Programming in Processing
Making Choices

- **If** the RPMs are higher than 4000 then shift into the next gear.
- Let the dough rise on the counter **until** it has doubled in size.
- **If** the ball reaches the bottom of the screen change its direction.
• Any expression that evaluates to true or false is a *boolean expression*.

• Making boolean expressions:
  – Relational operators <, <=, >, >=
  – Equality operators ==, !=

• Example:
```java
int i = 2;
int j = 5;

i < j
i == j
(j + 2) <= 10
```
• *Statements* are formed by adding a semicolon at the end of certain types of expressions.
  
  – An assignment
    
    ```java
    area = width * height;
    ```
  
  – A method call
    
    ```java
    rect(...);
    ```
Non-statements

• Not all expressions can be considered statements.
• Syntax errors:
  
  x+y;
  width > 20;
  
  – These don't make sense as statements. They don't *do* anything.

• Statements must *do* something
  
  – assign a new value to a variable (x = 5;)
  – cause output to occur (println(), rect())
  – change some internal state (background())
Blocks

• Several statements can be grouped into a block using `{ }`. A block is a statement.

```java
{ 
    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

```plaintext
if (BooleanExpression)  
Statement
```

Diagram:
- If `BooleanExpr` is `True`, execute `Statement`.
- If `BooleanExpr` is `False`, do nothing.

---

Consideration: The `if` statement is a fundamental control structure in programming that allows the execution of code based on a condition. It is used extensively in various programming languages. The diagram illustrates how the condition is evaluated, and the corresponding block of code is executed based on the result.
int count = 0;

void setup() {
    frameRate(2);
}

void draw() {
    background(120);
    if (count % 2 == 0) {
        ellipseMode(CORNER);
        fill(255, 0, 0);
        rect(20, 30, 40, 40);
        fill(0, 255, 0);
        ellipse(20, 30, 40, 40);
    }
    count = count + 1;
}
What does the code do?

```cpp
int ballX, ballDia = 100;
void setup() {
    size(400, 400);
    ballX = 0;
}
void draw() {
    background(120);
    if (ballX > width)
        ballX = 0;
    ellipse(ballX, height/2, ballDia, ballDia);
    ballX = ballX + 2;
}
```

A. Ball moves across until halfway off the right edge (showing half a circle) then jumps back to left side, repeats
B. Ball moves across then disappears and doesn't come back
C. Ball moves left to right then right to left after touching right edge, repeats
D. Ball moves across jumping back to left edge as soon as it touches the right edge
E. Ball moves across moving off the right edge entirely then moves in from the left edge
The if else statement

```java
if (BooleanExpression)
    Statement
else
    Statement
```

Diagram:
- **BooleanExpr**
- **True**: Statement
- **False**: Statement
```java
int count = 0;
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;
}
```
Fixing the code

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

A. Leave blank
B. if (ballY > ballDia/2)
C. if (ballY < ballDia/2)
D. (ballY > ballDia/2)
E. (ballY < ballDia/2)
```

- Which code goes into the blank line to make the ball move up and down, reversing direction when it hits the edges?
Semicolons

```java
if (posX < 0)
    posX = 50;

if (posX < 0) {
    posX = 50;
    posY = posY + 20;
}

if (posX < 0);
    posX = 50;
posY = posY + 1;
```
Logical Operators

• Operators that take boolean values as inputs
  • $x \&\& y$
    - is true if $x$ and $y$ are both true ($x$ AND $y$)
  • $x \|\| y$
    - is true if $x$ or $y$ are true or both ($x$ OR $y$)
  • $!x$
    - is true if $x$ is false (NOT $x$)
```java
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);
}
```
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);
}

How would you make it turn yellow when the mouse is not over the rectangle?
Bouncing Ball

• Simulating gravity
  - $\text{pos}_{t+1} = \text{pos}_t + \text{velocity}$
  - $\text{velocity}_{t+1} = \text{velocity}_t + \text{acceleration}$
  - Gravity is a constant acceleration downwards
Gravity

• (Demo)
```java
if (BooleanExpression) {
    Statements
}
else {
    Statements
}

Relational operators
x < y
x <= y
x > y
x >= y
x == y
x != y

Boolean operators
p && q
p || q
!p
```
Repetition with for loops
Repetition

- Repeating commands is powerful
  - Lightbot 2.0 used recursion
    - function calls itself
  - Symbolic Lightbot used a prefix number
    - 4:Step means do Step Step Step Step
  - Processing uses a `for` loop

```java
void setup() {
  size(500, 200);
  background(0);
  fill(255);
  for(int i = 0; i < 16; i++) {
    ellipse(100 + 25*i, 100, 15, 15);
  }
}
```
Here are the parts of a for loop (all are required):

```java
for (int i = 0; i < 16; i++) {
    // stuff to be repeated
    ellipse(100 + 25*i, 100, 15, 15);
}
```

Result: we get 16 copies of the ellipse Pac-Man pills!
What does the above code do?
A. Draws 8 pellets horizontally
B. Draws 9 pellets horizontally
C. Draws 8 pellets vertically
D. Draws 7 pellets vertically
E. Doesn't draw any pellets
Bullseye

• Let's draw one frame with concentric filled ellipses
  – bullseye

```java
void setup() {
    size(400, 400);
    background(0);
    for(int i = 0; i < 10; i++) {
        ellipse(200, 200, 200-20*i, 200-20*i);
    }
}
```
Playing around

- (demo)
Nested for loops

• You are allowed to put a for loop inside of another for loop
  – Example:
    • Repeat 3 times
      – Repeat 5 times
        • knit, purl, purl
      – Yarn over
    – Purl

(demo)
End