Score with a native class

The Score class provides you a structure for keeping track of the score.
Create a score.lua file, next to your main.lua file:

```
local Score = {}
function Score:create()
   local score = {}

   score.value = 0

   function score:show()
      print("Score: " .. tostring(self.score))
   end

   function score:increment(value)
      self.value = self.value + 1
   end

   function score:set(value)
      self.value = value
   end

   return score
end

Return Score
```


Score with a native class

The Score class provides you a structure for keeping track of the score.
Create a score.lua file, next to your main.lua file:

```
local Score = {}
function Score:create()
   local score = {}

   score.value = 0

   function score:show()
      print("Score: " .. tostring(self.score))
   end

   function score:increment(value)
      self.value = self.value + 1
   end

   function score:set(value)
      self.value = value
   end

   return score
end

Return Score
```

Score with a native class

The Score class provides you a structure for keeping track of the score.
Create a score.lua file, next to your main.lua file:

```
local Score = {}
function Score:create()
   local score = {}

   score.value = 0

   function score:show()
      print("Score: " .. tostring(self.score))
   end

   function score:increment(value)
      self.value = self.value + 1
   end

   function score:set(value)
      self.value = value
   end

   return score
end

Return Score
```

Score with a native class

The Score class provides you a structure for keeping track of the score.
Create a score.lua file, next to your main.lua file:

```
local Score = {}
function Score:create()
   local score = {}

   score.value = 0

   function score:show()
      print("Score: " .. tostring(self.score))
   end

   function score:increment(value)
      self.value = self.value + 1
   end

   function score:set(value)
      self.value = value
   end

   return score
end

Return Score
```

Using the Score class

You can now use the Score class in your main.lua:

```
local Score = require "score"

playerA = Score:create()
playerA:show()

playerB = Score:create()
playerB:show()
playerB:increment()
player:show()

score:show()
```

Before using it, we have to first include the Score class from the score.lua file: most of the time you will name the class with the same name as defined in the module, which is probably also the name of the file, capitalized, without the .lua extension.. But it's not mandatory.

When you run your program you will get the output:

```
1 -- the score for player A
1 -- the initial score for player B
2 -- the incremented score for player B
1 -- the unmodified score for player A
```

PlayerA and PlayerB have different scores but share the implementation of the show() and increment() functions.

Using the Score class

You can now use the Score class in your main.lua:

```
local Score = require "score"

playerA = Score:create()
playerA:show()

playerB = Score:create()
playerB:show()
playerB:increment()
player:show()

score:show()
```

Before using it, we have to first include the Score class from the score.lua file: most of the time you will name the class with the same name as defined in the module, which is probably also the name of the file, capitalized, without the .lua extension.. But it's not mandatory.

When you run your program you will get the output:

```
1 -- the score for player A
1 -- the initial score for player B
2 -- the incremented score for player B
1 -- the unmodified score for player A
```

PlayerA and PlayerB have different scores but share the implementation of the show() and increment() functions.

Using the Score class

You can now use the Score class in your main.lua:

```
local Score = require "score"

playerA = Score:create()
playerA:show()

playerB = Score:create()
playerB:show()
playerB:increment()
player:show()

score:show()
```

Before using it, we have to first include the Score class from the score.lua file: most of the time you will name the class with the same name as defined in the module, which is probably also the name of the file, capitalized, without the .lua extension.. But it's not mandatory.

When you run your program you will get the output:

```
1 -- the score for player A
1 -- the initial score for player B
2 -- the incremented score for player B
1 -- the unmodified score for player A
```

PlayerA and PlayerB have different scores but share the implementation of the show() and increment() functions.