## (I) Warm-up Exercises

Q1. What will be done by the following programs?

```
var a = 10;
var b = 20;

a = b;
b = a;
```

```
var a = 10;
var b = 20;
var temp;

temp = a;
a = b;
b = temp;
```

Q2. Un-intentional Empty Statements I

Please **download** the given program and open it in Atom. The program is based on a previous program that bounces a ball left-and-right inside the Canvas.

```
var x = 50;
var step = 3;

function setup() {
  createCanvas(300, 300);
  frameRate(6); // <== a slow animation for better viewing
}

function draw() {
  background(100);
  stroke(0);
  fill(250, 255, 0);
  ellipse(x, 150, 50, 50);

  x = x + step;

  if (x < 0 || x > width); //a semicolon by mistake
    step = step * -1;
}
```

(a) Run the program. What do you see?
Note: *Adding semicolon at the end of an if-clause is wrong!*

(b) Study the following explanation:

Explanation:
If we put a ; after the if-condition, it is treated as one valid statement (known as *empty-statement* that performs nothing). Such an empty-statement is the only action body of the if-statement.

Therefore,

```
if (x < 0 || x > width);
  step = step * -1;
```

runs in this way ======>

```
if (x < 0 || x > width)
      ; //do nothing
step = step * -1;
```

(c) In every frame, the value of *step* is toggled between +3 and -3. What are the location where the ball is drawn?

Frame 1: (50 , 150), Frame 2: (_____,150), Frame 3: (_____,150), Frame 4: (_____,150) and so on.

Q3. Un-intentional Empty Statements II

What will be shown if the following are executed? Why?

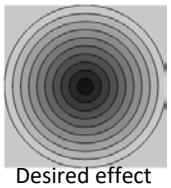
```
var i = 0;
while (i < 10) {
 text(i, i * 20, 20);
 i++;
}
```

- 0 1 2 3 4 5 6 7 8 9

```
var i = 0;
while (i < 10); {
 text(i, i * 20, 20);
 i++;
}
```

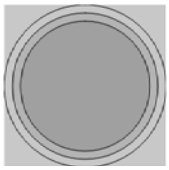
- 0 1 2 3 4 5 6 7 8 9 10
- *Only 10 is shown*
- Nothing is shown (the loop can't stop)

Q4 Rewrite the given program with while-loop and for-loop to give the desired effect



Desired effect

```
function setup() {
  createCanvas(200, 200);
  background(200);
  var w;
  w = width;
  fill(w);
  ellipse(width/2, height/2, w, w);
  w = w - 20;
  fill(w);
  ellipse(width/2, height/2, w, w);
  w = w - 20;
  fill(w);
  ellipse(width/2, height/2, w, w);
}
```



Using while-loop:

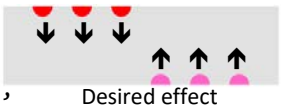
```
function setup() {
  createCanvas(200, 200);
  background(200);









}
```

Using for-loop:

```
function setup() {
  createCanvas(200, 200);
  background(200);









}
```

Q5 Rewrite the following program using <u>for-loop</u> to give the desired effect

```
var yRed, yPink;

function setup() {
  createCanvas(350, 100);
  yRed = 0;
  yPink = height;
}

function draw() {
  background(225);
  noStroke();

  fill(255, 0, 0);  //red ball
  ellipse(50, yRed, 25, 25);
  yRed = yRed + 1;
  if (yRed - 25 > height) {
    yRed = 0;
  }

  fill(255, 100, 200);  //pink ball
  ellipse(200, yPink, 25, 25);
  yPink = yPink - 1;
  if (yPink + 25 < 0) {
    yPink = height;
  }
}
```



Desired effect

```
var yRed, yPink;

function setup() {
  createCanvas(350, 100);
  yRed = 0;
  yPink = height;
}

function draw() {
  background(225);
  noStroke();









}
```

Q6 Rewrite the following program using for-loop to give the desired effect

```
function setup() {

  createCanvas(300, 100);
  background(255,255,200);

  noStroke();

  var w = 80;

  for (var x = 50; x < width; x += 100) {
    fill(w + w);
    ellipse(x, height / 2, w, w);
  }
}
```

What is the output?

```
function setup() {

  createCanvas(300, 100);
  background(255,255,200);

  noStroke();









}
```

Desired effect

Q7. Suppose now we need to write a longer program (say totally 70 lines or more), choose the better one below:

1. *To complete the program in shortest time, I can write all steps in a long draw function first. Before I submit it to the boss, I will split it part-by-part into several functions.*

2. *To complete a program in shortest time, the fastest way is to apply the top-down approach: The draw function should be short and contain least details. Through designing the draw function, I also plan for a list of required functions and I will create the functions later.*

## (II) Working out Class Exercises

Job 1:
Change the getSpeed() function such that
it returns the speed of 2 when the ball is at the left half of the canvas, and
It returns the speed of 10 when the ball is at the right half of the canvas.

```
var circleX = 25;
var circleY = 100;

function setup() {
  createCanvas(500, 200);
}

function draw() {
  background(0,10);
  stroke(0);
  fill(175);

  move(getSpeed());
  ellipse(circleX, circleY, 50, 50);
}

function getSpeed() {
  return frameCount / 100;
}

function move(speed) {
  if (circleX > width) {
    circleX = 0;
  }
  circleX += speed;
}
```

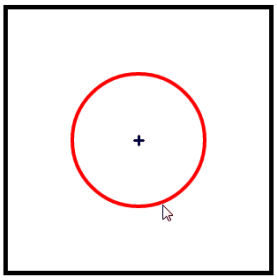
Job 2:

Given: [Example 5] The canvas turns red gradually when the cursor moves towards the center of canvas (white +).

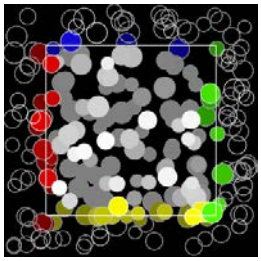
```
function setup() {
  createCanvas(200, 200);
}

function draw() {
  background(255 - distance(width / 2, height / 2, mouseX, mouseY), 0, 0);

  strokeWeight(2);
  stroke(255);
  line(width / 2 - 5, height / 2, width / 2 + 5, height / 2);
  line(width / 2, height / 2 - 5, width / 2, height / 2 + 5);
}

function distance(x1, y1, x2, y2) {
  var dx = x1 - x2;
  var dy = y1 - y2;
  var d = sqrt(dx * dx + dy * dy);
  return d;
}
```

Your task:
Create this animation: The ellipse expands or shrinks in proportion to the distance of mouse cursor from the center of canvas (blue +). The ellipse touches the border of canvas only when the cursor is on the wall or outside of the wall. You do not need to draw the black outer border.

```
function setup() {
  createCanvas(200, 200);
}

function draw() {









}

function distance(x1, y1, x2, y2) {
  var dx = x1 - x2;
  var dy = y1 - y2;
  var d = sqrt(dx * dx + dy * dy);
  return d;
}
```



Job 3:



```
function setup() {
  createCanvas(300, 300);
  background(0);
}

function draw() {
  background(0, 1);

  noFill();
  stroke(255);
  rect(50, 50, 200, 200);

  var d = random(15, 25);
  var x = random(d / 2, width - d / 2);
  var y = random(d / 2, height - d / 2);
  stroke(255);
  noFill();
  var ez = enteringZone(x, y, d, 50, 50, 250, 250);
  if (ez == 0) {
    fill(255);
  } // else if (ez == 1) {









  ellipse(x, y, d, d);
}

//check whether a ball entering a rectangular zone
//return -1 means: not even touching
//return 0 means: entered totally
//return 1/2/3/4 means: entering at left/top/right/bottom
function enteringZone(x, y, ballSize, left, top, right, bottom) {
```

```
}
```

Job 4:

```
var x;
var y;
var xStep;
var yStep;
var ballSize;

function setup() {
  createCanvas(250, 400);
  background(0);
  x = width/2; //width and height are available after canvas is created
  y = 10;
  yStep = 2;
  ballSize = random(30,60); //random() is available after setup() starts
  xStep = random(5, 10);
}

function draw() {
  background(0,5);

  fill(255);
  ellipse(x, y, ballSize, ballSize);
  x += xStep;
  y += yStep;

  var bm = bounceAtMargin();

  if (bm == 1) {
    x = ballSize / 2;
    xStep = random(5, 10);
  } //else if (bm == 2) {









}

//check whether the ball is bouncing at the margins at left, top, right,
bottom
//return 0 means: not bouncing
//return 1/2/3/4 means: bouncing at left/top/right/bottom
function bounceAtMargin() {









}
```