General instructions:

1. This exam is open-book. You may use any reference material you require, and you may use a laptop computer if you like.
2. There are 3 questions, each worth 10 points, for a total of 30 points possible. Attempt all the questions. All questions will be graded and contribute to your score.
3. Record your answers on the USB memory stick provided. You may also add any other information about your answers in the examination booklet provided. Make sure that your name is printed legibly on the front of the booklet and that your name and ID are on the inside of the booklet before you turn it in.
4. You MUST return the question booklet.
5. There are 5 pages in the question book. Make sure you have all the pages before you begin.
1. [10] Suppose you have the following Processing program.

```java
noStroke();
fill(0);

scale(5);
rotate(radians(30));
translate(width/2, height/2);

triangle(3, 4, -3, 4, 0, -4);
```

When you run the program, you see this image below on the left, but what you really want to see is the image on the right. Fix the code so that you get the image on the right.
2. [10] Write the function `drawLeaf67()` to complete the program below and draw the picture shown on the right. The diagram on the following page may help.

```java
void setup() {
    drawLeaf67( width/2, height/2, 50.0 );
    noLoop();
}

void drawLeaf67( float x, float y, float s ) {
    // draw centenial logo centered at x,y in s-by-s box
}

void equilateralTriangle( boolean up ) {
    // draw an equilateral triangle pointing up
    // centre at origin, random colour
    noStroke();
    fill( int(random(256)), int(random(256)), int(random(256)) );
    pushMatrix();
    if ( ! up ) {
        rotate( PI );
    }
    triangle( 0, -0.433, 0.5, 0.433, -0.5, 0.433 );
    popMatrix();
}

void stem() {
    // draw a stem centred at origin, random colour
    noStroke();
    fill( int(random(256)), int(random(256)), int(random(256)) );
    rect( -0.1, -0.433, 0.2, 0.866 );
}

Your function should use the functions `equilateralTriangle()` and `stem()`. The `setup()` function below produces the image on the right, demonstrating how to use these functions. Use `translate()`, `rotate()` and `scale()` as needed to produce a facsimile of the image above.

```java
void setup() {
    scale(25);
    translate(1, 2);
    equilateralTriangle(true);
    translate( 1, 0 );
    equilateralTriangle(false);
    translate( 1, 0 );
    stem();
    noLoop();
}
```
**Hints:** Build incrementally – try one triangle. Adjust the scale. Then add more triangles, and add the stem.
3. [10] Using the following code skeleton, write a program that displays a dot on the screen whenever a mouse button is clicked, but at the position where the mouse was 10 clicks prior. For the first 10 clicks, you can draw the dot in an arbitrary position.

```cpp
int nSave; // number of old mouse clicks to remember

// arrays of previous mouse positions
int[] pastMouseX;
int[] pastMouseY;

// keep track of where you are in the history of clicks
int pastMouseIndex = 0;

void setup() {
    noStroke();
    fill(0);
    smooth();
    noLoop();
}

// mousePressed() event function
void mousePressed() {
}

void draw() {
}
```

Remember, you can control when processing draws with the `redraw()` function. Also, if you use `redraw()`, you will need a `draw()` function, even if does not do anything.

You can look at the invigilators computer if you want to see an example of what your final product should look like.

**Hints:** You will need to give `nSave` an initial value, and use that to initialize the arrays `pastMouseX` and `pastMouseY` to contain the correct number of elements. You then need to get the variable `pastMouseIndex` to loop through the elements of the array stepping once each time the mouse is pressed. The diagram below might help. When the mouse is pressed, draw an ellipse as indicated by `pastMouseIndex`. Then put new values into the arrays, and step `pastMouseIndex`.

---

JEB

---

5/5