Exercises - Session 20

In case you get stuck anywhere, don't be afraid to ask the coaches! They are here to help and will gladly explain everything to you!
Take notes during the exercises. Even if you never look at them again, they will help you memorise things!

The Todo list App - Part 2

This exercise builds on the exercise of session 19. It's highly recommended that you first complete part 1 before you start with this exercise. As always, if you need help, please ask one of your coaches.

Now that we have the foundation of our todo list app -- namely our todos -- we can start meaningfully manage them. Our goal is to have a command line app that manages todos. The result of this exercise could look like this:

```
$ ruby todo_manager.rb
> list
Here's all your todos:
> add
What todo do I have to add? Finish session 20
> list
Here's all your todos:
☐ Finish session 20 (0)
> done 0
> list
Here's all your todos:
▯ Finish session 20 (0)
> quit
Goodbye!
$
```

Before you start with any programming, read the following paragraph which explains what's actually happening in the snippet above.

You first start your script called `todo_manager.rb`. And after that, a prompt (" >") waits for the first command to be entered. In this case, it's `list`. After you type `list` and hit enter, all the todos are listed. Since we don't have any todos, this list is empty. We're again shown a prompt where we type the command `add`. After that, we're asked to give the todo a title. We enter a title and hit enter. After that, our todo is added to the todo list. We're again shown the prompt where we type `list` to verify that our todo was indeed added to the list. After that, we decide to tick this todo off by typing `done 0`. The zero after done stays for the index. To verify that our todo is indeed completed, we `list` again all todos before we quit the application.

So let's build this app.

Step 1: Showing a command prompt that can be quit

Your goal is to show a command prompt that can be exited with the command quit. Every other command should just show the command prompt again waiting for the next command. If you want to be polite, you could print “I could not understand your command, please try again”.

Step 2: Listing the todos

Since we now have a command prompt that can be quit, it’s time to add the second command: list. If the command list is chosen, iterate over all your todos and print them one by one to the command line. After that, the user should be able to enter the next command (e.g. quit).

Step 3: Adding a todo

The next command we implement is add. If the user chooses add, we follow up with the question about the title. With the title the user gives us, we are able to create a todo object that is then added to our todo collection. To verify that our add command works, we should see the newly created todo with list.

Step 4: Completing a todo

Now that we added the todo, we want to be able to complete the todo. Assuming we only added 1 todo so far, we should be able to tick it off with `done 0`. The zero stays for the index. After that, we are able to verify that the todo is indeed done by typing `list`.

And that’s it.

Optional part

Step 5: Only show undone todos

By typing `undone`, only show the todos that are undone and hide the complete ones. How can you solve that?