Practice Problems for Test 1

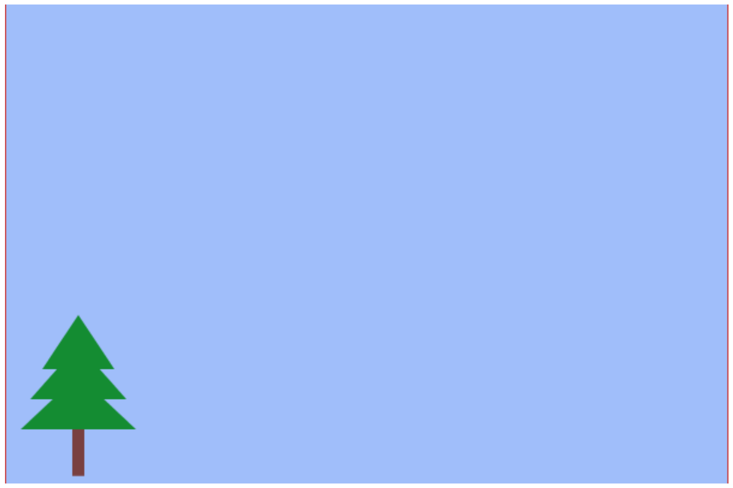
If I give problems that are connected such as Problems 1, 2, 3, 4, I will make sure that they are not dependent upon one another. In that way, if you have issues with one of them, it will not prevent you from being success with another problem/task.

Problem 1

From skills learned by Chapter 2, use triangles to draw a Christmas tree similar to the one shown. There is no need for variables because it will not move.

Specifications:

- Set screen size to 600 X 400
- Must use triangles
- The Christmas tree must be symmetric and colored green. The trunk should be brown.
- There are no other specifications on size or position of tree.
- Save as cedar_tree.pde

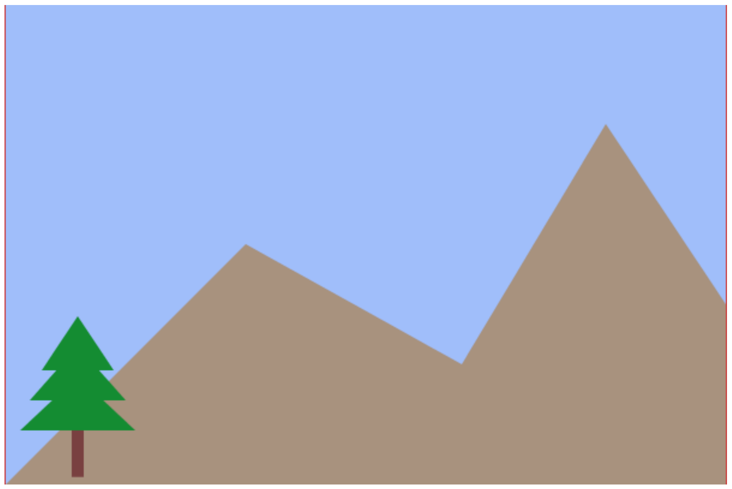


Problem 2

From skills learned by Chapter 2, add to the cedar_tree sketch by drawing two mountains. There is no need for variables because the mountains will not move. Start by opening cedar_tree.pde.

Specifications:

- Use a single set of beginShape/endShape to draw two connected mountains similar to the illustration
- The mountains must be behind the Christmas tree.
- The mountains must be some shade of brown.
- There are no other specifications.
- Save as mountain_tree.pde

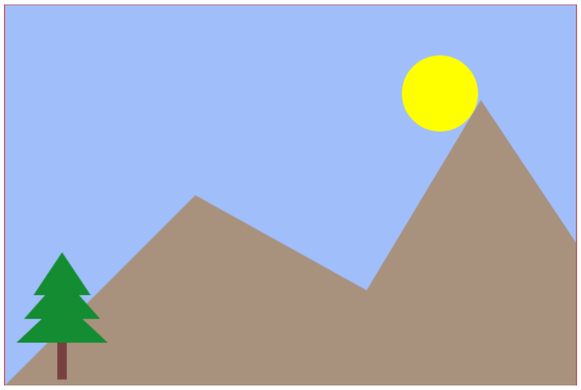


Problem 3

Using skills learned in Chapter 4, add a sunrise to the mountain/tree scene. Start by opening mountain_tree.pde

Specifications:

- The sun starts near the bottom left and is completely hidden by the mountains initially.
- In a diagonal movement, the sun rises from the bottom left to set near the top left. It never leaves the screen completely.
- Save it as sunrise.pde.

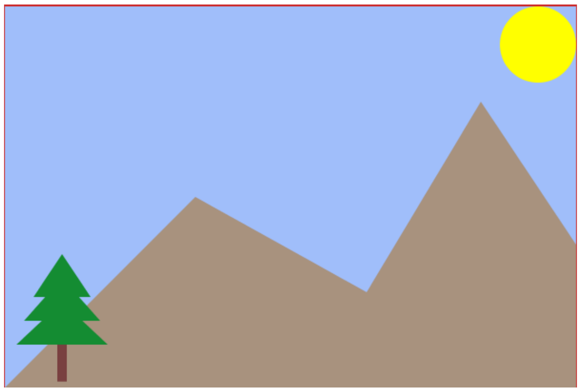


Problem 4

Use the constrain() function (from Chap. 6) to keep the sun from leaving the screen. Start by Opening sunrise.pde.

Specifications:

- There are not specs on exactly where it stops, as long as it's somewhere in top right area.
- Save as constrain_sun.pde.



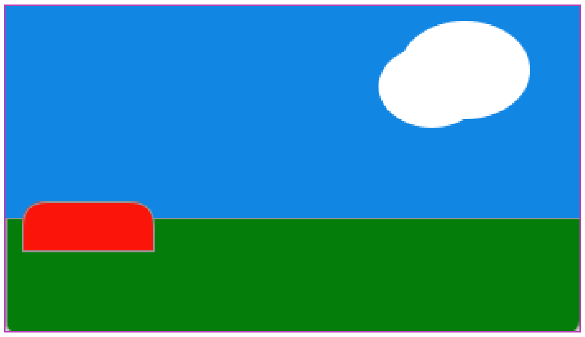
Problem 5

From skills learned in Chapter 4, move the simple green car to the right. Simultaneously, move the simple cloud backwards to the left.

Specifications:

- Begin with the starter code below.
- Put in dynamic mode with draw() and setup()
- Change the necessary constant numbers to variables.
- Write the code to create the movement
- It's OK for them to go off screen.
- Save as opposite_move.pde

```
/* This one is a super simple rendition of a cloud going
   in one direction and car going in another
   Only horizontal movement is happening.
   Start by creating variable(s) and go from there.
*/

  size(350, 200);
  background(#1286E3);

//The cloud
  fill(255);
  noStroke();
  ellipse(280, 40, 80, 60);
  ellipse(260, 50, 65, 50);

  stroke(150);
//The grass
  fill(5,125, 10);
  rect(0, 130, width, height-130);

 //The car
  fill(250, 20, 10);
  rect(10, 120, 80, 30, 15,15,0,0);
```



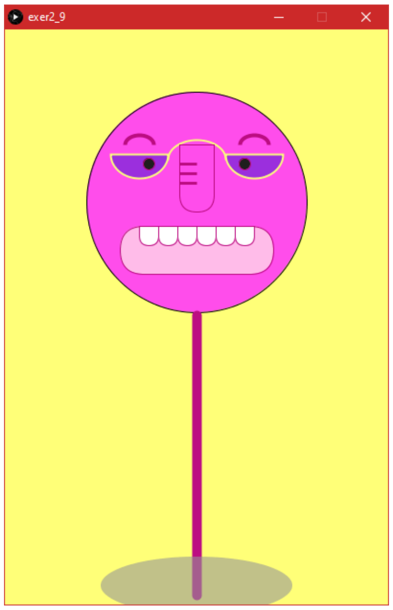
PROBLEM 6

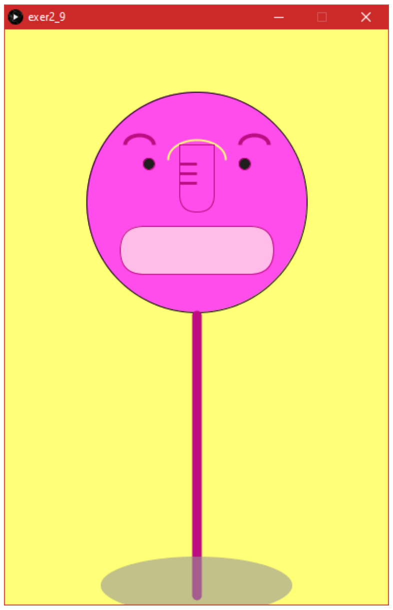
This is my Exercise 2-9 character—Pink Pop. He has lost part of his glasses and his teeth. You are to put glasses and teeth back on the character.

Specifications:

- Start with this code below.
- Use arcs to put the rest of his glasses on. To aid you, I left the top of his glasses on the bridge of his nose. The glasses do not have to be identical to mine but must use arcs.
- Shade the glasses in translucent blue
- Put **at least two** teeth in, using rectangles with round bottoms.
- Save as complete_pop.pde

```
//Exercise 2-9, my Pink Pop
size(400, 600);
background(#ffff78);
rectMode(CENTER);

//hot pink head
fill(#FF4DEB);
circle(200,180,230);

//eyeglasses
stroke(#fafa80);
strokeWeight(2);


noFill();
arc(200, 135, 60, 40, PI, TWO_PI);//yellow bridge of eyeglasses
strokeWeight(1);
stroke(#BC0D82);

rect(200, 155, 36, 70,    0, 0, 20, 20); //Nose

fill(0); //little black eyes
circle(150, 140, 12);
circle(250, 140, 12);
//continuation of nows
 stroke(#BC0D82);
strokeWeight(3);
strokeCap(SQUARE);
line(182, 140, 200, 140);
line(182, 150, 200, 150);
line(182, 160, 200, 160);

//eyebrows
noFill();
strokeWeight(4);
arc(140, 120, 30, 20,   PI, TWO_PI);
arc(260, 120, 30, 20,  PI, TWO_PI);
strokeWeight(1);

//Mouth & teeth
fill(#FFBCE9);
rect(200, 230, 160, 50, 40);
fill(255);

//stick
strokeCap(ROUND);
strokeWeight(10);
line(200,298, 200, 590);

//shadow
noStroke();
fill(150, 150);
ellipse(200, 580, 200,60);
```





Problem 7

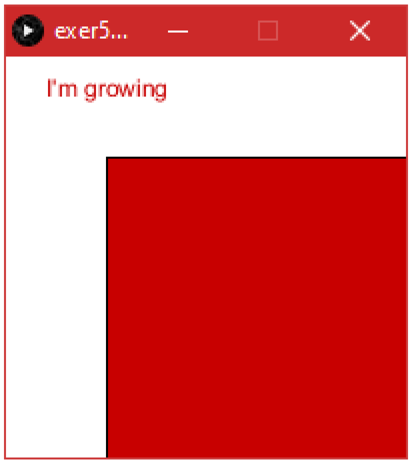
Using Exercise 5-5 as the starter code, write a program where the specified three actions occur when the mouse rolls over the rectangle.

Specifications:

In summary, when mouseX and mouseY hover over the rectangle, the size increases, the color changes, and text displays on screen. Here are the specifics.

1. The size (w & h) increases until it is off the page.
2. The rectangle fills with red.
3. Red text is displayed at 20x and 20y that says, "I'm growing"

The illustration shows what it looks like while the mouse is on it and it has grown off the page.



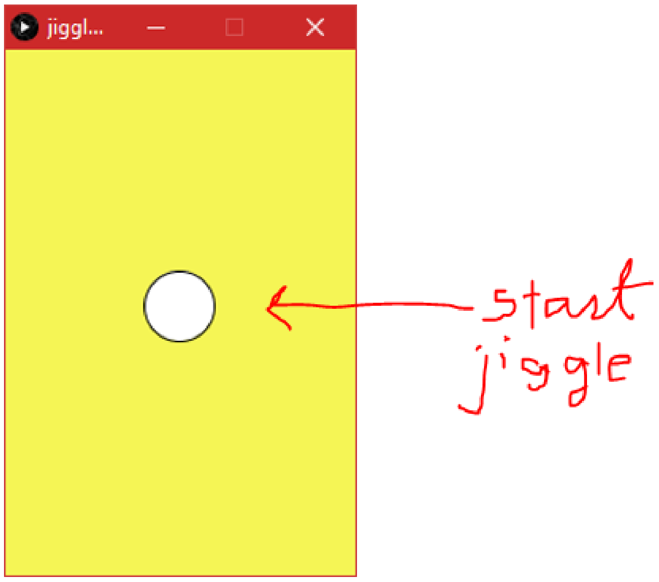
I'm growing

Problem 8

Using the idea from Exercise 4-6, I've created a simplified program consisting of a circle at the bottom of the screen. The goal is to make it move upwards, jiggle when it gets halfway up the screen, then println when it's off the screen.

Specifications:

- Move the circle upwards and ultimately off the screen.
- When the circle gets halfway up the screen, make it shake left and right as it moves up, i.e., it jiggles. (Hint: random, starting with a negative number)
- When it leaves the screen completely, print a line that says "Little Zoog has left the Building"
- Save as jiggle.pde



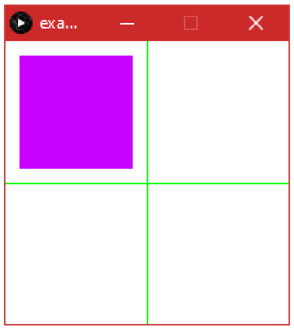
```
//A simplified zoog jiggles off the screen
float x = 100;
float y = 300;
int size = 40;

void setup() {
  size(200, 300);
}

void draw() {
  background(#F5F555);
  circle(x, y, size);
  fill(255);

}
```

Problem 9

Remix Example 5-3 on page 79 as described in the specifications below:

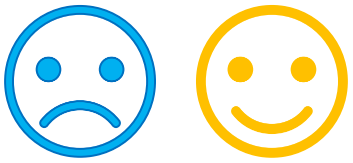
Specifications:

- Create an int variable called **size** and set at 80. Then replace each rectangle's width and height with the **size** variable.
- Set each x and y positions so that the rectangles are centered vertically and horizontally.
- Save as remix_example5_3.pde



Problem 10

Create a button that turns a smiley face on and off in the following manner specified below



Specifications:

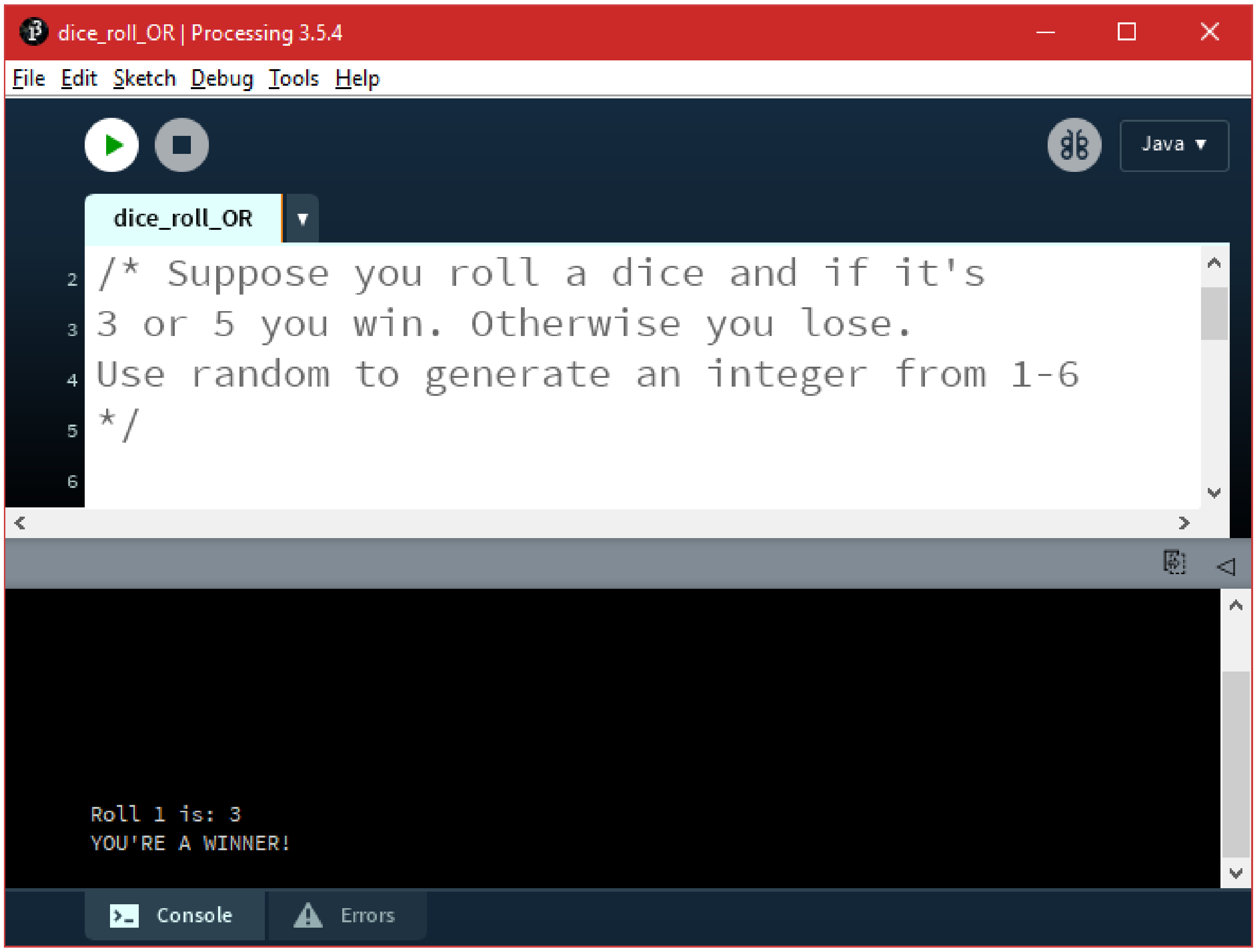
- The size of the window and objects do not matter.
- Create the necessary variables and other parts of the program.
- If the boolean variable is true, the face is yellow and smiling with an arc.
- If variable is false, face is blue and frowning with a different arc. It doesn't matter whether you use fill() or stroke() to specify the sad and happy mood.

Problem 11

Create a program where a 6 sided dice is rolled. If the random number is 3 or 5, display println text “You Win”. Otherwise, display “You lose”.

Specifications:

- Use random() to generate an integer between 1 and 6.
- There are no additional specifications beyond the ones noted above.
- Save as dice_roll.pde.



dice_roll_OR | Processing 3.5.4
File Edit Sketch Debug Tools Help
Java
dice_roll_OR
Console
Errors

```
/* Suppose you roll a dice and if it's
3 or 5 you win. Otherwise you lose.
Use random to generate an integer from 1-6
*/
```

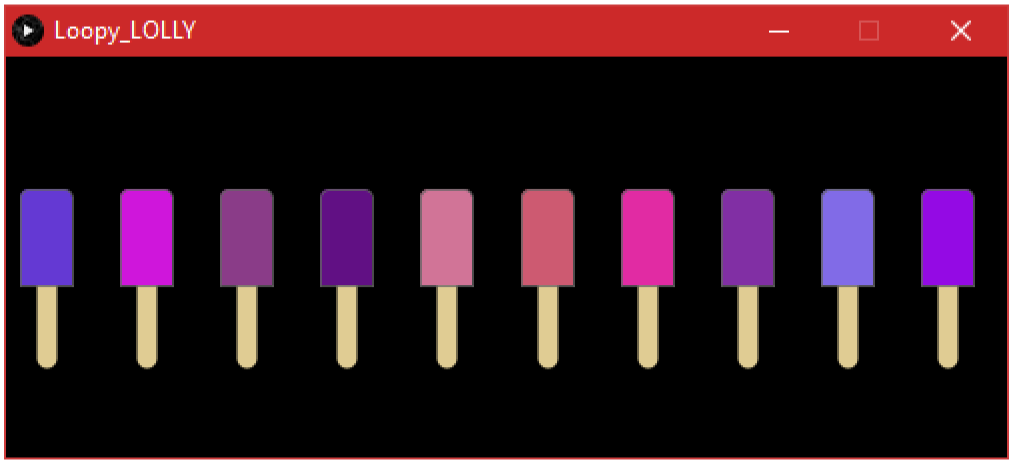
```
Roll 1 is: 3
YOU'RE A WINNER!
```

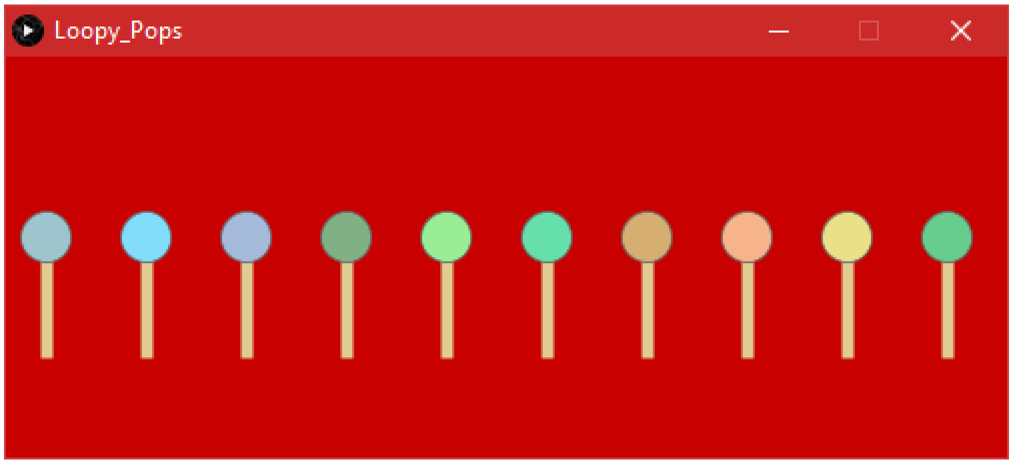
Problem 12

Using the starter code of the sticks, please add lollipops or popsicles to them.

Specifications:

- Use random() to generate random colors of any kind you wish.
- If you create popsicles, they must have round tops and square bottoms, and be centered on the stick.
- If you create lollipops, you must reduce the stroke weight and change the strokecap to square.
- Background does not matter.
- Save as loopy_pops.pde





```
//Here are some sticks; please add lollipops or
popsicles
size(500, 200);
stroke(0);
background(255);

for (int x = 20; x < width; x = x+50) {
  stroke(#E0CC93); //beige for stick
  strokeWeight(10);
  line(x, 150, x, 90);
```