COMS 4115 Programming Languages and Translators

Project Proposal

Ryan Chun (ec3255)
Garrison Grogan (gg2652)
Ryan Jay Koning (rjk2153)
Trisha Maniar (trm2144)

SOSL – Set Operations Simplification Language

SOSL is a language that focuses on simplifying formal set theory operations in order to make it easy to manipulate sets and create functions that act on them, their elements, and/or subsets. Rather than rely on the user to create functions for basic set operations, SOSL will provide special operators for these functions (intersection, union, Cartesian product, etc.).
SOSL is meant to allow the programmer to write programs that evaluate set theoretic equations and functions without having to go through the cumbersome process of defining the functions long hand in a given language. The power of our language is in the operator set, because the programmer can forego calling a cumbersome standard library function and simply use SOSL's special operators. This makes things concise and simplifies, relative to other languages like Java, the notion of working with sets in a programming environment.

Language Features:

- Sets and their operations: Easily define sets, and perform set theoretic operations like union, intersection, and complement. Cartesian products are represented as a set of arrays.
- Element types such as integers, floating point numbers, characters, booleans, arrays, and sets themselves. Elements can be placed in sets or manipulated themselves.
- Loops and iterators over sets. Semantically these are different since sets are not ordered while arrays are.
- Conditionals for checking various set or element conditions
- Global variables and functions defined at the top level of the program. The program executes linearly down the file. Functions are defined in called in a C style fashion.
- A standard library providing functions like print for usability.

Reserved Keywords:

- int
- float
- char

- boolean
- set
- for
- forEach
- in
- if
- else
- void

Data Types	
Symbol	**Description**
int	Integer
float	Float
char	Character
boolean	Boolean
[]	Array
void	Void
set	Set

Operators		
Symbol	**Description**	**Returns**
∩ (:n)	Intersection	Set
∪ (:u)	Union	Set
∈ (:i)	Element of	boolean
:c	Complement	Set
×	Cartesian Product	Set of Arrays

|Set|	Cardinality	int
+	Integer Addition	int
-	Integer Subtraction	int
*	Integer Multiplication	int
/	Integer Division	int
+.	Float Addition	float
-.	Float Subtraction	float
*.	Float Multiplication	float
/.	Float Division	float

Logics	
Symbol	**Description**
if/else	condition
for/forEach in	loop/iterate

Comparers	
Symbol	**Description**
<	Less than
>	Greater than
<=	Less than or Equal To
>=	Greater than or Equal To
==	Equal To

Program Example

```
set subtract(Set A, Set B) {
    return A∩(B :c A);
}

set dotProduct(Set X, Set Y) {
    if (|X| != |Y|) return NULL;
    int result = 0;
    forEach element in X×Y{
        result = result + element[0]*element[1];
    }

}

boolean isSubset(Set X, Set Y){
    if (|X| < |Y|) return NULL;
    forEach element in Y{
        if (!(element ∈ X)) { return false;}
    }
    return true;
}

void main(){
    Set A = { 1, 2, 3, 4 };
    Set B = { 3, 4 };

    Set C = subtract(A, B); // C = { 1, 2 }
    boolean subset = isSubset(A, C);

    if (subset) {int dotP = dotProduct(B, C);}
    else {}
}

main();
```