SIMPLEX
Syntax for International Monetary, Property and Liquidity Exchange

Steven Chen
Gilbert Hom
Kelvin Jiang
Eric Zhang

{stc2104, gch2102, kxj1, ehz2101}@columbia.edu
Is SIMPLEX really that SIMPLE?

- Syntax for International Monetary, Property, and Liquidity Exchange
- Lightweight non-object oriented C like language
- Superior “Dynamic Data Casting”
- Superior data type output representation
- Similar syntax to that of C, C++, and Java
Is SIMPLEX for me?

• SIMPLEX was created for the financial audience
• Provides data types that revolve around financial applications
• Allows focus on analysis rather than the minute details that languages like C++ and Java need
• Allows handling of mathematical equations with many data type
• Simple and quick
Data Types

• Five main Data Types / Two types of Constants
  – Number: [number constant] 64bit floating point numbers
  – Currencies: [number constant]
    USD, YEN, EUR, CAD, GBP, AUD, CHF, CNY, MXN, SOS
  – Date: [number constant] Three data types:
    year, month, day
  – Rate: [number constant] Representation for percentages
  – String: [string constant] Data type that allows storage of arbitrary sequences of characters

• Type Casting
  • (type) expression

• All number constant Data Type expressions are handled by SIMPLEX and will not return any exceptions
Statements

- To keep programming familiar with today’s popular program languages, SIMPLEX follows similar statement syntax to that of C, C++, and java
  - Assignment Statements
  - Jump Statements
  - Procedure Call Statements
  - Return Statements
  - Conditional Statements
  - Iterative Statements
  - Output Statements
  - Input Statement
Procedures / Scoping

- SIMPLEX allows creation of user defined procedures
  
  ```
  type-specifier procedure-identifier (identifier-list) {
    statement-list
  }
  ```

- Static Scoping
- Applicative Order Evaluation
- Recursion
SIMPLEX Special Features

• User Defined Function and casting of currencies
  
  USD calculateYearlyRent(USD a, EUR b, CAD c, YEN d){
  USD yearly;
  yearly = ( (USD)a + (USD)b + (USD)c + (USD)d );
  return yearly;
}

• Preformatted Currency Outputs
  
  rent_2008 = 3496340.6446234623462346;
  print ("Total 2008 Rent: " + rent_2008);
  Output: Total 2007 Rent: $3496340.64

• Built-in Input Function
  
  – Takes user input from the command line
  
  print("Enter Total Mortgage amount: ");
  input(totalMortgage);
• Intuitive mathematical operators
  – Percent Sign (automatically divides by 100)
    ```
    rate a;
    a = 5%;       // stored as 0.05
    ```
  – Exponential Powers
    ```
    number a;
    a = 3 ^ 3;    // a stores 27
    ```
Top Level Design

Input File (Characters) → Lexer → Token Stream → Parser

Abstract

Syntax → Tree → Walker → IR Classes → Translated Code
Class Hierarchy
Class Hierarchy
void main() {
    USD x;
    x = x + 1;
    print(x);
}
Compiling and Testing

• Console
  – Customized interface to compile and run SIMPLEX programs
  – Does so through a series of shell commands

• Regression Test Suite
  – Suite built on top of the console
  – Test cases isolate specific features
    • Test suite includes: declaration, for, arithmetic, if, dangling else, user-defined functions
  – Test Suite run constantly after updates to compiler

• Test Applications
  – Larger scale applications integrating several features of the language at the same time
Issues We Faced and Lessons Learned

• Dynamic group with different personalities and specialties
• KISS – Keep it Simple Stupid
• Strong foundation and working platform
• Thorough and complete testing

• Planning
  – Solid foundations based on solid ideas
  – Allow for adaptability and changes
  – Revisions Revisions Revisions!
SIMPLEX

Syntax for International Monetary, Property and Liquidity Exchange

Steven Chen
Gilbert Hom
Kelvin Jiang
Eric Zhang

{stc2104, gch2102, kxj1, ehz2101}@columbia.edu