



- ▶ Introduction and overview
- ▶ Basic types, definitions and functions
- ▶ Basic data structures
- ▶ More advanced data structures
- ▶ Higher order functions
- ▼ Exceptions, input/output and imperative constructs

Table of Contents

Imperative features in OCaml

Getting and handling your Exceptions

Week 5 Échéance le déc 12, 2016 at 23:30 UTC

Getting information in and out

Week 5 Échéance le déc 12, 2016 at 23:30 UTC

Sequences and iterations

Week 5 Échéance le déc 12, 2016 at 23:30 UTC

Mutable arrays

Week 5 Échéance le déc 12, 2016 at 23:30 UTC

Mutable record fields

Week 5 Échéance le déc 12, 2016 at 23:30 UTC

Variables, aka References

Week 5 Échéance le déc 12, 2016 at 23:30 UTC

- ▶ Modules and data abstraction
- ▶ Project

READING LINES FROM THE STANDARD INPUT (10/10 points)

The code given in the template is an attempt to reading a list of lines from the standard input, slightly different from the one shown in the course notes.
At first sight, it behaves well.

```
# read_lines ();;  
Hello  
mister  
the  
caml!  
- : string list = [ "Hello" ; "mister" ; "the" ; "caml!" ]
```

But if you call this function several times, you get unexpected results.

1. Study the code, to understand what is going on, compare with the example shown in the course, and then fix this code.

Important note: it is not possible to implement reading functions that actually ask the user to input something in the current toplevel environment.

For that, the environment defines an alternative version of `read_line` that simulates user interaction. Some calls will return a string, some others will (less often) raise `End_of_file`.

YOUR OCAML ENVIRONMENT

```
1 let read_lines () =  
2   let sl = ref [] in  
3   let rec aux () =  
4     try  
5       sl := read_line () :: !sl ;  
6       aux ()  
7     with  
8       End_of_file -> () in  
9   aux ();  
10  List.rev !sl;;  
11
```

Evaluate >

Switch >>

Typechecked

Reset Templ

Full-screen |

Check & Sa

Exercise complete (click for details)

10 pts

Completed, 10 pts

v Exercise 1: read_lines

Found read_lines with compatible type.

Computing read_lines ()

This time, the input sequence is:

```
Never gonna give you up.  
Never gonna make you cry.  
Never gonna let you down
```

Correct value

```
["Never gonna give you up."; "Never gonna make you cry.";  
 "Never gonna let you down"]
```

Computing read_lines ()

This time, the input sequence is:

```
Never gonna make you cry.  
Never gonna let you down  
Never gonna tell a lie and hurt you.
```

Correct value

```
["Never gonna make you cry."; "Never gonna let you down";  
 "Never gonna tell a lie and hurt you."]
```

1 pt

1 pt

```
Never gonna give you up.  
Never gonna say goodbye.  
Never gonna let you down  
Correct value  
["Never gonna make you cry."; "Never gonna give you up.";  
"Never gonna say goodbye."; "Never gonna let you down"]  
Computing read_lines ()  
This time, the input sequence is:  
Never gonna say goodbye.  
Never gonna let you down  
Never gonna run around and desert you.  
Correct value  
["Never gonna say goodbye."; "Never gonna let you down";  
"Never gonna run around and desert you."]  
Computing read_lines ()  
This time, the input sequence is:  
Never gonna give you up.  
Never gonna run around and desert you.  
Never gonna say goodbye.  
Never gonna give you up.  
Never gonna tell a lie and hurt you.  
Never gonna run around and desert you.  
Correct value  
["Never gonna give you up."; "Never gonna run around and desert you.";  
"Never gonna say goodbye."; "Never gonna give you up.";  
"Never gonna tell a lie and hurt you.";  
"Never gonna run around and desert you."]  
Computing read_lines ()  
This time, the input sequence is:  
Correct value []  
Computing read_lines ()  
This time, the input sequence is:  
Never gonna tell a lie and hurt you.  
Never gonna let you down  
Never gonna run around and desert you.  
Never gonna give you up.  
Never gonna say goodbye.  
Correct value  
["Never gonna tell a lie and hurt you."; "Never gonna let you down";  
"Never gonna run around and desert you."; "Never gonna give you up.";  
"Never gonna say goodbye."]  
Computing read_lines ()  
This time, the input sequence is:  
Never gonna let you down  
Never gonna make you cry.  
Correct value ["Never gonna let you down"; "Never gonna make you cry."]  
Computing read_lines ()  
This time, the input sequence is:  
Never gonna give you up.  
Never gonna run around and desert you.  
Never gonna run around and desert you.  
Correct value  
["Never gonna give you up."; "Never gonna run around and desert you.";  
"Never gonna run around and desert you."]  
Computing read_lines ()  
This time, the input sequence is:  
Never gonna make you cry.  
Never gonna tell a lie and hurt you.  
Correct value ["Never gonna make you cry."; "Never gonna tell a lie and hurt you."] 1 pt
```



Rechercher un cours



OPENedX