A Language with Beautiful Syntax (ALBS)

Programming Languages and Translators
Spring 2016
February 9, 2016

Team:

Name	UNI	Role
Adam Fowler	ajf2117	Infrastructure
Brennan Wallace	bgw2119	Language Guru
Leandro Nogueria	len2120	Testing
Suhani Singhal	ss4925	Product Manager

Goal:

Our goal is to a general purpose C like language with first class functions. Our language will include the datatypes of int, char, arrays and function types. The language will be based on a C-like syntax and compile to LLVM. We want to maintain high readability by keeping all keywords of our language capitalized and disallowing programmers from using capitalized variables.

Sample Program:

```
Fn main = : Int [
  Int a = 4. // sample declaration and assignment, immutable
  Int b = 5.
  Int[5] array = [1,2,3,4,5]. // array declaration and assignment, immutable

  Fn add = Int x, Int y: Int [
            Return x + y. // explicit return statements
  ]
```

```
 Int c = add a, b. //parentheses not required for calling functions

 If c < 2 [
         //todo
 ] Elif c > 5 Or (c * 2 != 0) [
         Return -1.
 ] Else [
         Return 0.
 ]
 Print c.
 Return 0.
]
```

Data Types:

Type	Syntax	Details
function	Fn	First Class
integer	Int	32-bit, 2’s complement
character	Char	32-bit, Ascii
float	Fl	32-bit, IEEE 754
array	<type syntax>[<size>]	Single Type, Immutable

How Our Language Should Be Used:

ALBS is a general purpose programming language with immutable data types using square brackets scope indicators and featuring first class functions as data types. There are no loops in the language and recursion is used instead to generate the same functionality. The basic data types beyond functions include integers, characters (chars), floats, and arrays. The language itself will compile directly to LLVM.

Control Flow:

Control flow is achieved via if, else, and elif (else if) as well as expressions which can either be variable declarations and assignment(as variables are immutable) or function calls. Every expression returns a value. The expressions And and Or also exist in our language and return an integer value of either 0 or 1 regarding the truth value of the expressions involved. Return types are explicitly stated in each function declaration and scope is defined using square brackets “[“, “]”.

Operators:

Operators for our language include +, -, /, *, <, >, ==, =, !=, >=, <= which are the typical mathematical operations on integers and doubles. All operators operate on two values of the same type and return the same type, no casting is implicit. The less than, greater than, etc act on two datatypes of the same value and return an integer value of 0 or 1 given the truth value of false or true.

Declarations:

<<variable type>> name = <<value>>.

Where variable type is one of the data types mentioned in the data types section and name starts with a lowercase alphabetic symbol and consists of one or more of the printable ascii symbols and the value is a constant of type variable type. Each variable declaration is ended with a period(.).

Compile:

Low level LLVM code using a OCaml compiler.