Single quoted strings are interpreted literally.

Double quoted strings allows further processing on the string, like the substitution of variables into `${VARIABLE}`.

```
puts some_string
```

```
obj.class
obj.methods
```

Block form:

```
if condition
   #statements
end
```

One-line form:

```
statement if condition
```

They are more idiomatic Ruby for "if not" and "while not" constructions.

Every type can be interpreted as a boolean. `nil` and and the literal `false` evaluate to `false`. Everything else evaluates to `true`.

`&&`/`||` short-circuit, as in Java. Also, `&&`/`||` have the alternatives `and`/`or`.

Type checking isn't done until run-time, when an operation is actually executed.

Ruby is duck-typed, meaning one type can be used as another so long as it has the needed fields and methods to make an operation work. This is to be contrasted with languages like Java where one type can be used as another only if one is a subclass of the other.

The last value computed is the result of the function.
This is also true of other kinds of blocks, not just the bodies of functions.

... optional. Whether in the definition of a function or a call.

Arrays.

They use square brackets for definition and indexing, two dots for slicing.

```
nums = ['one', 'two', 'three']
nums[0]     #'one'
nums[-1]    #'three'
```

Use `Range` objects like `0..3`.

```
someArray[n1..n2]
```

Ruby returns `nil`, unlike Java or Python.

```
el.include?(arr)
```

There is no equivalent to Python's `in`.

```
captitals = {'France' => 'Paris', 'USA' => 'Washington D.C.'}
capitals['France'] #'Paris'
```

An idntifier that follows a colon. They are interned, guaranteeing that two uses of the symbol `:symb` actually refer to the same object in memory.

```
obj.object_id
```

- Concise, elegant syntax.
- Flexible polymorphism from duck typing.
- Consistent object-orientation.
- Short time to market.

- Scripting, especially to glue programs together.
- Web development with Ruby on Rails.

- Poor performance.
- Weak concurrency due to mutable state and the global interpreter lock.
- Dynamic & duck typing reduces safety and rules out many useful development tools.