Pre-Ada Live

Rename yourself in Zoom with

```
Preferred Name
Pronouns
```

Peer code review: Number Guessing Game

Introduce Yourselves!

Pointers for code review:

- Ruby style guide: snake_case, spaces, comments, indentation
- Are all minimal requirements followed and working?
- Are any additional, optional requirements coded and working?
- User experience
    - Are the prompts useful, educational and sufficient for the user of the program who is unaware of the requirements?
    - What if the user enters information in an incorrect format? (case insensitive input, detailed error messages, prompt for re-entry)
- P.S. For easier code reviews, save files with `.rb` extensions in your gists. This allows for better formatting. Saving multiple `.rb` files helps scope the feedback and test code as you make changes

What will get printed on the terminal?

```
x = 1
y = 2
z = 3

if z * x - y > 0
  puts "positive"
else
  puts "zero or negative"
end
```

Reference: boolean expressions

What will get printed on the terminal?

```ruby
x = 1
y = 2
z = 3

if z * x - y > 0
  puts "positive"
else
  puts "zero or negative"
end
```

Reference: boolean expressions

```
CheezItMan@ada:~$ ruby exercise.rb
positive
```

What will get printed on the terminal?

```
x = 1
y = 2
z = 3

if z * (x - y) > 0
  puts "positive"
else
  puts "zero or negative"
end
```

Reference: boolean expressions

What will get printed on the terminal?

```
x = 1
y = 2
z = 3

if z * (x - y) > 0
  puts "positive"
else
  puts "zero or negative"
end
```

Reference: boolean expressions

```
ChrisM@ada:~$ ruby exercise.rb
zero or negative
```

What will get printed on the terminal?

```
x = 2
y = -2
z = 4

if x = 1 || y = -1
  puts "x = 1 or y = -1"
elsif z = 4
  puts "z = 4"
end
```

Reference: boolean expressions

What will get printed on the terminal?

```
x = 2
y = -2
z = 4

if x = 1 || y = -1
  puts "x = 1 or y = -1"
elsif z = 4
  puts "z = 4"
end
```

Reference: boolean expressions

```
chrisM@ada:~$ ruby exercise.rb
exercise.rb:7: warning: found `= literal' in conditional, should be ==
x = 1 or y = -1
```

What will get printed on the terminal?

```
x = 2
y = -2
z = 4

if x == 3 || 4
  puts "x is 3 or 4"
else
  puts "x is not 3 or 4"
end
```

Reference: boolean expressions

What will get printed on the terminal?

```
x = 2
y = -2
z = 4

if x == 3 || 4
  puts "x is 3 or 4"
else
  puts "x is not 3 or 4"
end
```

Reference: boolean expressions

```
ChrisM@ada:~$ ruby exercise.rb
x is 3 or 4
```

Exercise

- If the value of the variable `x` is `2`, the value of the variable `y` is `-2`, and the value of the variable `z` is `4`, write a conditional statement that prints `"it's true"` if `x` is greater than `0` and `y` is less than `0`.
- **Challenge** - come up with a different way to write the conditional statement that behaves the same. Hint: use negation (`!`).

```
x = 2
y = -2
z = 4
```

Exercise

- If the value of the variable `x` is `2`, the value of the variable `y` is `-2`, and the value of the variable `z` is `4`, write a conditional statement that prints `"it's true"` if `x` is greater than `0` and `y` is less than `0`.
- **Challenge** - come up with a different way to write the conditional statement that behaves the same. Hint: use negation (`!`).

```
x = 2
y = -2
z = 4
```

```
# Option 1
if x > 0 && y < 0
  puts "it's true!"
end
```

Exercise

- If the value of the variable `x` is `2`, the value of the variable `y` is `-2`, and the value of the variable `z` is `4`, write a conditional statement that prints `"it's true"` if `x` is greater than `0` and `y` is less than `0`.
- **Challenge** - come up with a different way to write the conditional statement that behaves the same. Hint: use negation (`!`).

```
x = 2
y = -2
z = 4
```

```
# Option 1
if x > 0 && y < 0
  puts "it's true!"
end
```

```
# Option 2
if !(x <= 0 || y >= 0)
  puts "it's true!"
end
```

De Morgan's laws

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
else order == "GRANDE"
  puts "16"
end
```

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
else order == "GRANDE"
  puts "16"
end
```

```
ChrisM@ada:~$ ruby exercise.rb
What size drink would you like? (SHORT or TALL or GRANDE): short
8
```

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
else order == "GRANDE"
  puts "16"
end
```

```
ChrisM@ada:~$ ruby exercise.rb
What size drink would you like? (SHORT or TALL or GRANDE): short
8
```

```
chrisM@ada:~$ ruby exercise.rb
What size drink would you like? (SHORT or TALL or GRANDE): grande
16
```

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
else order == "GRANDE"
  puts "16"
end
```

```
ChrisM@ada:~$ ruby exercise.rb
What size drink would you like? (SHORT or TALL or GRANDE): short
8
```

```
chrisM@ada:~$ ruby exercise.rb
What size drink would you like? (SHORT or TALL or GRANDE): grande
16
```

```
chrisM@ada:~$ ruby exercise.rb
What size drink would you like? (SHORT or TALL or GRANDE): no thank you
16
```

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
else order == "GRANDE"
  puts "16"
end
```

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
else
  order == "GRANDE"
  puts "16"
end
```

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
else
  order == "GRANDE"
  puts "16"
end
```

But this was probably meant to be an `elsif` statement

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
elsif order = "GRANDE"
  puts "16"
else
  puts "unknown size"
end
```

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
elsif order = "GRANDE"
  puts "16"
else
  puts "unknown size"
end
```

```
ChrisM@ada:~$ ruby exercise.rb
exercise.rb:8: warning: found `= literal' in conditional, should be ==
What size drink would you like? (SHORT or TALL or GRANDE): grande
16
```

```
ChrisM@ada:~$ ruby exercise.rb
exercise.rb:8: warning: found `= literal' in conditional, should be ==
What size drink would you like? (SHORT or TALL or GRANDE): no thank you
16
```

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
elsif order = "GRANDE"
  puts "16"
else
  puts "unknown size"
end
```

```
ChrisM@ada:~$ ruby exercise.rb
exercise.rb:8: warning: found `= literal' in conditional, should be ==
What size drink would you like? (SHORT or TALL or GRANDE): grande
16
```

```
ChrisM@ada:~$ ruby exercise.rb
exercise.rb:8: warning: found `= literal' in conditional, should be ==
What size drink would you like? (SHORT or TALL or GRANDE): no thank you
16
```

What does the code below do?

```
print "What size drink would you like? (SHORT or TALL or GRANDE): "
order = gets.chomp.upcase

if order == "SHORT"
  puts "8"
elsif order == "TALL"
  puts "12"
elsif order == "GRANDE"
  puts "16"
else
  puts "unknown size"
end
```

```
ChrisM@ada:~$ ruby exercise.rb
What size drink would you like? (SHORT or TALL or GRANDE): grande
16
```

```
ChrisM@ada:~$ ruby exercise.rb
What size drink would you like? (SHORT or TALL or GRANDE): no thank you
unknown size
```

Exercise 8 discussion: drink sizes

```
# Prompt for a one of the following: SHORT, TALL, GRANDE, VENTI.
# Print out the number of ounces that drink includes (8, 12, 16, 20 respectively).
```

Exercise 8 discussion: drink sizes

```
# Prompt for a one of the following: SHORT, TALL, GRANDE, VENTI.
# Print out the number of ounces that drink includes (8, 12, 16, 20 respectively).
```

```
# Option 1
puts "SHORT, TALL, GRANDE, or VENTI? "
order = gets.chomp.upcase
if order == "SHORT"
  print "8"
elsif order == "TALL"
  print "12"
elsif order == "GRANDE"
  print "16"
elsif order == "VENTI"
  print "20"
else
  print "unknown size"
end
```

Exercise 8 discussion: drink sizes

```
# Prompt for a one of the following: SHORT, TALL, GRANDE, VENTI.
# Print out the number of ounces that drink includes (8, 12, 16, 20 respectively).
```

```
# Option 1
puts "SHORT, TALL, GRANDE, or VENTI? "
order = gets.chomp.upcase
if order == "SHORT"
  print "8"
elsif order == "TALL"
  print "12"
elsif order == "GRANDE"
  print "16"
elsif order == "VENTI"
  print "20"
else
  print "unknown size"
end
```

```
# Option 2
puts "SHORT, TALL, GRANDE, or VENTI? "
order = gets.chomp.upcase
if order == "SHORT"
  print "8"
end
if order == "TALL"
  print "12"
end
if order == "GRANDE"
  print "16"
end
if order == "VENTI"
  print "20"
end
```

Exercise 8 discussion: drink sizes

```
# Prompt for a one of the following: SHORT, TALL, GRANDE, VENTI.
# Print out the number of ounces that drink includes (8, 12, 16, 20 respectively).
```

```
# Option 1
puts "SHORT, TALL, GRANDE, or VENTI? "
order = gets.chomp.upcase
if order == "SHORT"
  print "8"
elsif order == "TALL"
  print "12"
elsif order == "GRANDE"
  print "16"
elsif order == "VENTI"
  print "20"
else
  print "unknown size"
end
```

```
# Option 2
puts "SHORT, TALL, GRANDE, or VENTI? "
order = gets.chomp.upcase
if order == "SHORT"
  print "8"
end
if order == "TALL"
  print "12"
end
if order == "GRANDE"
  print "16"
end
if order == "VENTI"
  print "20"
end
```

Why is option 1 better than option 2?

Discussion - user input retries

Which option should we choose?

```ruby
# Option 1
puts "How much money do you have? "
money = gets.chomp
if money.to_i.to_s != money
  puts "Not a number, try again"
  exit # exit program
else
  money = money.to_f
  # ... remaining program logic
end
```

```ruby
# Option 1
puts "How much money do you have? "
money = gets.chomp
if money.to_i.to_s != money
  puts "Not a number, try again"
  exit # exit program
else
  money = money.to_f
  # ... remaining program logic
end
```

```ruby
# Option 2
puts "How much money do you have? "
money = gets.chomp
while money.to_i.to_s != money
  puts "Not a number, try again"
  puts "How much money do you have? "
  money = gets.chomp
end
money = money.to_f
# ... remaining program logic
```

```ruby
# Option 1
puts "How much money do you have? "
money = gets.chomp
if money.to_i.to_s != money
  puts "Not a number, try again"
  exit # exit program
else
  money = money.to_f
  # ... remaining program logic
end
```

```ruby
# Option 2
puts "How much money do you have? "
money = gets.chomp
while money.to_i.to_s != money
  puts "Not a number, try again"
  puts "How much money do you have? "
  money = gets.chomp
end
money = money.to_f
# ... remaining program logic
```

```ruby
# Option 3
puts "How much money do you have? "
money = gets.chomp
attempt = 0
while money.to_i.to_s != money
  if attempt < 3
    puts "Not a number, try again"
    puts "How much money do you have? "
    money = gets.chomp
    attempt += 1
  else
    puts "Not a number, exiting"
    exit # no more chances
  end
end
money = money.to_f
puts "money: $ #{money}"
# ... remaining program logic
```

What is this line doing?

```
while money.to_i.to_s != money
  # ... prompt user again to enter a value
end
```

What is this line doing?

```
while money.to_i.to_s != money
  # ... prompt user again to enter a value
end
```

```
ChrisM@ada:~$ irb
irb(main):001:0> x = gets.chomp
50  # <-- enter in a number
irb(main):002:0> x
=> "50"
irb(main):003:0> x.to_i
=> 50
irb(main):004:0> x.to_i.to_s
=> "50"
irb(main):005:0> x.to_i.to_s != x
=> false
```

What is this line doing?

```ruby
while money.to_i.to_s != money
  # ... prompt user again to enter a value
end
```

```
ChrisM@ada:~$ irb
irb(main):001:0> x = gets.chomp
50  # <-- enter in a number
irb(main):002:0> x
=> "50"
irb(main):003:0> x.to_i
=> 50
irb(main):004:0> x.to_i.to_s
=> "50"
irb(main):005:0> x.to_i.to_s != x
=> false
```

```
ChrisM@ada:~$ irb
irb(main):001:0> x = gets.chomp
help  # <-- enter a word
irb(main):002:0> x
=> "help"
irb(main):003:0> x.to_i
=> 0
irb(main):004:0> x.to_i.to_s
=> "0"
irb(main):005:0> x.to_i.to_s != x
=> true
```

Debrief

- What new things did you learn today?
- What topics are you still struggling with?
- What did you especially enjoy about today's class?
- What can be improved for future classes?

Give us feedback

Give us feedback