Expressions & Conditionals

CSE 120 Winter 2019

Instructor: Justin Hsia

Teaching Assistants: Ann Shan, Eunia Lee, Pei Lee Yap, Sam Wolfson, Travis McGaha

Delivery Robot Engulfed in Flames, Honored on Campus With Candlelight Vigil

"The University of California, Berkeley, lost a beloved member of their campus last week, when a delivery robot was engulfed in flames outside the student union.

"The courier rovers started delivering food to students about two years ago. The service seems to have been operating relatively smoothly until… a Kiwibot spontaneously combusted while on the job.

"After an investigation, the company revealed … 'a defective battery was put in place of a functioning one. This caused an exceedingly rare occurrence of the battery experiencing thermal runaway.' "

- https://gizmodo.com/delivery-robot-engulfed-in-flames-honored-on-campus-wi-1831145871

Administrivia

- Assignments:
  - Animal Functions due tonight (1/28)
  - Reading Check 4 due Thursday @ 3:30 (1/31)
  - Jumping Monster due Friday (2/1)

↑ significantly harder!

- "Big Ideas" this week: Digital Distribution

Outline

- **Expressions & Operators**
- Conditionals
  - `if`
  - `else`
  - `else if`

Expressions

- "An expression is a combination of one or more *values*, *constants*, *variables*, *operators* and *functions* that the programming language interprets and computes to produce another value."
  - https://en.wikipedia.org/wiki/Expression_(computer_science)

- Expressions are *evaluated* and resulting value is used
  - Assignment: `x = x + 1;`
  - Assignment: `x_pos = min(x_pos + 3, 460);`
  - Argument: `ellipse(50+x, 50+y, 50, 50);`
  - Argument: `drawMouse(rowX+4*50,rowY,rowC);`

Operators

- Built-in "functions" in Processing that use special symbols:
  - Multiplicative: * mult, / div, % modulus
  - Additive: + add, - sub
  - Relational: < less than, > greater than, <= less than or equal to, >= greater than or equal to
  - Equality: == equal to, != not equal to
  - Logical: && and, || or, ! not

new

- Operators can only be used with certain data types and return certain data types
  - Multiplicative/Additive: 1+2 give numbers, get number (3)
  - Relational: 1 < 5 give numbers, get Boolean (true)
  - Logical: true && true give Boolean, get Boolean (true)
  - Equality: color(0) == color(255) give same type, get Boolean (false)

Operators

- Built-in "functions" in Processing that use special symbols:
  - Multiplicative: * mult, / div, % modulus
  - Additive: + add, - sub
  - Relational: < less than, > greater than, <= less than or equal to, >= greater than or equal to
  - Equality: == equal to, != not equal to
  - Logical: && and, || or, ! not

- Logical operators use Boolean values (`true`, `false`)

AND (&&) — put in-between

x	y	x && y
false	false	
false	true	
true	false	
true	true	

OR (||) — put in-between

x	y	x || y
false	false	
false	true	
true	false	
true	true	

NOT (!) — put in front

x	!x
false	
true	

Operators

- Built-in "functions" in Processing that use special symbols:
  - Multiplicative: `*` mult, `/` div, `%` modulus
  - Additive: `+` add, `-` sub
  - Relational: `<` less than, `>` greater than, `<=` less than or equal to, `>=` greater than or equal to
  - Equality: `==` equal to, `!=` not equal to
  - Logical: `&&` and, `||` or, `!` not

- In expressions, use parentheses for evaluation ordering and readability
  - e.g. `x + (y * z)` is the same as `x + y * z`, but easier to read

order of operations!

`(x + y) * z` is required if you want addition to happen first.

Modulus Operator: %

- `x % y` is read as "`x` mod `y`" and returns the remainder after `y` divides `x`
  - For short, we say "mod" instead of modulus

- Example Uses:
  - Parity: Number `n` is even if `n%2 == 0` (is even if divisible by 2)
  - Leap Year: Year `year` is a leap year if `year%4 == 0` (divisible by 4) (e.g. 2016, 2020)
  - Chinese Zodiac: `year1` and `year2` are the same animal if `year1%12 == year2%12` (12 Zodiac animals)

Conditionals Worksheet

- Work just on **Page 1 (Questions 1-6)**

- Operators:
  - Arithmetic: + - * / %
  - Relational: < > <= >=
  - Equality: == !=
  - Logical: && || !
- Data Types:
  - Arithmetic: give numbers, get number
  - Relational: give numbers, get Boolean
  - Logical: give Boolean, get Boolean
  - Equality: give same type, get Boolean

Modulus Example in Processing

- Use mod to "wrap around"
  - Replace `min`/`max` function to "connect" edges of drawing canvas

x_pos = 459;

- `x_pos = min(x_pos + 3, 460); // stores 460`
- `x_pos = (x_pos + 3) % 460; // stores 2`

462

462

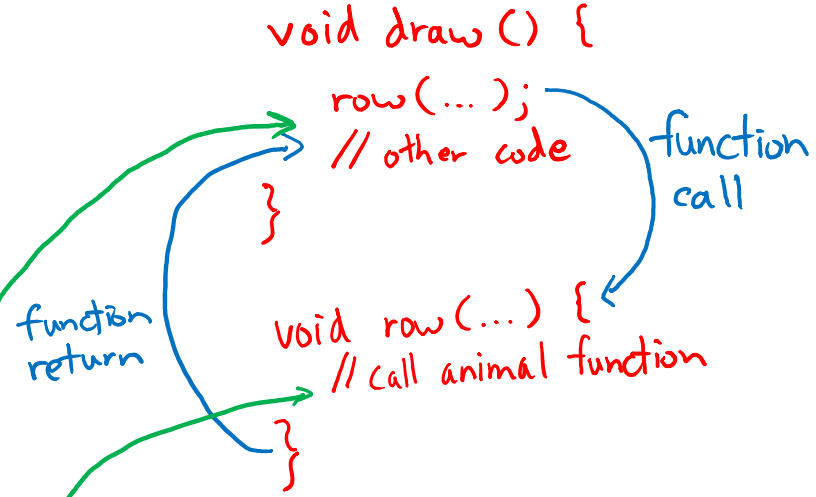
right edge of canvas

left side of canvas

Control Flow

- The order in which instructions are executed

- We typically say that a program is executed in sequence from top to bottom, but that's not always the case:
  - last week: Function calls and `return` calls
  - today: Conditional/branching statements
  - next week: Loops



- Curly braces { } are used to group statements
  - Help parse control flow
  - Remember to use indentation!

Outline

- Expressions & Operators
- **Conditionals**
  - `if`
  - `else`
  - `else if`

If-Statements

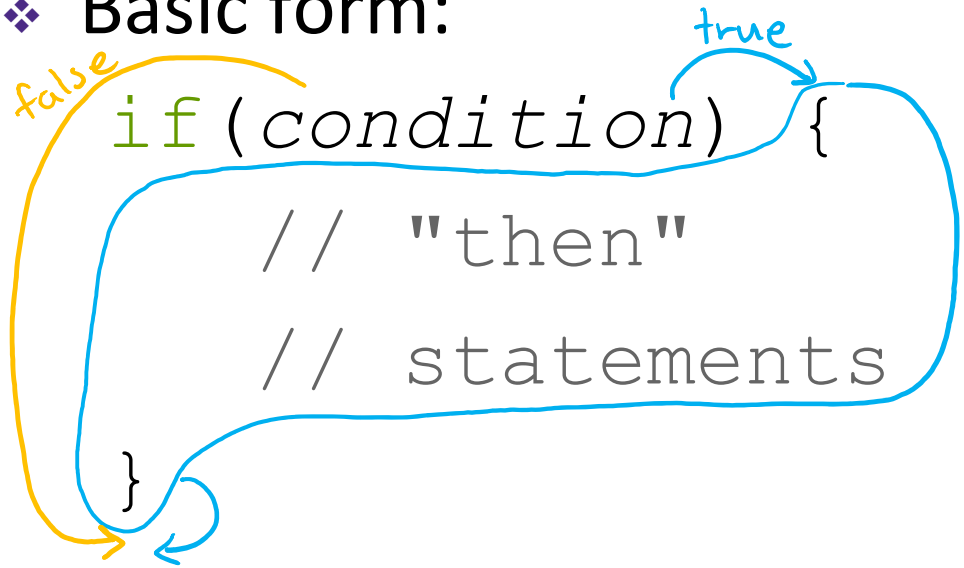
- Sometimes you don't want to execute *every* instruction
  - Situationally-dependent

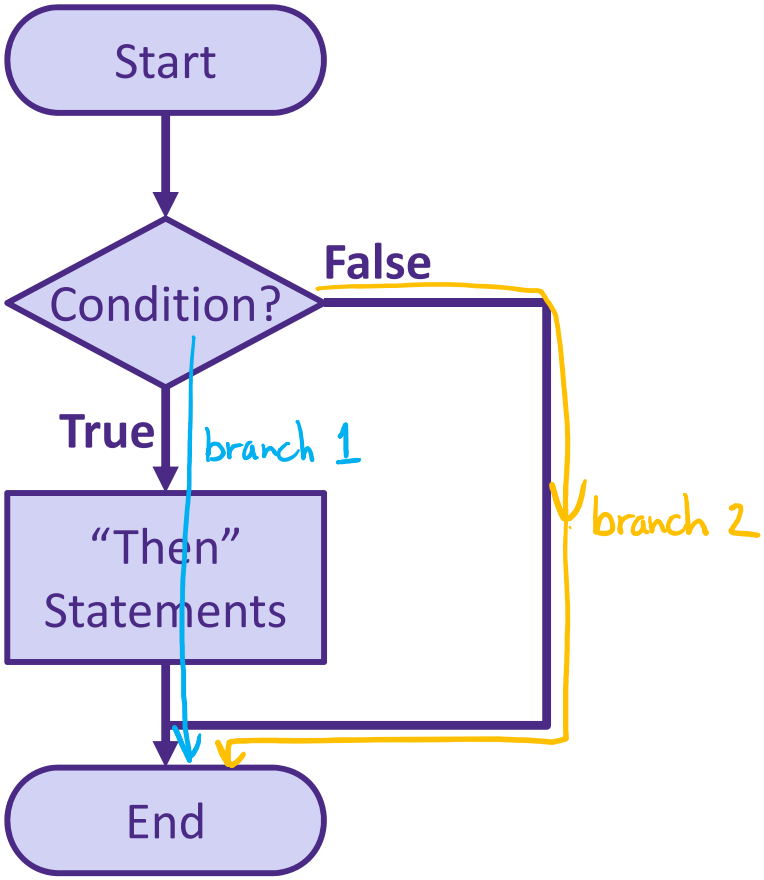
- Conditionals give the programmer the ability to make decisions
  - The next instruction executed depends on a specified *condition*
    - The condition must evaluate to a `boolean` (*i.e.* `true` or `false`)
    - Sometimes referred to as "branching"
  - This generally lines up well with natural language intuition

If-Statements

- Basic form:

```
if (condition) {
    // "then"
    // statements
}
```





- Example conditions:
  - Variable: `if( done == true )`
  - Variable: `if( done )`
  - Expression: `if( x_pos > 460 )`
  - Expression: `if( x_pos > 100 && y_pos > 100 )`

boolean

equivalent

number

cond 1 AND cond 2

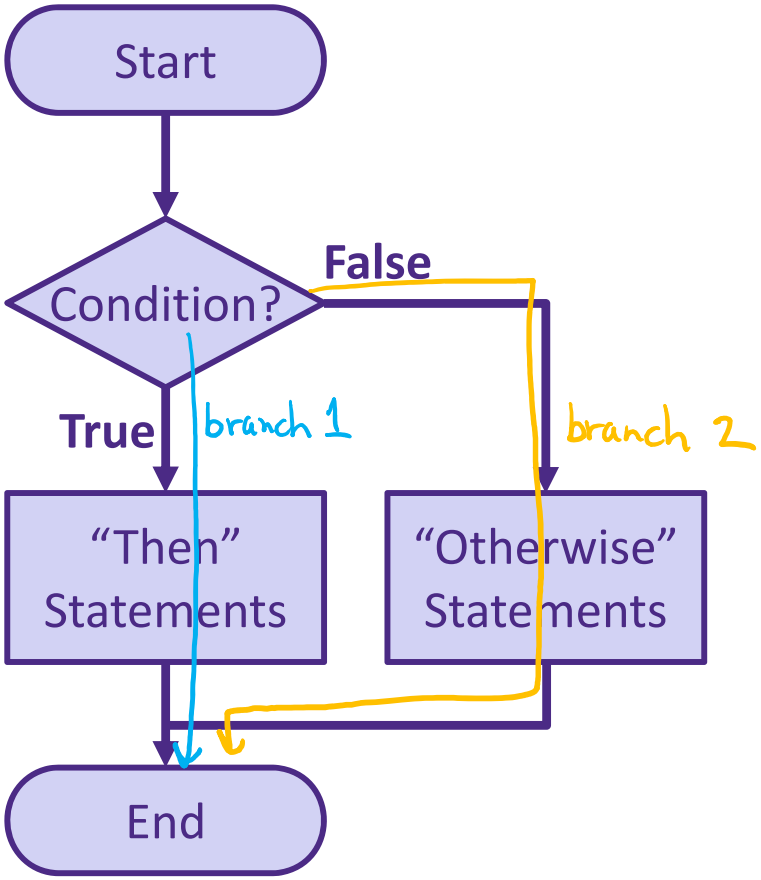
If-Statements

- With `else` clause:

```
if(condition) {
    // "then"
    // statements
} else {
    // "otherwise"
    // statements
}
```

true

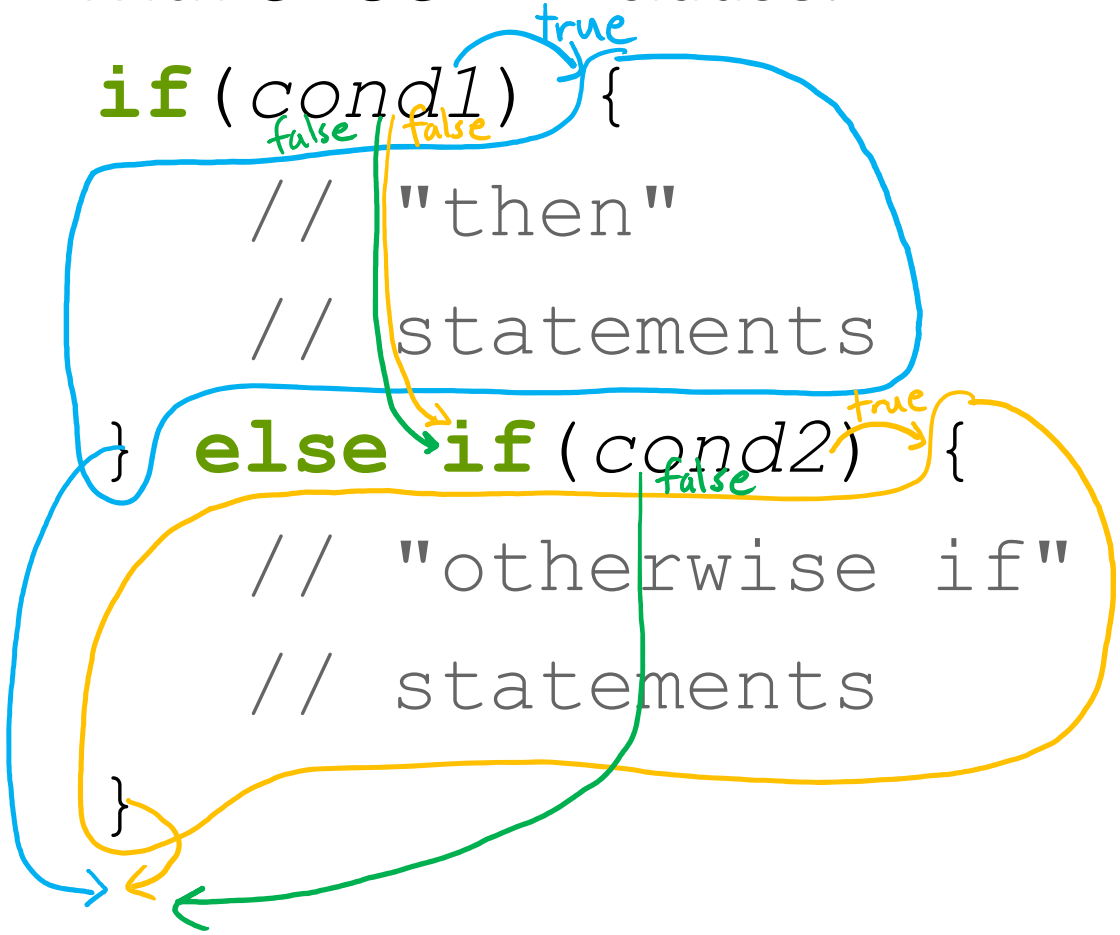
false

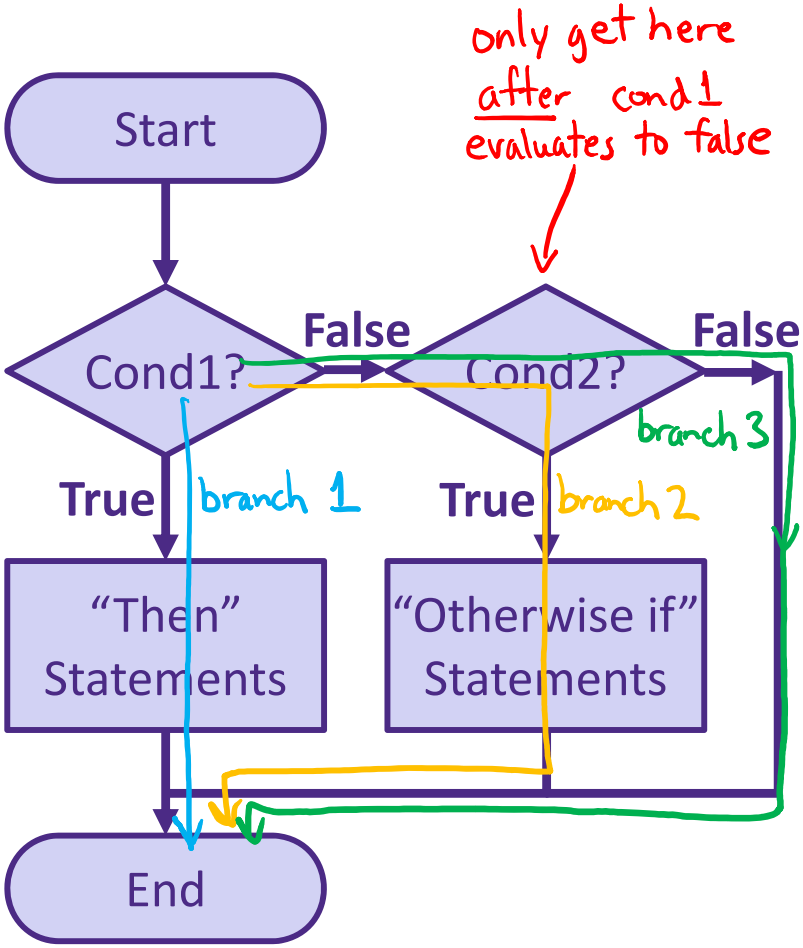


If-Statements

- With `else if` clause:

```
if (cond1) {
    // "then"
    // statements
} else if (cond2) {
    // "otherwise if"
    // statements
}
```



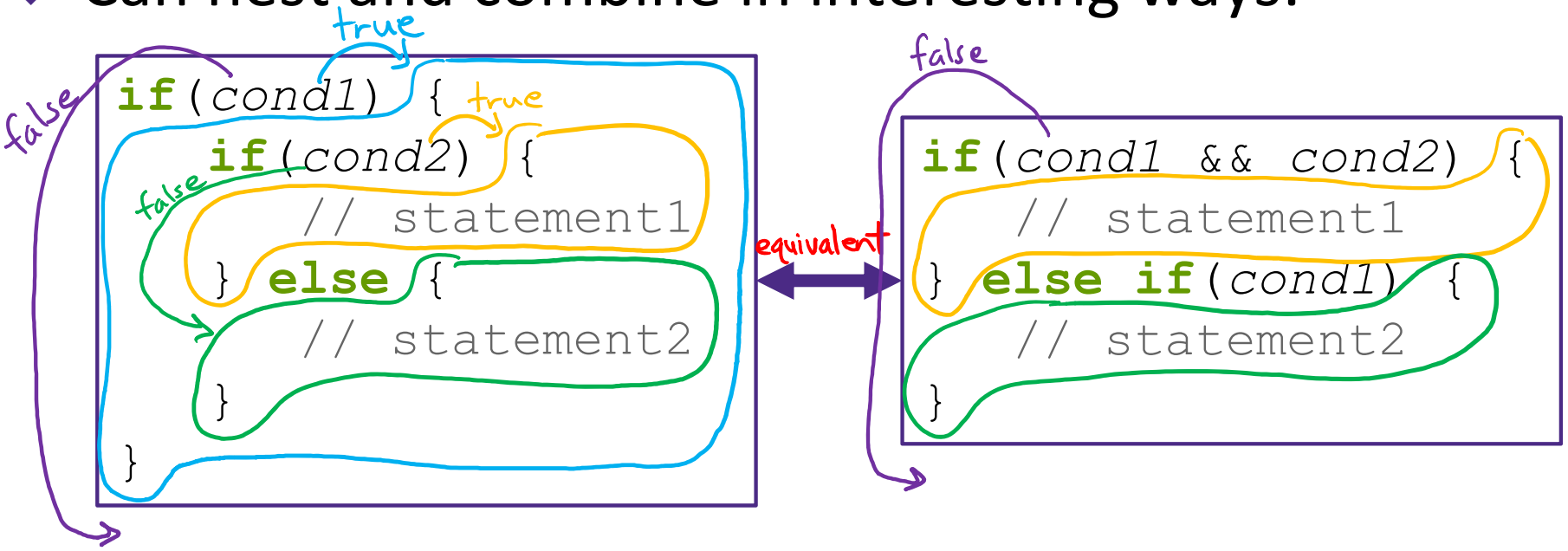


If-Statements

- Notice that conditionals *always* go from Start to End
  - Choose one of many *branches*
  - A conditional must have a single `if`, as many `else if` as desired, and at most one `else`

"catch all" / default

- Can nest and combine in interesting ways:



Practice Question

- Which value of `x` will get the following code to print out `"Maybe"`?

A. 1 No

B. 3 Maybe

C. 5 Yes

D. 7 No

E. We're lost…

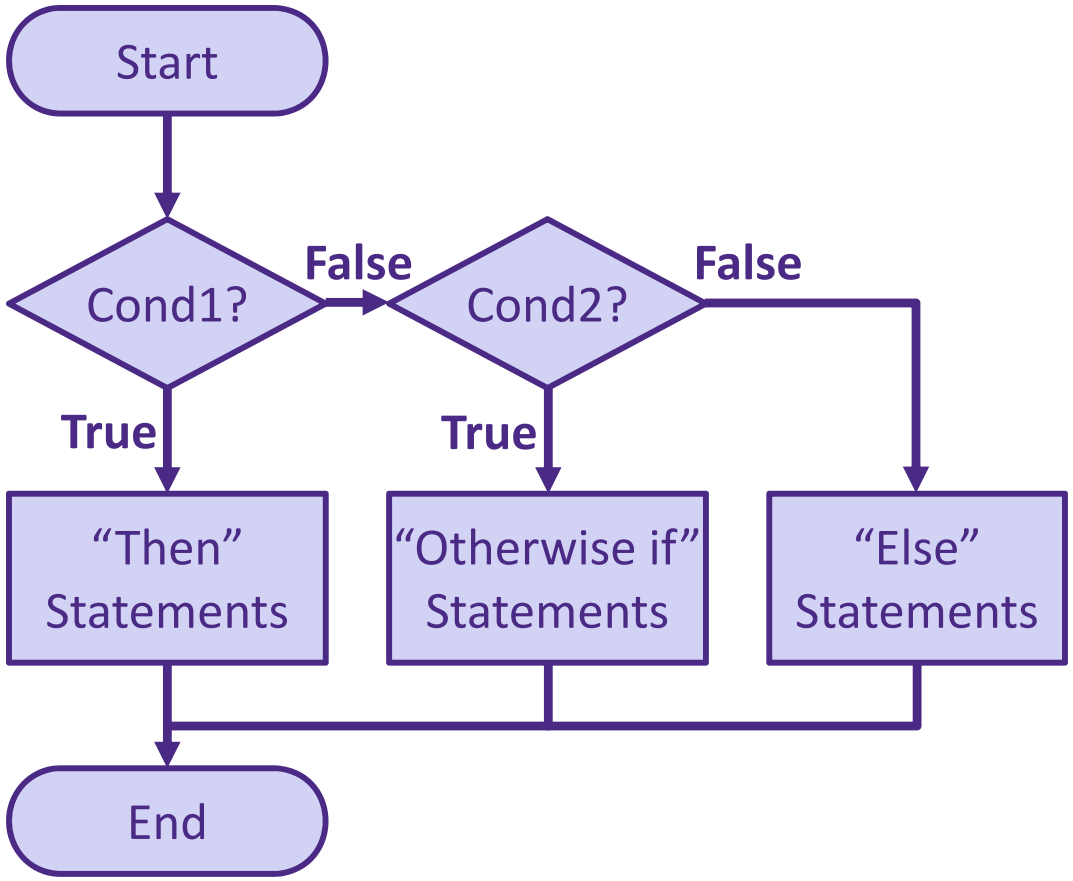
```
if (x == 5) {
    print("Yes");
} else if ((x >= 6) || (x < 2)) {
    print("No");
} else {
    print("Maybe");
}
```


- Think for a minute, then discuss with your neighbor(s)
  - Vote at http://PollEv.com/justinh

Conditionals Worksheet

- Work on **Page 2 (Questions 7-9)**

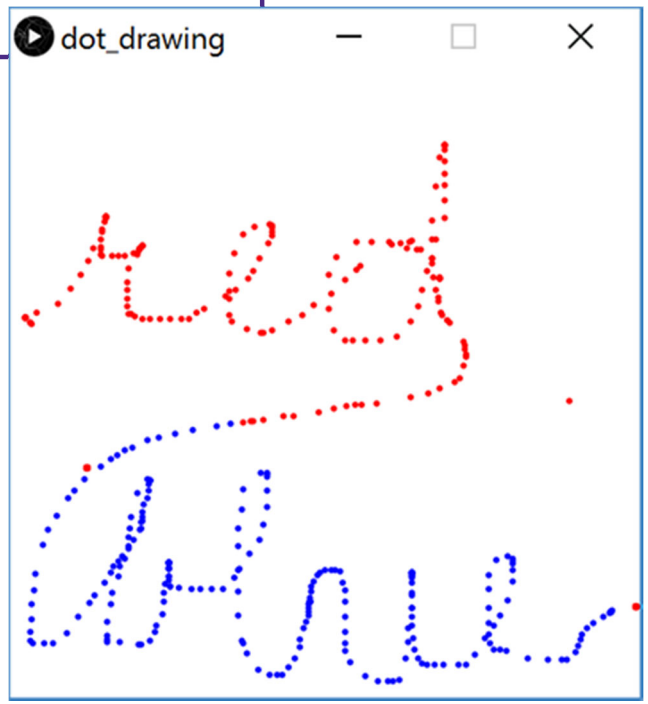
```
if(cond1) {
    // "then"
} else if(cond2) {
    // "otherwise if"
} else {
    // "else"
}
```



Processing Demo: Drawing Dots

true, if mouse is physically being pressed down
false, otherwise

```
14 void draw() {
15   if(mousePressed) {
16     fill(0, 0, 255);    // blue if mouse is pressed
17   } else {
18     fill(255, 0, 0);    // red otherwise
19   }
20   ellipse(mouseX, mouseY, 5, 5);  // draw circle
21 }
```



Jumping Monster

- Using *expressions* and *conditionals* in conjunction with *variables* and *user input* (Wed) to control what is drawn as well as motion: