Quick Assignment Review

Chapter 6- Ruby Basics

Any questions on String formatting or Math Functions?

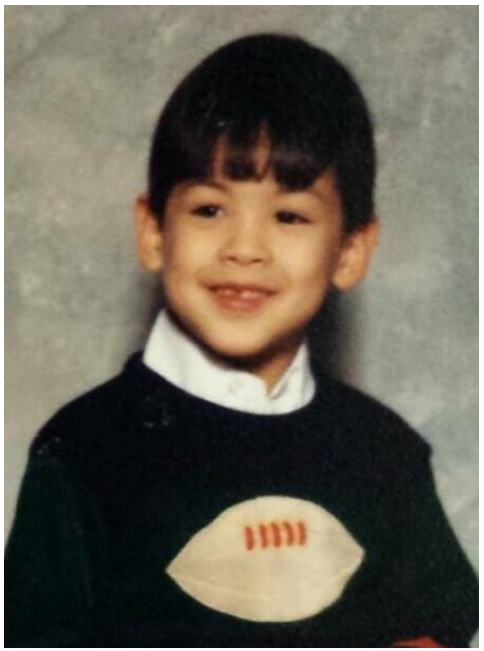
Conditionals

Chapter 7 - Ruby Basics

Logic Paths

Comparisons are used to dictate the flow of a program

*Computers are very good at making very quick decisions, but only if the program outlines all of the possible paths. The computers logic is only as good as the logic of the human writing the program.**



** - This is changing with the introduction of Artificial Intelligence Algorithms.*

Flow Charts

Booleans

a binary variable, having two possible values called "true" and "false."

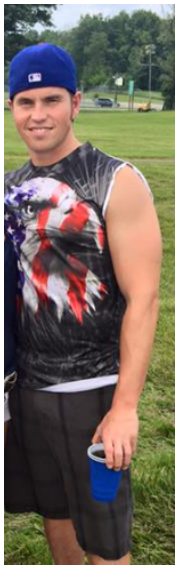
Used to store the value of a comparison

5<6 = true
7<2 = false

Comparison Operators

- == - *used to see if two objects are equal*
- != - *used to see if two objects are not equal*
- < - *used to see if an object is less than another object*
- > - *used to see if an object is more than another object*
- <= - *used for comparison and also equal*
- >= - *used for comparison and also equal*
- ! - *(Logical NOT) used to negate a comparison*
- && - *(Logical AND)used to group multiple comparisons together, all comparisons must be true for the whole statement to be true*
- || - *(Logical OR) used to group multiple comparisons together, only one of the of the comparisons has to be true for the whole statement to be true*

A Comparison of sleeves



<

Comparison Example

```
number_of_no_sleeve_rob = 0
number_of_two_sleeve_rob =2

# can create a boolean
comparison_variable = (number_of_no_sleeve_rob < number_of_two_sleeve_rob)

if comparison_variable
      puts ‘looks good’
else
      puts ‘needs sleeves’
end
```

More complex Comparisons

(5 < 6) && (7 < 8) = ??

(12 > 13) || (14 < 15) = ??

(true && false) || true = ??

!(true) = ??

!((true || false) || (false && true)) = ??

Nested If statements

- Creating a logic tree with multiple conditionals

```
if (1 < my_variable)
    puts 'yeah'
else
    if (2 < my_variable)
        puts 'something'
    else
        if (3 < my variable)
            puts 'crazy'
        end
    end
end
```

elsif

- Another way to do nested ifs

```
if (1 = my_variable)
    puts ‘you only have one?’
elsif (2 = my_variable)
    puts ‘two, really two?’
elsif (3 = my_variable)
    puts ‘you have the magic number’
elsif (4 < my_variable)
    puts ‘so big’
end
```

Stylistic Discussion

Ruby Basics

Indenting

- We use indenting to show grouping of logic code
- This is very important for conditionals and loops

```
if 1 < my_variable
    puts 'yeah'
else
    if 2 < my_variable
        puts 'something'
    else
        if 3 < my variable
            puts 'crazy'
        end
    end
end
```

Next week, remote remote

- Class Trip, Feb 20th

WHAT WHY WHERE WHEN WHO HOW