**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |
| --- |
| **Part 0: Variables** |

1. Choose the line of code that demonstrates the correct way to declare a variable.

|  |  |  |
| --- | --- | --- |
| variable myScore; | **let myScore;** | myScore = let; |

1. Choose the line of code that demonstrates the correct way to initialize a variable.

|  |  |  |
| --- | --- | --- |
| **myScore = 0;** | 0 = myScore; | myScore 0; |

1. **True** or false (circle one): A variable can be declared and initialized in the same line.
2. **True** or false (circle one): A variable can be declared and initialized on separate lines.
3. mouseY is a p5-defined:
   1. boolean variable
   2. **number variable**
   3. string variable
4. key is a p5-defined:
   1. boolean variable
   2. number variable
   3. **string variable**
5. width is a p5-defined:
   1. boolean variable
   2. **number variable**
   3. string variable
6. mouseIsPressed is a p5-defined:
   1. **boolean variable**
   2. number variable
   3. string variable
7. Which of the following will NOTwork for incrementing variable n by 1?
   1. n += 1;
   2. n = n + 1;
   3. **1 += n;**
   4. n++;

|  |
| --- |
| **Part 1: Math with variables** |

Calculate the variable’s value after the following lines of code run.

1. let x = 20; x’s value is \_\_\_\_\_\_\_25\_\_\_\_\_\_\_\_

x += 5;

1. let y = 15; y’s value is \_\_\_\_\_\_\_30\_\_\_\_\_\_\_\_\_

y \*= 2;

1. let z = x + 4 / 3; z’s value is \_\_\_\_\_\_\_\_23.333\_\_\_\_\_\_\_\_\_
2. let k = 20 % 3; k’s value is \_\_\_\_\_\_\_\_2\_\_\_\_\_\_\_\_\_
3. let w = (6.5 + 3) \* (2 \* 5); w’s value is \_\_\_\_\_\_\_\_95\_\_\_\_\_\_\_\_\_
4. let i = round(6.5 + 3); i’s value is \_\_\_\_\_\_\_\_\_10\_\_\_\_\_\_\_\_\_
5. let j = !(false && true) || false; j’s value is \_\_\_\_\_\_\_\_true\_\_\_\_\_\_\_\_\_\_
6. let t = 25 \* 5 / 5; t’s value is \_\_\_\_\_\_25\_\_\_\_\_\_\_\_\_\_\_\_
7. let flag = false;

let penalty = true;

let loss = penalty && flag; loss’s value is \_\_\_\_\_\_\_false\_\_\_\_\_\_\_\_\_

1. let num1 = 10;

let num2 = 4;

let num3 = num1 + num2; num3’s value is \_\_\_\_\_\_\_14\_\_\_\_\_\_\_\_

1. let num1 = 10;

let num2 = 10;

num2 += num1;

let num3 = num1 + num2; num3’s value is \_\_\_\_\_\_\_\_30\_\_\_\_\_\_\_\_\_\_

|  |
| --- |
| **Part 3: P5 Structure** |

Refer to the following complete program for questions below:

let wSize = 300; // Statement 1

function setup(){

createCanvas(wSize, wSize);

background(0);

stroke(255); // Statement 2

}

function draw(){

line(pmouseX, pmouseY, mouseX, mouseY); // Statement 3

}

function keyPressed(){

background(0); // Statement 4

}

For each of the four commented statements, describe what happens when that line of code runs.

1. Statement 1: initialize wSize as a variable and make it equal 300
2. Statement 2: Make the stroke have a weight of 255 for all objects/shape that can have a stroke
3. Statement 3: Line is drawn from the previous position of the mouse in the last frame to the current position of the mouse in the current frame
4. Statement 4: When any key is pressed on the keyboard, the background color is change to black(covering the last draw()). Basically, resetting the canvas so that the line drawn disappears.
5. Which statement/s will be executed only once?

-Statement 1 and 2

1. Which statement/s will be executed the most?

-Most likely Statement 3 depending on how key-press happy the user is.

1. Which of the following are true about the mousePressed() function?
   1. Must be defined by you
   2. Accepts parameters
   3. Returns a value
2. Which of the following are true about the random() function?
   1. Must be defined by you
   2. Accepts parameters
   3. Returns a value
3. Which of the following are true about the keyTyped() function?
   1. Must be defined by you
   2. Accepts parameters
   3. Returns a value
4. Which of the following are true about the createCanvas() function?
   1. Must be defined by you
   2. Accepts parameters
   3. Returns a value
5. Which of the following are true about the arc() function?
   1. Must be defined by you
   2. Accepts parameters
   3. Returns a value
6. Which of the following are true about the text() function?
   1. Must be defined by you
   2. Accepts parameters
   3. Returns a value
7. Which of the following are true about the setup() function?
   1. Must be defined by you
   2. Accepts parameters
   3. Returns a value

|  |
| --- |
| **Part 2: Math in Loops** |

Refer to the following code fragment for questions 1 to 4:

let i = 0;

while (i < 5) {

line (i \* 50, 0, i \* 50, 100);

i = i + 1;

}

1. How many lines will this code draw? \_\_\_\_\_5\_\_\_\_\_\_
2. Draw a circle around the initialization statement.
3. Draw a box around the test condition.
4. Underline the change statement.

For questions 5-7, calculate the variable’s value as the loop stops running:

1. for(let num = 1; num < 5; num++){

rect(num \* 1, num \* 2, 10, 10);

}

**num** holds \_\_\_\_\_\_\_\_5\_\_\_\_\_\_\_\_\_\_ - loops 4x

1. let f = 0.0;

while(f < 2.2){

f += 0.5;

}

**f** holds \_\_\_\_\_\_\_\_\_\_2.5\_\_\_\_\_\_\_\_\_\_\_\_ -loops 5x

1. let mathIsDone = false;  
   let value = 2;

while(!mathIsDone){

value++;

if(value === 7){

mathIsDone = true;

}

}

**mathIsDone** holds \_\_\_\_\_\_\_\_\_true\_\_\_\_\_\_\_\_\_\_\_\_\_

**value** holds \_\_\_\_\_\_\_\_\_\_7\_\_\_\_\_\_\_\_\_\_\_\_

1. Write a *for* loop that will print the following values to the console: 30, 25, 20, 15, 10, 5.

for (let a = 30; a > 0; a -= 5) {

console.log(a);

}

1. If we want to fill the canvas with horizontal lines spaced [spacing variable value] pixels apart vertically, what should fill in the blank?

let spacing = 20;

for (let newY = 0; newY < height; \_\_\_\_\_\_\_\_\_\_\_newY += spacing\_\_\_\_\_\_\_\_\_\_\_) {

line(0, newY, width, newY);

}

Refer to the following code fragment for questions 10 and 11:

let keepGoing = true;

while (keepGoing = true) {

line(150, 150, random(0, 300), random(0, 300));

(11) if(random() < .9) { keepGoing == true;

} else {

keepGoing == false;

}

1. What’s wrong with this code?

-the while loop won’t stop because there is no condition that will make keepGoing = false

-also, keepGoing is defined as a Boolean var been then treated as number var later on

keepGoing == true

keepGoing === true

1. Fix the problem by writing in a correction/s.

|  |
| --- |
| **Part 3: Loop Output** |

Without running this code, determine what the console output of these loops will be. For example, console.log(“hi”); produces the console output: hi

(If you absolutely get stuck after trying everything, you may run the code in p5, but only as a last resort.)

1. let number1 = 20;

for (let n = 0; n < 3; n++) {

number1 \*= 2;  
 console.log(“Number 1 is: ”, number1);

}

**Console Output:**

**Number 1 is: 40**

**Number 1 is: 80**

**Number 1 is: 160**

1. let incrementer = 1;

while (incrementer < 5) {  
 console.log(“I like math.”);

incrementer++;

}

**Console Output:**

I like math.

I like math.

I like math.

I like math.

|  |
| --- |
| **Part 4: Loop Re-writes** |

Study the code below. Determine how the code can be re-written into both a while and a for loop. Write the code for your loop below. Feel free to test your loops in p5.

function draw(){

colorMode(HSB);

fill(100, 100, 100);

rect(10, 10, 50, 50);

fill(80, 100, 100);

rect(60, 60, 50, 50);

fill(60, 100, 100);

rect(110, 110, 50, 50);

fill(40, 100, 100);

rect(160, 160, 50, 50);

fill(20, 100, 100);

rect(210, 210, 50, 50);

}

**Your while loop:**

function draw(){

Let side = 10

Let shade = 100

Let i = 0

colorMode(HSB);

While (i < 5) {

colorMode(HSB);

fill(shade, 100, 100);

rect(side, side, 50, 50);

i++;

shade -= 20;

side += 50;

}

}

**Your for loop:**

function draw(){

colorMode(HSB);

Let side = 10

Let shade = 100

for (let i = 0; i < 5; i++) {

fill(shade, 100, 100);

rect(side, side, 50, 50);

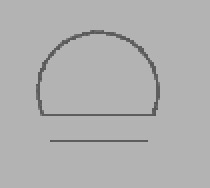
shade -= 20;

side += 50;

}

}

|  |
| --- |
| **Part 5: Functions** |

1. Study the code below for the skyDome function:

function skyDome(diameter, x, y) {

arc(x, y, diameter, diameter, PI - PI/8,

TWO\_PI + PI/8, CHORD);

line(x - diameter \* .4, y + diameter \* .4,

x + diameter \* .4, y + diameter \* .4);

}

Write the function call statement that would draw a skyDome of diameter 100 at the current mouse position:

skyDome(100, mouseX, mouseY);

Write the statement/s (including function call) that would draw a skyDome with diameter that is the remainder of **frameCount** divided by 300, and located at the center of the canvas:

skyDome((frameCount % 300), (width/2), (height/2));

Write the statement/s (including function call) that would draw a skyDome with a random diameter between 100 and 499.99999, and a random location on the canvas (but not hanging off any edge):

Let dia = random(100, 500)),

skyDome(dia,(random(0+dia/2, width-dia/2), (random((0+dia/2, height-dia/2),)));

1. The following code fragment is missing one key statement. Fix the problem by writing in a correction:

/\*\*

\* x2HueJitter – Maps an x coordinate (xCoord) to a hue value

\* and adds a random jitter in the range - to + jitter. Return  
\* a jittery hue number (constrained to 0...360).

\* Example call: let ballHue = x2HueJitter(mouseX, 5);

\* @param xCoord - x coordinate to map to hue  
\* @param jitter - amount of potential jitter to be added to mapped hue

\* @return hue value, number

\*/

function x2HueJitter(xCoord, jitter) {

// Calc hue from x based on range of hue values is 0 to 360

let hueVal = 360.0 \* xCoord / width;

// Add some randomness - some jitter - to the hue

hueVal = hueVal + random( -jitter, jitter );

// Constrain hueVal to 0...360

hueVal = constrain(hueVal, 0, 360);

**return hueVal**

}

}

1. Write a function:

* named createRectangle,
* with no parameters,
* that draws a rectangle to the screen with a random x coordinate between 20 and 50, a y coordinate of 100, a width of 15, and a random height between 0 and 50,
* returns the x coordinate of the rectangle just drawn.

Function createRectangle(){

let xPos = (random(20,50))

let rHei = (random(0,50))

rect( xPos,100, 15, rHei);

return xPos;

}

|  |
| --- |
| **Part 6: Conditionals** |

1. Write an if block that will set the canvas background color according to the current value of the variable, **bestSummer**, which should hold a month name (string):

* "June": "pink"
* "July": "orange"
* "August": "red"
* [any other bestSummer value]: "gray"

If (bestSummer === “June”) {

background(“pink”);

} else if (bestSummer === “July”) {

background(“orange”);

} else if (bestSummer === “August”) {

background(“red”);

} else {

background(“gray”);

}

Refer to the following code fragment for questions 2 through 4:

if (keyIsPressed){

if (key === 'w' || key === 'W') {

background(255);

} else if (key === 'b' || key === 'B') {

background(0);

} else if (key === 'x' || key === 'X') {

background(255 \* mouseX / width);

} else {

background(random(0, 255));

}

}

1. Draw circles around the logical operators.
2. Draw boxes around the comparison operators.
3. Underline the boolean variable.