LAMBDA-EXPRESSIONS AS VALUES (20/20 points)

In this exercise, we will define functions as values, also called *lambda expressions*, using the keyword `function` to do a pattern-matching on their argument. As a result, you are not allowed to use `match ... with` in your code.

1. Define a function `last_element: 'a list -> 'a` that returns the last element of a list. Your function may return `(invalid_arg "last_element")` when the list is empty.
2. Write a function `is_sorted: 'a list -> bool` that takes a list `l` as argument, and that checks that the list is sorted in increasing order according to the polymorphic comparison operator `<`.

YOUR OCAML ENVIRONMENT

```
1  let rec last_element = function
2      [] -> (invalid_arg "last_element")
3    | [e] -> e
4    | x :: xs -> last_element xs
5  ;;
6
7  let rec is_sorted = function
8      [] -> true
9    | [_] -> true
10   | x :: (y :: xs) -> if x < y then true && is_sorted (y :: xs) else false
11 ;;
12
```

Evaluate >

Switch >>

Typecheck

Reset Templ

Full-screen

Check & Sa

Exercise complete (click for details) 20 pts

v Exercise 1: `last_element` Completed, 10 pts

Found a toplevel definition for `last_element` .
Found `last_element` with compatible type.
Computing `last_element [-5; 3; -5; -3; 4; -3; -3; 2; 0; 3]`
Correct value `3` 1 pt
Computing `last_element [2; -1; 0; 3; -5; 4; 2; -5; -2; -2]`
Correct value `-2` 1 pt
Computing `last_element [2; -3; -1; -2]`
Correct value `-2` 1 pt
Computing `last_element [-4; -3; -4; 2; 1; 3; 2; 3]`
Correct value `3` 1 pt
Computing `last_element [3; 1; 0]`
Correct value `0` 1 pt
Found `last_element` with compatible type.
Computing `last_element []`
Correct exception `Invalid_argument(last_element)` 1 pt
Computing
```
last_element
  ["//OCaml"; "OCaml4456"; "OCP// 4456OCamlbebe OCaml"; "";
   "OCamlOCP#ba- OCP, 4456"; "//OCaml// OCaml"; " "; "beOCPba4456OCaml, ba"]
```
Correct value `"beOCPba4456OCaml, ba"` 1 pt
Computing
```
last_element
  ["4456, "; "-- "; "4456baOCP-OCP, ba4456ba"; "baba4456";
   " be#OCamlOCamlOCaml"]
```
Correct value `" be#OCamlOCamlOCaml"` 1 pt
Computing `last_element [", OCaml, ba , , baba"]`

```
    "OCaml, OCP"]
```
Correct value "OCaml, OCP" 1 pt

v Exercise 2: `is_sorted` Completed, 10 pts

Found a toplevel definition for `is_sorted` .

Found `is_sorted` with compatible type.

Computing `is_sorted [-4; -2; 1; 4; 4]`

Correct value `false` 1 pt

Computing `is_sorted [4; -2; -1; 1; 3]`

Correct value `false` 1 pt

Computing `is_sorted [3; 2; 1; -1; 4]`

Correct value `false` 1 pt

Computing `is_sorted [4; 2; 1; -4; -1]`

Correct value `false` 1 pt

Computing `is_sorted [-5; -1; 0; 2; 3]`

Correct value `true` 1 pt

Found `is_sorted` with compatible type.

```
Computing
  is_sorted
    [-3.12348878516788186; -2.88804196894348264; -2.79876276612591379;
     2.7623274126117261; 1.87974652549282162]
```
Correct value `false` 1 pt

```
Computing
  is_sorted
    [-2.24170697456471313; -1.75009844955452376; 2.20649214561690776;
     2.35053659900690626; 4.35954193801873657]
```
Correct value `true` 1 pt

```
Computing
  is_sorted
    [-0.637371607496442394; -3.09184834766956662; 3.34626980199034207;
     0.154549405143090901; 1.58700884239558704]
```
Correct value `false` 1 pt

```
Computing
  is_sorted
    [4.92500836591445434; 0.118358912624385582; -3.19989771807083301;
     0.310781542545805145; 0.597139301284993707]
```
Correct value `false` 1 pt

```
Computing
  is_sorted
    [-4.54184766387093575; -4.20873639156226798; -2.877331835019457;
     -1.39276252234172304; 0.968644227576296757]
```
Correct value `true` 1 pt