



► Introduction and overview

► Basic types, definitions and functions

▼ Basic data structures

Table of Contents

Greetings

User-defined types

Week 2 Échéance le déc 12, 2016 at 23:30 UTC



Tuples

Week 2 Échéance le déc 12, 2016 at 23:30 UTC



Records

Week 2 Échéance le déc 12, 2016 at 23:30 UTC



Arrays

Week 2 Échéance le déc 12, 2016 at 23:30 UTC



Case study: A small typed database

Week 2 Échéance le déc 12, 2016 at 23:30 UTC



► More advanced data structures

► Higher order functions

► Exceptions, input/output and imperative constructs

► Modules and data abstraction

## ENIGMA (30/30 points)

Let us solve the following puzzle:

If you multiply my grand-son age by four, you know how old I am. Now, if you exchange the two digits of our ages then you have to multiply by three my age to get the age of my grand-son!

1. Write a function `exchange` of type `int -> int` that takes an integer `x` between 10 and 99 and returns an integer which is `x` whose digits have been exchanged. For instance, `exchange 73 = 37`.
2. Define `is_valid_answer` of type `int * int -> bool` such that `is_valid_answer (grand_father_age, grand_son_age)` returns `true` if and only if `grand_father_age` and `grand_son_age` verify the constraints of the puzzle.
3. Write a function `find : (int * int) -> (int * int)` that takes a pair `(max_grand_father_age, min_grand_son_age)` and returns a solution `(grand_father_age, grand_son_age)` to the problem, where `min_grand_son_age <= grand_son_age < grand_father_age <= max_grand_father_age` or `(-1, -1)` if there was no valid answer in the given range.

## YOUR OCAML ENVIRONMENT

```
1 let exchange k =
2   (k mod 10) * 10 + (k / 10) ;;
3
4 let is_valid_answer (grand_father_age, grand_son_age) =
5   grand_son_age * 4 = grand_father_age
6   &&
7   (exchange grand_father_age) * 3 = exchange grand_son_age ;;
8
9
10 let rec find answer =
11   let (grand_father_age, grand_son_age) = answer in
12   if grand_son_age >= grand_father_age then (-1,-1) else
13   if find_down answer <> (-1,-1) then find_down answer else
14     find (grand_father_age, grand_son_age + 1)
15 and
16   find_down answer =
17     let (grand_father_age, grand_son_age) = answer in
18     if grand_son_age >= grand_father_age then (-1,-1) else
19     if is_valid_answer answer = true then answer else
20       find_down (grand_father_age - 1, grand_son_age);;
21
22 |
```

Evaluate >

Switch >>

Typecheck

Reset Templ

Full-screen |

Check & Sa

### Exercise complete (click for details)

30 pts

Completed, 10 pts

#### ▼ Exercise 1: exchange

Found exchange with compatible type.

Computing exchange 27

Correct value 72

1 pt

Computing exchange 98

Correct value 89

1 pt

Computing exchange 66

Correct value 66

1 pt

Computing exchange 13

Correct value 31

1 pt

Computing exchange 54

Correct value 45

1 pt

Computing exchange 67

Correct value 76

1 pt

Computing exchange 11

Correct value 11

1 pt

Computing exchange 78

Correct value 87

1 pt

Correct value 11	1 pt
✓ Exercise 2: is_valid_answer	Completed, 10 pts
Found is_valid_answer with compatible type.	
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value true	1 pt
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value false	1 pt
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value false	1 pt
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value false	1 pt
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value false	1 pt
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value false	1 pt
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value false	1 pt
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value false	1 pt
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value false	1 pt
Computing [REDACTED FOR YOUR OWN GOOD]	
Correct value false	1 pt
✓ Exercise 3: find	Completed, 10 pts
Found find with compatible type.	
Computing find (99, 10)	
Correct value (72, 18)	1 pt
Computing find (99, 11)	
Correct value (72, 18)	1 pt
Computing find (87, 10)	
Correct value (72, 18)	1 pt
Computing find (92, 10)	
Correct value (72, 18)	1 pt
Computing find (76, 23)	
Correct value (-1, -1)	1 pt
Computing find (76, 24)	
Correct value (-1, -1)	1 pt
Computing find (80, 23)	
Correct value (-1, -1)	1 pt
Computing find (96, 30)	
Correct value (-1, -1)	1 pt
Computing find (83, 10)	
Correct value (72, 18)	1 pt
Computing find (76, 24)	
Correct value (-1, -1)	1 pt



Rechercher un cours



OPENedX