```
/*
  Joan Chirinos
 APCS2 pd08
 HW52 -- Grow and Shrink
 2018-05-22
 */

//instance vars
int sWidth, sHeight;
int circleX, circleY, maxCircleSize, currentCircleSize;
int clicks;
color faveColor;
boolean growing, shrinking, toggleUserInput;

//methods
void setup() {
  //setting screen width and height
  sWidth = sHeight = 500;

  //a 1000x1000 screen
  size(500, 500);

  //background color of screen is black
  background(0, 0, 0);

  //start at 0 clicks
  clicks = 0;

  //circle shall start at size 0
  currentCircleSize = 0;

  //circle should neither be growing nor shrinking at
first
  growing = shrinking = false;

  //toggle user action should be true (to let user
click be valid)
  toggleUserInput = true;

  //set favorite color
  faveColor = color(255, 131, 112);
  //set fill to favorite color
  fill(faveColor);
}

void draw() {
  //after user clicks screen, make the circle grow
  if (growing && currentCircleSize <= maxCircleSize) {
    //user should not be able to click until animation
is over
    toggleUserInput = false;

    //draw circle of current size
    ellipse(circleX, circleY, currentCircleSize,
currentCircleSize);

    //increment current size
    currentCircleSize++;
  } else if (growing && currentCircleSize >=
maxCircleSize) {
    //make the program accept another click
    currentCircleSize--;
    toggleUserInput = true;
    growing = false;
  } else if (shrinking && currentCircleSize >= 0) {
    //clear old circle by drawing new black circle over
it
    fill(0, 0, 0);
    ellipse(circleX, circleY, currentCircleSize + 2,
currentCircleSize + 2);

    //draw new circle
    fill(faveColor);
    ellipse(circleX, circleY, currentCircleSize,
currentCircleSize);

    //decrement circle size
    currentCircleSize--;
  } else if (shrinking) {
    //basically a reset
    shrinking = false;
    toggleUserInput = true;
  }
}

void mouseClicked() {
  if (toggleUserInput) {
    clicks++;
    if (clicks % 2 == 1) {
      growing = true;
      circleX = mouseX;
      circleY = mouseY;
      maxCircleSize = (min(min(abs(sWidth - mouseX),
mouseX), min(abs(sHeight - mouseY), mouseY))) * 2;
      toggleUserInput = false;
    } else {
      shrinking = true;
      toggleUserInput = false;
    }
  }
}
```