MiniML Extension

Language was extended with support for *lexical environment*, based on suggested idea 5.1 (2).

Lexical scoping was explained very well in lecture 13. Following example:

```
let x = 1 in let f = fun y -> x + y in let x = 2 in f 3 ;;
```

should be evaluated to 4, but in dynamic environment there is no support for closures or variable binding, thus it is evaluated to 5 with a respect to last x declaration. So a support for lexical environment was added as extension for MiniML language.

Doing that required implementing Closure which was given as a possible value type for Environment. Such update required changes in `let`, `let rec`, `fun` and `app` constructs compare to dynamic environment.

In addition, a division binary operation was added to the language for all evaluate models.

Testing framework was implemented based on lecture notes. It was used to test evaluation for substitution mode.

TODO:

1. ~~Implement `Letrec` for substitution model doesn't work.~~
2. ~~Implement `Letrec` for lexical environment.~~
3. ~~Add unit testing framework and test evaluation for substitution, dynamic and lexical environments.~~
4. (Optional) do not exit when exception is raised.

Some commonly used examples for testing:

1. `let x = 1 in let f = fun y -> x + y in let x = 2 in f 3 ;;` Should return `Num(5)` in dynamic environment and 4 for lexical environment.
2. `let rec f = (fun x -> x +1) in f (f (f 5)) ;;` Should return `Num(8)` for all evaluations