Programming language theory, 2021 srping.

Instructor: Amanov Alimzhan

Home work 3.

We've discovered untyped lambda-calculus. Now let's play around with implementation.

Exercise 0.
1) Prior to doing home work, please, read chapters 5 for lecture recap. Read chapter 6 for general understanding of how lambda calculus is implemented.
2) Download *fulluntyped* repository (as it was with *arith*, see previous HW for details).

Your work format

- Each exercise you need to complete the function definitions and implement it.
- To make it clear, each function with tests must be in **separate file**
- To show your implementation, you need to add the screenshots to your main pdf that contain:
  – left part = *func.f* file (name the files as functions) function definition, then tests
  – right part = terminal compilation\run command + output of run

Files to send: *HW3.pdf* (here you do theoretical part and screenshorts), *func.f* (for each function you did)

Format for each *func.f*

We may be will automatize checking of your works, so for each file *func.f* add between main function definition (and after auxiliary functions if there are such), but before the tests

/* >>>>>>> tests after this line <<<<< */

This line is comment so it won't affect the code.

Exercise 1. Use Church Booleans to complete the functions:
NOT=?
NAND=?
XOR=?

And implement same functions within compiler of *fulluntyped* repository.

Exercise 2. using Churh Numerals
pred = ?
minus = ?
isequal = ? // isequal a b means *a == b*
(bonus) *div = ? // interger division of numbers*

And implement same functions within compiler of *fulluntyped* repository.

Exercise 3. Implement function fib(n) that returns n-th Fibonacci number using
a) Y-combinator
b) Z-combinator
What's the difference in these 2 implementations? implement Fibonacci numbers within compiler of *fulluntyped* repository.

Exercise 4*. Exc 5.2.8. (a) do it theoretically, (b) implement it within *fulluntyped* compiler