FPV Tutorübung

Woche 5

Ocaml

Manuel Lerchner

22.05.2023

T01: Expressions

So far, you learned about the following types of expressions:

- Constants
- Variables
- Unary operators
- Binary operators
- Tuples
- Records
- Lists
- If-then-else
- Pattern matching
- Function definition
- Function application
- Variable binding

1. For each of the aforementioned types of expressions, give the general structure and two concrete examples with different subexpressions.

T01: Expressions

- Constants:
- Variables:
- Unary Operator:
- Binary Operator:
- Tuples:

T01: Expressions

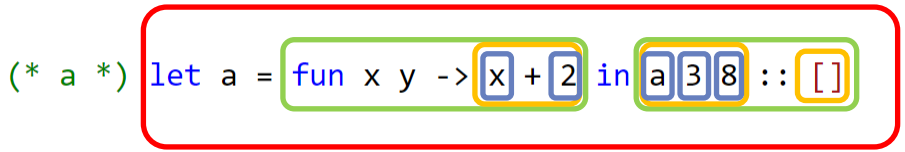
- Records (definition):
- Records (access):
- Lists:
- if-then-else:

T01: Expressions

- Pattern Matching:
- Function Definition :
- Function Application :
- Variable Binding:

T01: Expressions

2. For the following expressions, list all contained subexpressions and give their corresponding types. Then evaluate the expressions:




```
(* a *) let a = fun x y -> x + 2 in a 3 8 :: []

(* b *) ((fun x -> x::[]) (9 - 5), true, ('a', 7))
```

T01: Expressions

```
(* a *) let a = fun x y -> x + 2 in a 3 8 :: []
```

T01: Expressions

```
(* b *) ((fun x -> x::[]) (9 - 5), true, ('a', 7))
```

T02: What‘s the Point

Using what you learned about tuple types in the lecture, implement functionality for computing with three-dimensional vectors.

1. ⊗ **Define a suitable data type for your point.** 0 of 1 tests passing
   The type `vector3` should be a tuple of 3 float values.
2. ? **Define three points** 0 of 1 tests passing
   The points `p1`, `p2` and `p3` should all be different, but their exact values don't matter. Use them, along with other points, to test your functions.
3. ? **string_of_vector3** 0 of 1 tests passing
   Implement a function `string_of_vector3 : vector3 -> string` to convert a vector into a human-readable representation.
   For example, the string for the zero vector should be: `(0.,0.,0.)`.
   Hint: use `string_of_float` to convert components.
4. ? **vector3_add** 0 of 1 tests passing
   Write a function `vector3_add : vector3 -> vector3 -> vector3` that adds two vectors component-wise.
5. ? **vector3_max** 0 of 1 tests passing
   Write a function `vector3_max : vector3 -> vector3 -> vector3` that returns the larger argument vector (the vector with the greater magnitude).
6. ? **combine** 0 of 1 tests passing
   Write a function `combine : vector3 -> vector3 -> vector3 -> string` that adds its first argument to the larger of the other two arguments and returns the result as a string.

T03: Student Database

In this assignment, you have to manage the students of a university.

1. **Type** No results
   First you need to define some types.
   - Define a data type for a `student`.
     A student should be represented as a record of the students `first_name`, `last_name`, identification number `id`, number of the current `semester` as well as the list of `grades` received in different courses.
     The grades should be a pair of the course number and the grade value, a floating point number.
   - To actually manage student you need a `database` which shall be represented as a list of students.
2. **insert** No results
   Write a function `insert : student -> database -> database` that inserts a student into the database.
3. **find_by_id** No results
   Write a function `find_by_id : int -> database -> student list` that returns a list with the (first) student with the given id (either a single student or an empty list, if no such student exists).
4. **find_by_last_name** No results
   Implement a function `find_by_last_name : string -> database -> student list` to find all students with a given last name.