CS1103 Workshop 6

MORE ON FUNCTIONS

Function with return value

FUNCTIONS

With return value

A function not only accepts input(s), but may also return value or result.

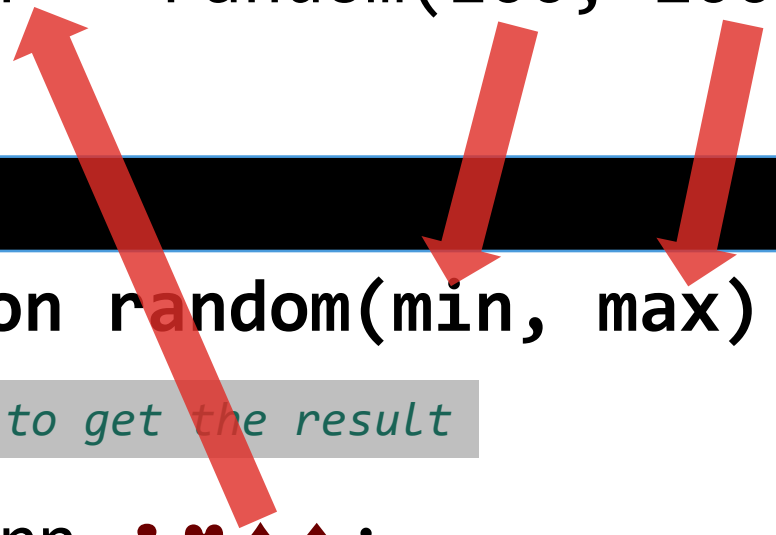
For example, the ***random*** function returns a random value when the function is invoked.

FUNCTIONS with RETURN VALUE

Example 1: The random() by p5.js

```
function setup() {

   var r = random(100, 200);

}
```

```
function random(min, max) {

   steps to get the result

   return ♣♥♠♦;
}
```



p5.js already provides the *random* function.
It *returns a random value to the caller*.

FUNCTIONS with RETURN VALUE

Example 2: The sqrt() by p5.js

```
function setup() {

   var result = sqrt(1103);

}
```

Calculate the square root of a number

33.21

1103

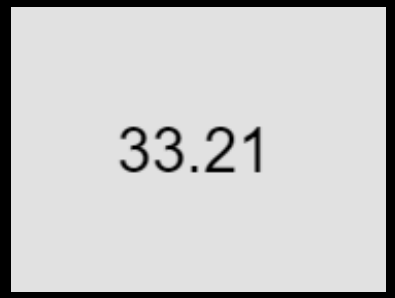
```
function sqrt(k) {

   steps to get the result

   return ♣♥♠♦♣♥♠;
}
```

p5.js already provides the *sqrt* function.
It *returns the square root of a value to the caller*.

FUNCTIONS with RETURN VALUE

Example 2: The sqrt() by p5.js

The complete program:




```
function setup() {
  createCanvas(200, 150);
  background(225);
  textAlign(CENTER, CENTER);
  textSize(30);
  fill(0);

  var result = sqrt(1103);

  text(result.toFixed(2), width / 2, height / 2);

}
```

[Note]

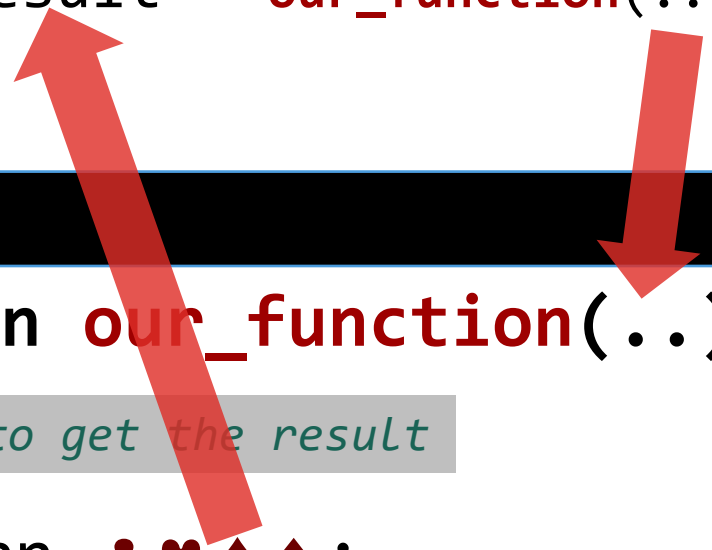
`result.toFixed(2)` gives the value of `result` in 2 decimal points

FUNCTIONS with RETURN VALUE

We can also write our own functions that:
returns a value to the caller.

```
function setup() {

   var result = our_function(..);

}
```

```
function our_function(..) {

   steps to get the result

   return ♣♥♠♦;
}
```



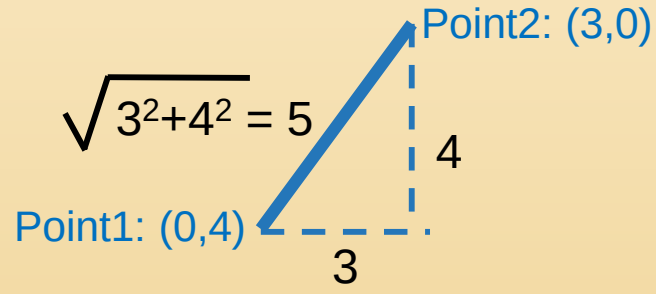
FUNCTIONS with RETURN VALUE

Example 3: The distance() function written by ourselves

To get the distance between (x1, y1) and (x2, y2), we can write the distance function that *returns* the calculation result.

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

*This is called **a return statement**.*
It *returns* the value d to the caller [and stops running the function].



$$\sqrt{(x1-x2)^2 + (y1-y2)^2}$$

Horizontal distance

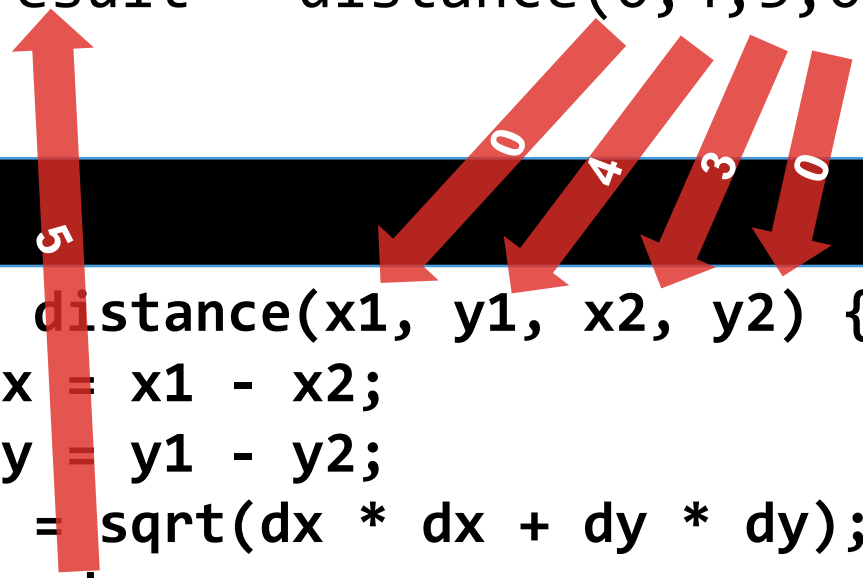
Vertical distance

FUNCTIONS with RETURN VALUE

Example 3: The distance() function written by ourselves

```
function setup() {

   var result = distance(0,4,3,0);

}
```

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



Here we create our own function, *distance*:

It accepts 4 input values into parameters `x1, y1, x2, y2;`

Then carries out calculation step by step.

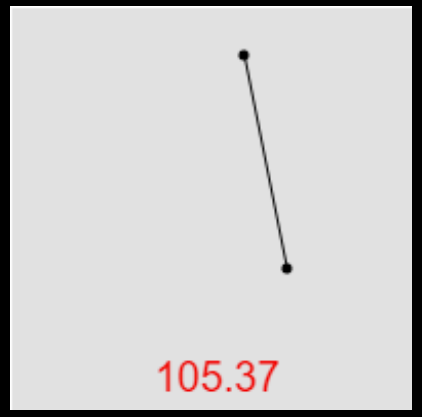
Temporary variables `dx, dy, d` are used to store intermediate values.

E.g., dx will get -3, dy will get 4, d will get 5.

Finally return the resultant value to the caller.

The *`setup`* function, as the caller, uses the `result` variable to hold the returned value.

Example 3
A complete program that uses the distance() function:



Note:

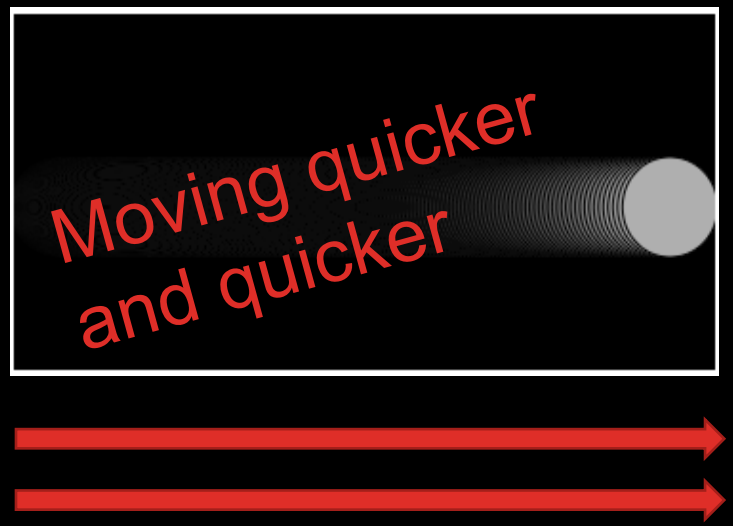
1. Recall: `result.toFixed(2)` gives the value of `result` in 2 decimal points
2. Actually, p5.js already provides the `dist` function, that calculates the distance like our function does.☺

```
function setup() {
   createCanvas(200, 200);  background(225);
   textAlign(CENTER, CENTER);  textSize(20);
   noFill(); stroke(0);

   var xPoint1 = random(20, 150);
   var yPoint1 = random(20, 150);
   var xPoint2 = random(20, 150);
   var yPoint2 = random(20, 150);

   strokeWeight(5);
   point(xPoint1, yPoint1);
   point(xPoint2, yPoint2);
   strokeWeight(1);
   line(xPoint1, yPoint1, xPoint2, yPoint2);

   var result = distance(xPoint1, yPoint1, xPoint2, yPoint2);
   noStroke();
   fill(255, 0, 0);
   text(result.toFixed(2), 100, 180);
}

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

FUNCTIONS with RETURN VALUE

Example 4: The simple getSpeed() function

The function `getSpeed()` returns the value of `frameCount/100`.




```
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;
}
```

FUNCTIONS with RETURN VALUE

Example 4: The simple getSpeed() function

The function `getSpeed()` returns the value of `frameCount/100`.

Question 1: Moving how fast?

Answer:
frameCount 1-99: <1 pixel per frame
frameCount 100-200: 1-2 pixels per frame
frameCount 200-300: 2-3 pixels per frame
…

Question 2: Can we write the same animation without writing and using the getSpeed() function?

Answer:
Yes. move(frameCount/100); gives the same effect.
It is not a must to use the getSpeed() function.
Discussion: Any advantage of writing a getSpeed() function?

```
function draw() {
  background(0,10);  stroke(0);  fill(175);
  move(getSpeed());
  ellipse(circleX, circleY, 50, 50);
}

function getSpeed() {
  return frameCount / 100;
}
```

FUNCTIONS with RETURN VALUE

Example 4: The simple getSpeed() function

The function `getSpeed()` returns the value of `frameCount/100`.

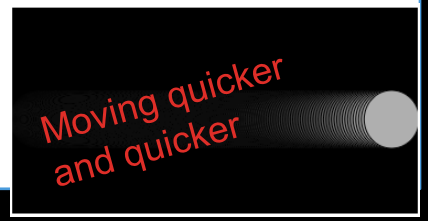
Question 3:
Here when we call getSpeed(), we do not need a variable to hold the result. Why?

Answer:
OKay, if you want, you can write it like:

```
var sp = getSpeed();
move(sp);  //use sp's value as the input to run the move() function.
```

Yet we can also simply write `move(getSpeed());`
that means
(1) first run `getSpeed();`
(2) then use its return value to provide the input speed value for `move(speed)`.

```
function draw() {
  background(0,10);  stroke(0);  fill(175);
  move(getSpeed());
  ellipse(circleX, circleY, 50, 50);
}

function getSpeed() {
  return frameCount / 100;
}
```



FUNCTIONS with RETURN VALUE

Example 4: The simple getSpeed() function

The function `getSpeed()` returns the value of `frameCount/100`.

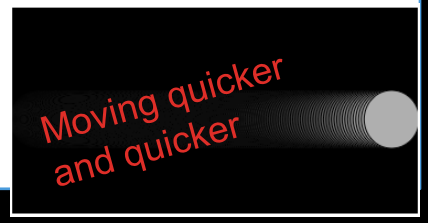
Question 4:
When we write the getSpeed() function, we do not need a variable to hold the result for returning. Why?

Answer:
OKay, if you want, you can write it like:

```
var result = frameCount/100;
return result;  //return the value of result to the caller
```

Yet we can also simply write `return frameCount/100;` that means (1) first evaluate `frameCount/100;` (2) then the return statement returns the evaluated value to the caller.

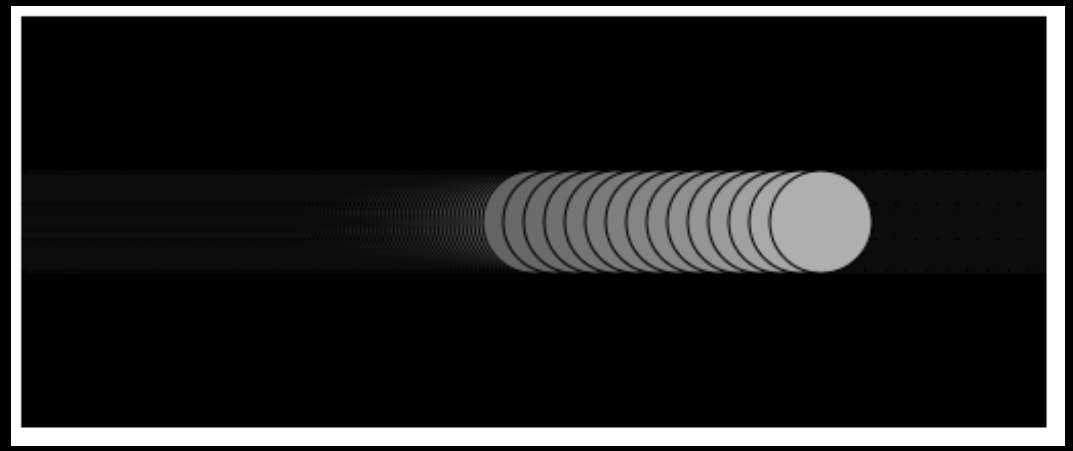
```
function draw() {
  background(0,10);  stroke(0);  fill(175);
  move(getSpeed());
  ellipse(circleX, circleY, 50, 50);
}

function getSpeed() {
  return frameCount / 100;
}
```



FUNCTIONS with RETURN VALUE

Job 1:

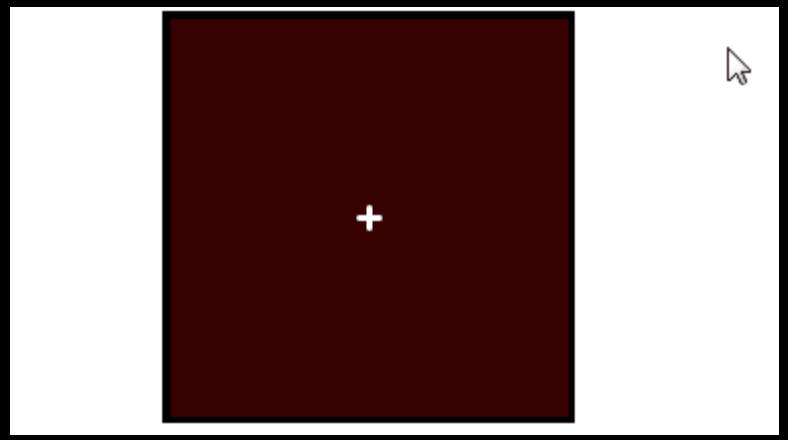
Change the `getSpeed()` function such that
it returns the speed of 2 when the ball is at the left half of the canvas, and
It returns the speed of 10 when the ball is at the right half of the canvas.



FUNCTIONS with RETURN VALUE

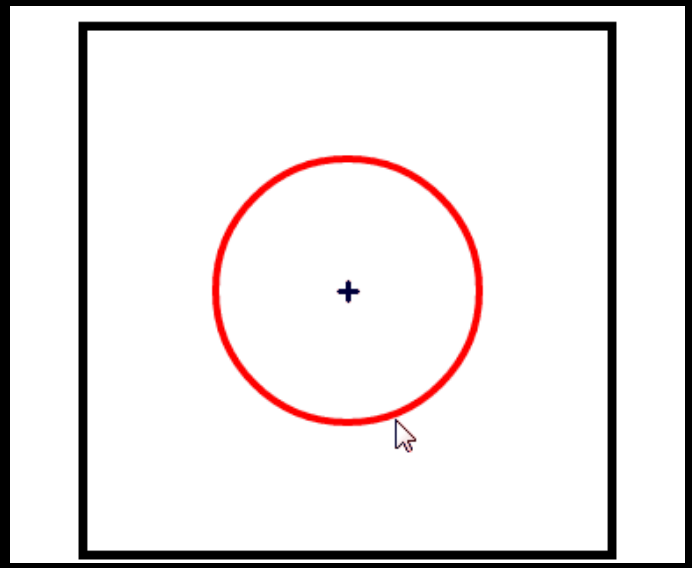
Example 5

The canvas turns red gradually when the cursor moves towards the center of canvas (white +).



Job 2:

Based on Example 5, create this animation:

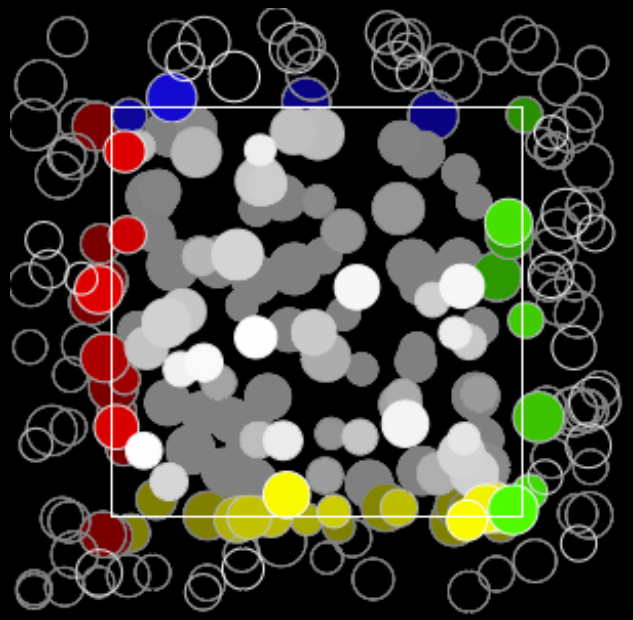


The ellipse generally expands or shrinks while touching the mouse cursor. The size of the ellipse is bounded by the canvas boundaries. You do not need to draw the black outer border.

Job 3 [Required for assignment]

Complete the program so that

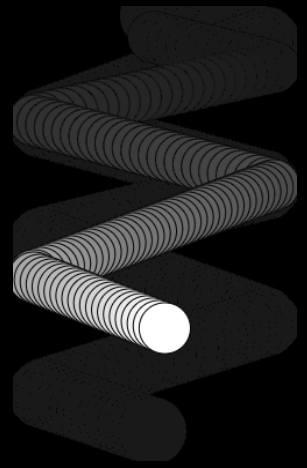
- It draws a rectangle frame,
- It paints random balls
- The balls outside / inside / cutting the frames are colored differently as shown in the picture.



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

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

  var d, x, y;  d = random(15, 25);
  x = random(d/2,300-d/2); y = random(d/2,300-d/2);

  stroke(255); noFill();

  var ez = enteringZone(x, y, d, 50, 50, 250, 250);
  if (ez == 0) {
    fill(255);
  } else if (..)
  .. //check the code in ez and choose the fill color



  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 [Required for assignment]

Complete the program for the animation:



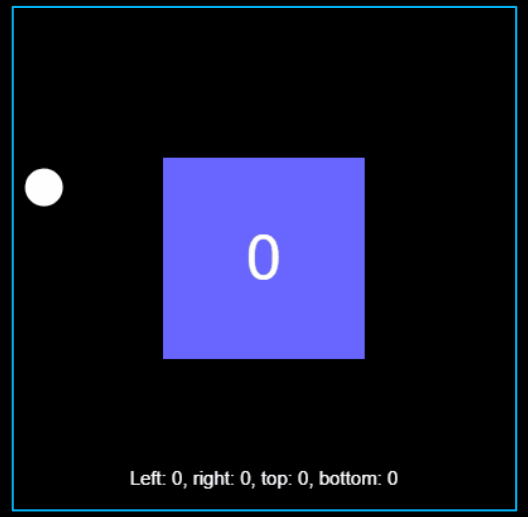
- A ball goes down in a zigzag manner.
- When it falls away down from the bottom, re-start from the top.
- The vertical speed (yStep) is 2
- The horizontal speed (xStep) is random(5,10) when it goes right, random(-10, -5) when it goes left.
- Generate a new speed whenever it bounces at the left/right margin
- Generate a random ball size at the beginning, and when it comes back from the top.

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

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

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; //adjust the ball to exactly touch the margin
    xStep = random(5, 10);
  } else if (...)



}

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

Assignment of this lesson:

Create a program according to the requirement below.
Submit it on Canvas.
Due: Please check the deadline on Canvas.

Create the animation according to the requirements below (5 marks)



Details of the basic requirements [4 marks]

1. Draw an rectangle zone inside the canvas. Bounce a ball inside the Canvas and at the edges of the zone. The ball size is fixed to 30.
2. Count the times the ball bounces at the 4 edges of the zone and show the counts at the bottom of the canvas. Also show the total sum of bounces at the center of the zone.
3. The vertical speed of the ball is random(5,10) or random(-10, -5) for downward or upward motions respectively; similar for the horizontal speed. Generate a new speed whenever it hits the margins or the edges of the zone. For example, if it hits the top edge of the zone, that means it is to bounce upward, therefore change the vertical speed to random(-10,-5).
4. Your solution should involve writing and using functions with return value. Please refer to Job 3-4 for doing so.

Creative component [1 mark]
Implement creative features through *writing and calling functions with parameters and return values.*

Animation: https://courses.cs.cityu.edu.hk/cs1103/public/Workshop06_AsgAnimation/