Exercises - Session 4

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!

Todo list

We're building a Todo list program in the command line! Before we can start, we should prepare ourselves. First, create a folder for your new program. It could be called "todolist". In that folder, create a Ruby file called "todolist.rb". We will program in that file for the rest of the exercise. We'll do it step by step, so that we have a working program after each step.

1. Say hi to the user and ask for a command to enter. If the user enters "quit", you should print "Goodbye!". Otherwise, the command is ignored and your program ends as usual.

Example output (user entered text is **bold**):

```
> ruby todolist.rb
Welcome to the todo list
Please enter your command!
something
> ruby todolist.rb
Welcome to the todo list
Please enter your command!
quit
Goodbye!
```

2. Now, allow the user to stay in the script. After the user entered her command, she should be able to type other commands. If she writes "quit", the program quits, otherwise she can enter commands indefinitely.

Example output (user entered text is **bold**):

```
> ruby todolist.rb
Welcome to the todo list
Please enter your command!
something
Please enter your command!
test
Please enter your command!
quit
Goodbye!
```

3. Now we want to be able to list todos. First, we prepare a `todos` variable and fill it with some todo examples (e.g. “Buy milk”, “Finish rubymonstas exercise”). The user should be able to list all todos with the command “list”. Our script now supports two commands: list and quit. It will still ask for new commands if the command was not “quit”.

Example output (user entered text is **bold**):

```
> ruby todolist.rb
Welcome to the todo list
Please enter your command!
list
Buy milk
Finish rubymonstas exercise
Please enter your command!
quit
Goodbye!
```

4. The command we implement next is “add”. After the user typed add, we will ask for a short text that is the new todo item. After the text was entered, we add this to the list of todos. After that, we can enter a new command.

Example output (user entered text is **bold**):

```
> ruby todolist.rb
Welcome to the todo list
Please enter your command!
add
What is your todo?
drive home
Please enter your command!
list
Buy milk
Finish rubymonstas exercise
drive home
Please enter your command!
quit
Goodbye!
```

5. The last command we implement is “done”. If the user is done with an item, she can type “done” and then enter one of the todo texts. This todo item should then be removed from the todo list. After that, a new command can be entered as usual.

Example output (user entered text is **bold**):

```
> ruby todolist.rb
Welcome to the todo list
Please enter your command!
```

```
list
Buy milk
Finish rubymonstas exercise
Please enter your command!
done
What todo is done?
Finish rubymonstas exercise
Please enter your command!
list
Buy milk
Please enter your command!
quit
Goodbye!
```