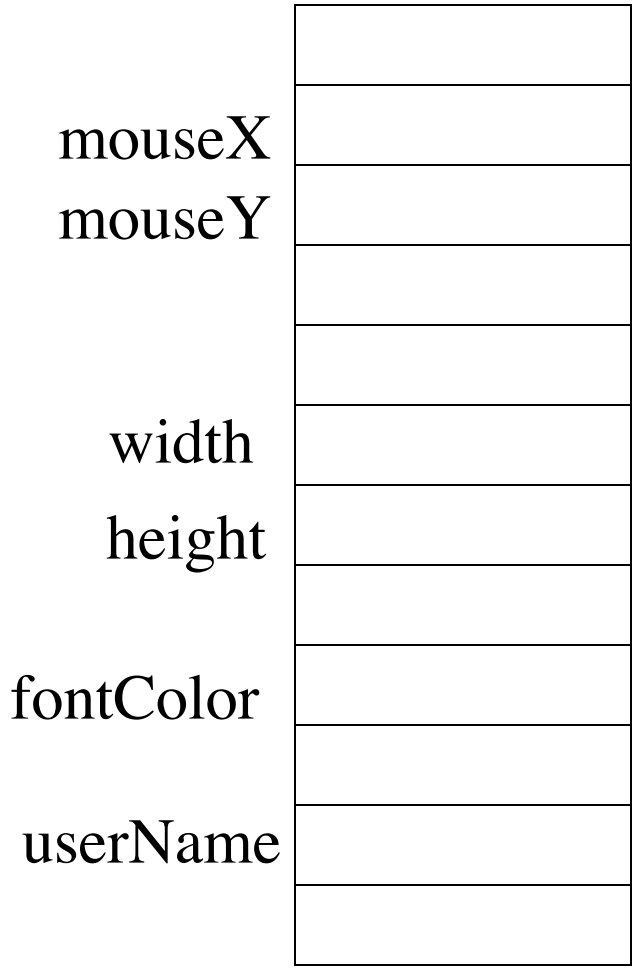
What is a variable?

- a named location in the computer's memory



Variables

- store/remember values
- can be changed
- must be declared to store a particular kind of value (e.g. whole number, fraction, character, color, image, boolean)
- should have a descriptive name
  - start with letter
  - then also include numbers
  - no spaces

```
void setup() {
  size(500,500);
}
int diameter = 0;
void draw() {
  ellipse(width/2, height/2, diameter,diameter);
  diameter = diameter + 5;
}
```

What does this display?

A) many nested circles (like a target) growing in size

B) a white circle growing in size

C) a black circle growing in size

D) a pulsing images of circles growing and shrinking

E) nothing (a circle with diameter 0)

primitive types (Java)

- boolean - true or false
- char - ‘a’ , ‘b’ , ‘c’ , …
- byte - small integer -128 to 127
- short - bigger integer -32768 to 32767
- int - even bigger integer +/- 2 billion
- long - really big integer
- float - numbers with fractional parts 3.1415
- double - like float but more precision

declaring vs initializing vs assigning

```
int carFront; // declare
int carFront = 100; // declare and initialize
carFront = carFront - 1; // assign
```

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

void draw() {
  background(255);
  int xPos = 0;
  ellipse(xPos, height/2, 20, 20);
  xPos = xPos + 1;
}
```

What does this draw?

A) circle moving across the screen left to right

B) circle moving across the screen right to left

C) circle moving across the screen top to bottom

D) circle moving across the screen bottom to top

E) half-circle on the left edge not moving

System Variables (Processing)

- mouseX, mouseY
- pmouseX, pmouseY
- width, height
- frameCount

```
void setup() {
  size(400, 400);
}

void draw() {
  fill(255-abs(mouseX-pmouseX));
  rect(pmouseX, pmouseY, mouseX, mouseY);
}
```

```
void setup() {
  size(400, 400);
  __________________ // position A
}

____________________ // position B

void draw() {
  __________________ // position C
  fill(255-abs(mouseX-pmouseX));
  rect(pmouseX, pmouseY, mouseX, mouseY);
  __________________ // position D
}
```

Where should the line "background(255);" be placed so that the sketch shows just a single moving rectangle?

Choose option E if it would work with either C or D.

Making Choices

- **If** you wish to defrost, press the defrost button; otherwise press the full power button.
- Let the dough rise in a warm place **until** it has doubled in size.
- **If** the ball reaches the side of the display change it's direction.

Boolean Expressions

- Any expression that evaluates to true or false.
- Relational operators, <, <=, >, >=.
- Equality operators, ==, !=.
- For example:

```
int i = 3, j = 4;

5 < 6
i == j
(j + 2) <= 6
```

Expressions and Statements

- Expression statements are formed by adding a semicolon to the end of certain types of expressions.
  - An assignment expression is an expression involving an assignment.
    ```
    area = width * height;
    ```
  - A method call expression has no assignment.
    ```
    rect(…);
    ```

Non-statements

- Not all expressions can be turned into statements. The following are syntax errors.
  - x+y;
  - width > 20;
- The above do not make sense as statements. They don't DO anything. Statements must DO something.
  - assign a new value to a variable
  - cause some output to occur (println(), rect())
  - change some internal "state" (background (255), noStroke())

Blocks

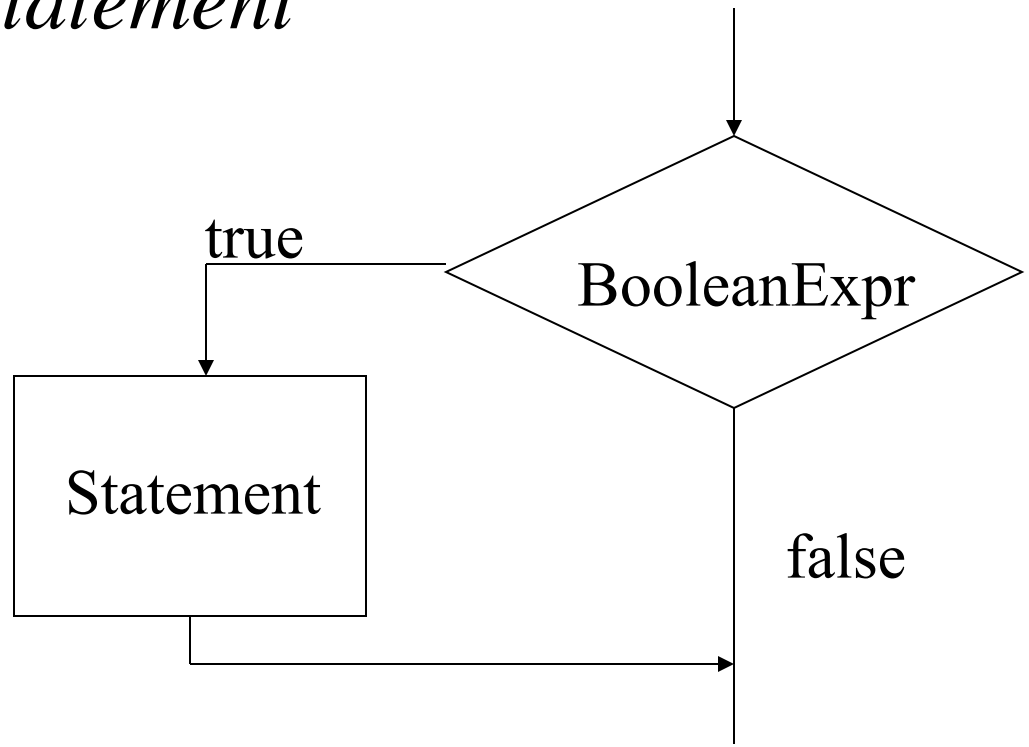
Several statements can be grouped into a block using { }.

```
{
  int x = 20, y = 30, size = 40;
  ellipseMode(CORNER);
  fill(255, 0, 0);
  rect(x, y, size, size);
  fill(0, 255, 0);
  ellipse(x, y, size, size);
}
// x, y, and size above cannot be used
// here
```

The `if` statement

`if` `(` *BooleanExpression* `)`

Statement



```
int count = 0;

void setup() {
  frameRate(2);
}

void draw() {
  background(120);
  int x = 20, y = 30, size = 40;
  if (count % 2 == 0) {
    ellipseMode(CORNER);
    fill(255, 0, 0);
    rect(x, y, size, size);
    fill(0, 255, 0);
    ellipse(x, y, size, size);
  }

  count = count + 1;
}
```

```
int ballX, ballDia = 50;
void setup() {
  size(400,400);
  ballX = -ballDia/2;
}
void draw() {
  background(120);
  if (ballX > width+ballDia/2)
    ballX = -ballDia/2;
  ellipse(ballX, height/2, ballDia, ballDia);
  ballX = ballX + 1;
}
```

A. Ball moves across jumping back to left edge as soon as it touches the right edge.

B. Ball moves across moving off the right edge then moves back in from the left edge.

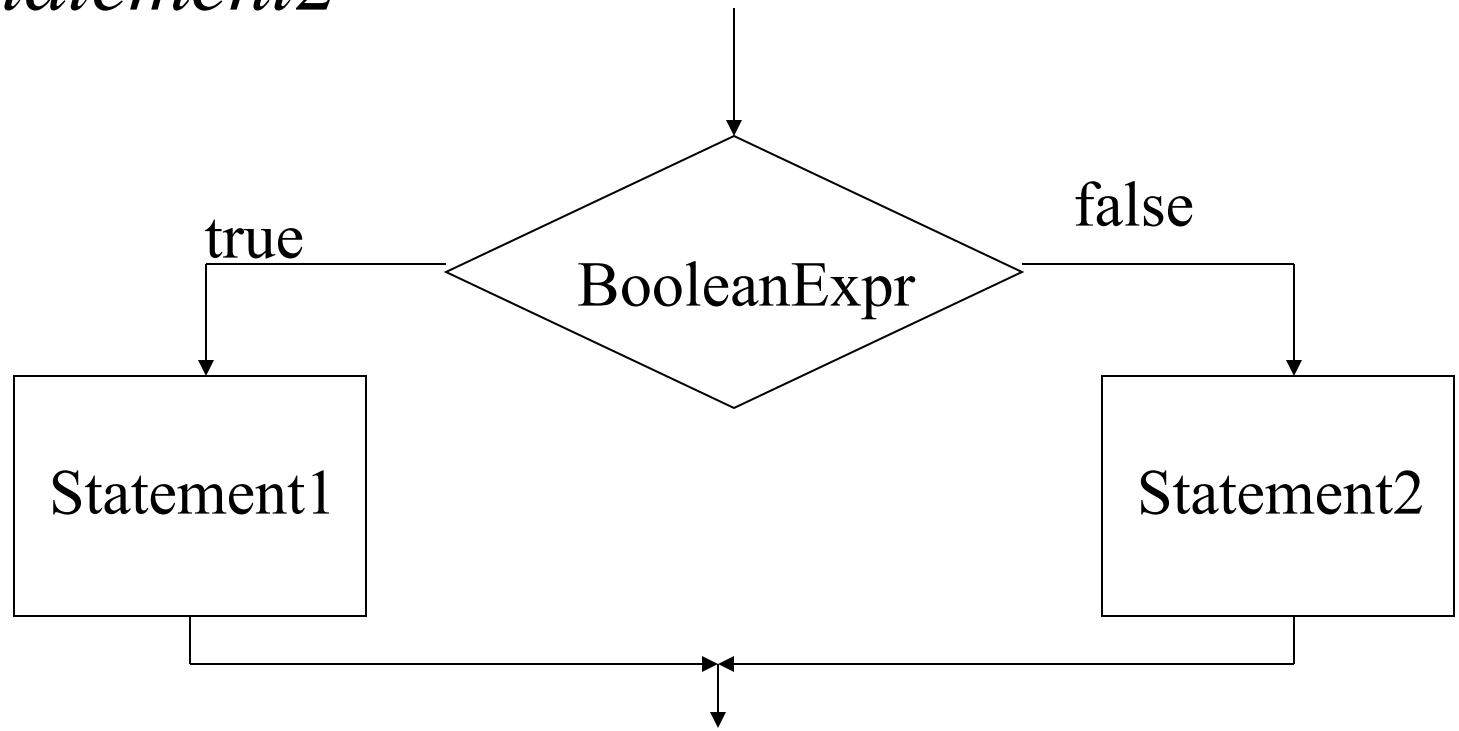
C. Ball moves across until half way off the right edge (showing just a half circle) then reappears as a half circle on the left edge.

D. Ball moves across then disappears and doesn't come back.

E. Ball moves left to right then right to left after reaching the right edge, and repeats.

The `if-else` statement

```
if ( BooleanExpression )
   Statement1
else
   Statement2
```



```
void draw() {
  background(120);
  int x = 20, y = 30, size = 40;
  ellipseMode(CORNER);
  if (count % 2 == 0) {
    fill(255, 0, 0);
    rect(x, y, size, size);
    fill(0, 255, 0);
    ellipse(x, y, size, size);
  }
  else {
    fill(0, 255, 0);
    rect(x, y, size, size);
    fill(255, 0, 0);
    ellipse(x, y, size, size);
  }

  count = count + 1;
}
```

```
int ballX, ballDia = 50, speed = 1;
void setup() {
  size(400,400);
  ballX = ballDia/2;
}
void draw() {
  background(120);
  if (ballX > width-ballDia/2) {
    speed = -1;
  }
  else _____________________________ {
    speed = 1;
  }
  ellipse(ballX, height/2, ballDia, ballDia);
  ballX = ballX + speed;
}
```

A. Leave it blanks (as is).
B. if (ballX > ballDia/2)
C. if (ballX < ballDia/2)
D. (ballX > ballDia/2)
E. (ballX < ballDia/2)

What goes in the blank so that the ball moves back and forth, reversing direction whenever it reaches the edge?

Semicolons and the if statement

```
if (carX < 0)
  carX = width;
// draw car
...
```

```
if (carX < 0) {
  carX = width;
  carY = carY + 10;// move down each time it wraps around
}
```

Semicolons and common error

```
void draw() {
  if (carX < 0);
    carX = width;
  // draw car
  ...
}
```

Logical Operators

- Operators that take boolean values as operands.
- x && y - true if x AND y are both true
- x || y - true if either x OR y are true, or both
- !x - true if x is false - read NOT x

```
void setup() {
  size(200, 200);
  rectMode(CORNERS);
}

int boxLeft = 50, boxRight = 150,
    boxTop = 50, boxBottom = 150;
void draw() {
  if (mouseX > boxLeft && mouseX < boxRight &&
      mouseY > boxTop && mouseY < boxBottom )
  {
      fill(255,0,0);
  }
  else {
    fill(0,255,0);
  }
  rect(boxLeft, boxTop, boxRight, boxBottom);
}
```

```
int circleX, circleY, dia = 40;
void setup() {
  circleX = width/2;
  circleY = height/2;
}
void draw() {
  if (_____________________________)
      fill(255,0,0);
  else
    fill(0,255,0);
  ellipse(circleX, circleY, dia, dia);
}
```

Which expression completes this program so that it shows a red circle when the mouse is inside of the circle and a green circle when the mouse is outside of the circle?

A. `dist(mouseX, mouseY, circleX, circleY) <= dia/2`

B. `dist(mouseX, mouseY, circleX, circleY) >= dia/2`

C. `dist(mouseX, mouseY, circleX, circleY) < dia`

D. `dist(mouseX, mouseY, circleX, circleY) > dia`

E. `abs(mouseX-circleX) < dia/2 && abs(mouseY-circleY) < dia/2`

Bouncing Ball

- $pos_{t+1} = pos_t + velocity$
- $velocity_{t+1} = velocity_t + acceleration$
- gravity provides constant acceleration downward

```
float velocity = 2;
float yPos = 0;
int ballRadius = 10;

void draw() {
  background(255);

  // if hit the ground reverse the velocity
  if (yPos > height-ballRadius) {
    velocity = -velocity;
  }
  // adjust position based on velocity
  yPos = yPos + velocity;

  // draw the ball
  ellipse(width/2, yPos, ballRadius*2, ballRadius*2);
}
```

```
float velocity = 2, yPos = 0;
int ballRadius = 10;
float gravity = 0.1;
void draw() {
  background(255);
  // if hit the ground reverse the velocity
  if (yPos > height-ballRadius) {
    velocity = -velocity;
  }
  // adjust position based on velocity
  yPos = yPos + velocity;

  // adjust the velocity - increasing due to gravity
  velocity = velocity + gravity;

  // draw the ball
  ellipse(width/2, yPos, ballRadius*2, ballRadius*2);
}
```

```
void draw() {
  background(255);
  // if hit the ground reverse the velocity
  if (yPos > height-ballRadius) {
    velocity = -velocity;
  }
  // adjust position based on velocity
  yPos = yPos + velocity;

  // adjust the velocity - increasing due to gravity
  velocity = velocity + gravity;

  //  add some drag
  velocity = velocity*0.99;

  // draw the ball
  ellipse(width/2, yPos, ballRadius*2, ballRadius*2);
}
```

Recap

- boolean valued expressions using relational operators: <, >, <=, >=, ==, !=
- boolean operators &&, ||, !
- if (booleanExpression) {
  sequence of statements
  }
  else {
  sequence of statements
  }