Hasnake

Hasnake is a pure-impure hybrid functional language, inspired by Haskell. It's basically a Haskell dialect and derives most of its grammar from it(except some innovations/liberties I decided to make). Its only implementation is the Hasnake Interpreter, written in Python(hence Hasnake). The Interpreter and the language serve as my 2nd term "Scripting languages" project.

Overview:

- The main difference between Hasnake and Standard Haskell is that Hasnake supports impure functionalities within pure functions. For example, the standard I/O functions Hasnake has can be used within pure functions(which constitute every single function definition in Hasnake).
- There is no tail-call optimization in Hasnake, while Haskell optimizes tail recursion.
- While lazy evaluation does indeed exist in Hasnake and is pretty much present in almost every Hasnake construction,

arguments passed in calls to functions are evaluated immediately.

Grammar specification:

- The grammar is specified in "hasnake-grammar.html"

How to use the interpreter:

- Clone the repository and run the "main.py" file
- You can pass a directory to a hasnake source file on the command line: "python main.py -m <source_dir>".
- use the -help flag to get a list of all flags that can be passed to the interpreter on the command line.
- When you're in the interpreter, you can type ":help" to get a help menu, describing all commands the interpreter provides.
- The interpreter only 'interprets' hasnake expressions, hence you cannot make statements in the interpreter command line, i.e. function declarations and importing modules.

Project by: Martin Nakov No:16 IX 'B' class

Additional info:

- Made in: 1 week
- Total lines of code: 1109