Lab 5

- “unnecessary variable”
  - you only need variables for values that change
  - (no points off, just a note)

Lab 5

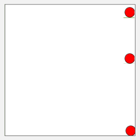
- conditionals questions
  - #1 and #2 gave a step-by-step process for developing the sketch

- Next, update the ellipse's position. Since the horizontal and vertical directions can be considered separately, start with just the horizontal movement. Ask yourself the conditionals questions below, write the answers in comments in your sketch (similar to what was done for the examples in class), and add the corresponding code/code structure for each as you go.
  - Can how to move the ellipse be determined from a snapshot? (Is what to do based on system variables and/or animation variables?)
    [Yes, this is an on-the-spot decision.]
  - How many alternatives are there for what happens? Is do nothing a possibility? This tells you which flavor of `if` is needed — add the right template to your sketch.
  - What are the alternatives? This tells you the body of each case of the `if` — fill in the code for each of the ways the ellipse's position can be updated.
  - How do you decide which alternative to do? This tells you the conditions for each case of the `if`. Fill them in.

Lab 5

```
if ( at right edge ) {
  change direction to down
} else if ( at bottom ) {
  change direction to left
} else if ( at left edge ) {
  change direction to up
} else if ( at top ) {
  change direction to right
}
```

- “conditions ~~not mutually exclusive~~ insufficient”
  - in #2, only considering the position of the ellipse is not quite enough to determine what to do at a given moment
    - e.g. in the example, the x coordinate of all three ellipses is the same but it should change direction to down, move down, or change direction to left



- When the ellipse reaches the edge of the window — if it is moving right and reaches the right edge of the window, it should start moving down instead of continuing to go right; if it is moving down and reaches the bottom edge of the window, it should start moving left instead of continuing to go down; and so forth.

```
if ( moving right and at right edge ) {
  change direction to down
} else if ( moving down and at bottom ) {
  change direction to left
} else if ( moving left and at left edge ) {
  change direction to up
} else if ( moving up and at top ) {
  change direction to right
}
```

Lab 5

- #2 – changing direction on mouse click
  - this is the state machine pattern – have state variable for direction of movement

    the pattern discussed in class for updating the state variable is to have a branch for each value of the state variable, with the conditions being when to change to that value

```
void mouseClicked () {
  if ( direction == 0 ) {
    direction = 1;
  } else if ( direction == 1 ) {
    direction = 2;
  } else if ( direction == 2 ) {
    direction = 3;
  } else {      // direction == 3
    direction = 0;
  }
}
```

    clever arithmetic using mod (%) is valid (no points off), but make sure that you understand the state machine pattern

```
void mouseClicked () {
  direction = (direction+1)%4;
}
```

  - use the patterns discussed in class
    - show the TFs the relevant handouts if you are getting help from them
    - if you are intentionally using an alternative solution because you understand the pattern and know of a better way, include a comment to that effect