to write out the whole program in your answer; just show any new code you would add, and indicate the line number(s) where you would insert it. [5]

(d) Add code to the program such that when the ball drops off the bottom of the screen, the `score` variable is incremented and the `resetPositions()` method is called. You do not need to write out the whole program in your answer; just show any new code you would add, and indicate the line number(s) where you would insert it. [4]

(e) The given code prints the score to the screen using the system's default font. Describe how you would modify the code to use a different font. Assume that the data folder of your sketch already contains a suitable font file named `NewFont-32.vlw`. State the code you would add to load and use this font, and say where you would insert these lines (by giving the line numbers in the given code after which you would insert each line). [4]

Question 5 Generative systems

(a) Name **ONE** kind of natural object that L-systems are suitable for modelling. What features of L-systems make them suitable for such modelling? [3]

(b) Consider an L-system with an alphabet $\{a, b, c\}$, an initiator a and the following three generators:

$$a \to bb$$
$$b \to ac$$
$$c \to ba$$

Write down the first **THREE** iterations of substitutions on the initiator in this L-system. [3]

(c)

Symbol	Command	Transformation
F	Draw a line of length 20 and move to its end	?
f	?	`translate(20,0);`
+	Rotate by `PI/4` radians (45 degrees)	?
-	Rotate by `-PI/4` radians (-45 degrees)	?
[	Open branch (save turtle state)	?
]	Close branch (restore turtle state)	?

The partially-completed table above shows a possible graphical interpretation for an L-system with alphabet $\{F, f, +, -, [,]\}$.

i. Complete the **SIX** missing items in the table (denoted by question marks). Where a transformation is missing, give *Processing* code that will implement the given command. Where a command is missing, provide a description of the given transformation in the context of the L-system interpretation. [6]

ii. Assuming a window size of 200 by 200, a starting position at (100,200), and a starting rotation of `-HALF_PI`, draw the output of applying the graphical interpretation shown in the table to the string `F[-FfF][++F]`. Annotate your drawing with lengths and angles for each part. [3]

(d) Genetic algorithms are models of biological evolution. They have been used in creative applications as a means of exploring large parameter spaces.

i. Genetic algorithms operate on a population of candidate solutions to a problem. Briefly explain what role the population plays in the operation of the genetic algorithm. [3]

ii. When creating a new generation, multiple children may be produced from a single parent. How, and why, is variety introduced among the children in this case? [3]

iii. Suppose that you are writing a *Processing* sketch to evolve L-system patterns using a genetic algorithm. Describe how you might implement a `mutateRule` method for this program, that takes a symbol string (which could be of any non-zero length) representing the right-hand side of a generator rule, and returns a mutated version of that string. Assume that the method is of the form: "`String mutateRule(String rule)`". You do not need to write code (although you may if you wish), but describe what the method should do, highlight any problem cases that might arise, and suggest how the method should deal with such cases. [4]

Question 6 3D drawing, animation and object-oriented programming

Study the *Processing* sketch below, then answer the questions that follow.

```
Particle p1;

void setup()
{
  size(600,600,P3D);
  p1 = new Particle();
  camera(0,-1000,1000,0,0,0,0,1,0);
  rectMode(CENTER);
  noStroke();
}

void draw()
{
  background(255);
  fill(0,255,0);
  box(600,1,600);
  p1.draw();
  p1.update();
}

class Particle       // definition of Particle class
{
  float px, py, pz; // particle position
  float vx, vy, vz; // particle velocity
  float ax, ay, az; // particle acceleration

  Particle()        // constructor
  {
    px = py = pz = 0.0;
    vx = random(-10.0, 10.0);
    vy = random(-10.0, -20.0);
    vz = random(-10.0, 10.0);
    ax = ?;
    ay = ?;
    az = ?;
  }

  void draw()
  {
    pushMatrix();
    fill(0,0,255);
    translate(px,py,pz);
    sphere(10);
    popMatrix();
```

```
45    }
46
47    void update()
48    {
49      ?
50      px += vx; py += vy; pz += vz;
51    }
52  }
```

(a) The sketch makes use of the `Particle` class, which is defined in lines 21–52. The class stores a particle's current position, velocity and acceleration. In line 6, a new instance of this class is created. When a new instance of a class is created in this way, *Processing* automatically calls the class constructor to initialise the new instance. The constructor is defined in lines 27–36.

We wish to add a gravitational force to the particle as it moves. Lines 33–35 of the constructor are incomplete. State what code should replace the question marks in these three lines so that the particle is initialised with an acceleration of 0.2 units in the downwards direction (*i.e.* a downwards acceleration of 0.2 spatial units per frame per frame). [3]

(b) The `update()` method of the `Particle` class (lines 47–51) is incomplete. What code should replace the question mark on line 49 so that the particle's velocity is updated appropriately according to its current state? [3]

(c) Line 7 of the sketch calls the `camera()` function. Explain the meaning of the nine arguments passed into the `camera()` function on this line. [3]

(d) Using a drawing with appropriate labels, describe the output of the sketch when it is run. Show any objects that are drawn and indicate any movement that happens. Specify positions, dimensions and colours where appropriate. [6]

(e) Extend the sketch so that **THREE** independent particles are animated rather than just one. You do not need to write out the whole sketch in your answer, just indicate any lines you would modify, delete or add to the original sketch. [5]

(f) By modifying the `update()` method of the `Particle` class (lines 47–51), extend the sketch to implement collision detection and response when a particle hits the ground. If a collision is detected you should reverse the vertical component of the particle's velocity. You do not need to write out the whole sketch in your answer, just indicate any lines you would modify, delete or add to the original sketch. [5]

END OF PAPER