Introduction

In this video, we're going to use arrays and loops to model a deck of playing cards.

Create a file called called *cards.rb*:

```
deck = []
suits = %w[ Hearts Diamonds Clubs Spades ]
values = %w[ Ace 2 3 4 5 6 7 8 9 10 Jack Queen King ]
suits.each do |suit|
  values.each do |value|
    deck << "#{value} of #{suit}"
  end
end
deck.shuffle!
puts deck.join(",")
```

It starts by setting up an empty array and assigning it to a variable called `deck`. This will actually represent our finished deck of cards, it will be an array, but we have to start off by initializing this array, because we'll be referring to it later, so it has to exist.

After this, we're going to use the fact that a pack of cards is made up of four suits, and each suit has the same cards in it, from ace up to jack, queen, and king. So with this in mind, we've created two arrays, one to represent the different suits, that you can see here, and the other to represent all the different values on the cards. The shorthand notations for creating arrays are used, so we don't need to separate items by commas.

Nested Loop

Next we loop through both of these arrays by using a nested loop:

```
suits.each do |suit|
  values.each do |value|
    deck << "#{value} of #{suit}"
  end
end
```

Inner Loop

The way nested loops work is that for each value in the outer loop we go through every single value in the inner loop before iterating on to the next value in suits. We gradually push different strings into `deck`. After the loops have finished its job we should have populated `deck` with every single value in a pack of cards.

Join Method

Next we use `join` to convert array to string and display it.

Lastly, we usually would want the pack to be shuffled, and it just so happens that Ruby has a very useful array method called `shuffle`.

Shuffle

By calling

```
deck.shuffle!
```

we shuffle the deck of cards. By employing the bang method we change the array and keep it changed.