

ICS 142: Compilers and Interpreters

Prof. Dr. Michael Franz & Dr. Andreas Gal, Fall Quarter 2007

Assignment 4

due at the beginning of class on November 27th

Another Compiler for the Programming Language VerySimPL+

In the previous assignment, you used the “DLX” processor as your target architecture. You will now write a second compiler, this time generating code for Microsoft’s Common Language Runtime (also known as “Dot Net”).

The objective of this assignment is learning how to adapt your knowledge to a new target architecture. Of course, since the Dot Net CLR is a *stack (0-address) machine*, this isn’t all that difficult...

In order to make this assignment even easier for you, we have created wrapper methods that will take the object code generated by you and “package” it into an executable that can be recognized by the Dot Net platform.

We have also created a way for you to access the built-in procedures of the DotNet runtime, so that your program can perform input and output.

The wrapper code can be accessed via the TA website.

Note: To obtain credit, **your solution must be submitted electronically by the due date**, using the instructions on the TA’s web page.

EBNF for VerySimPL+

letter = “a” | “b” | ... | “z”.

digit = “0” | “1” | ... | “9”.

relOp = “==” | “!=” | “<” | “<=” | “>” | “>=”.

ident = letter {letter | digit}.

number = digit {digit}.

factor = ident | number | (“(” expression “)”) | funcCall .

term = factor { (“*” | “/”) factor }.

expression = term { (“+” | “-”) term }.

relation = expression relOp expression .

assignment = “let” ident “<-” expression.

funcCall = “call” ident [(“(” [expression { “,” expression }] “)”)].

ifStatement = “if” relation “then” statSequence [“else” statSequence] “fi”.

whileStatement = “while” relation “do” statSequence “od”.

statement = assignment | funcCall | ifStatement | whileStatement .

statSequence = statement { “;” statement }.

varDecl = “var” indent { “,” ident } “;” .

computation = “main” [varDecl] “{” statSequence “}” “.” .

Predefined Function

InputNum() read a number from the standard input

Predefined Procedure

OutputNum(x) write a number to the standard output

OutputNewLine() write a carriage return to the standard output

The start symbol of this grammar is “computation”.