|  |
| --- |
| ARIZONA STATE UNIVERSITY |
| SuViv |
| SER 502 |

Suhas Chandrashekar & Vivek Jadon

4-27-2016

Contents

1. Introduction ………………………………………………2
2. Install Instructions ……………………………………….2
3. Compile and run SuViv programs ………………………..2
4. Language Design and features …………………………...2
5. Example Programs ………………………………………3
6. Introduction

SuViv is a modular and strongly typed language. The language is compiled using Java and the runtime is handled using python. A plug-in called ANTLR-4 is used to parse and generate intermediate code (assembly instructions).

1. Installation Instructions

Tools and Environment: Java 1.8 or higher, Python 2.7 and Eclipse Luna

The source code can be downloaded from the GIT Link: or the JAR file.

Compile all the java files to setup the project through Eclipse.

1. Compile and run SuViv programs
2. Give the input file name and directory which contains the sample program in the SJ\_Compiler/src/edu/asu/msse/sj/Constants.java file
3. Run the program.
4. Intermediate code is generated and will be stored in the path specified in the Constants.java file.
5. Open Command Prompt or Terminal.
6. Go to the directory where the project is stored. For example, in our case –

C:\Users\suhas\workspace\SJ\_Compiler>

1. Give the command –

python Runtime/Runtime.py Examples/if\_else\_program

1. This command will execute the intermediate code and display the output on the console.
2. Language Design and features
3. Arithmetic Operators in SuViv:
4. +
5. –
6. \*
7. /
8. Relational Operators supported in SuViv:
   1. ==
   2. !=
   3. <
   4. >
   5. <=
   6. >=
9. Data Types in SuViv:
   1. Int
   2. Bool
   3. String
   4. Stack
10. Scope Handling:

Global and local scope of variables are handled in SuViv. The scope of variables inside and outside the nested blocks are also handled in terms of their scope.

1. Conditional Statements:

SuViv provides the feature of handling conditional statements through if-else constructs.

1. Looping Constructs:

SuViv provides a simple while loop to support loops.

1. Example Programs
2. Factorial of a number

function fact(n){

result = 1;

print "number\_is";

print n;

while(n>1){

result = result\*n;

n = n-1;

}

print "Factorial\_is";

print result;

}

rand = 5;

call fact(rand);

1. If-else Program:

var x = 5;

var y = 10;

if( x == y ){

print "entered if block";

print x;

print "both are equal";

} else{

print "both the numers are unequal";

}

1. Stack Program:

var stackVar = STACK;

print "push element onto stack";

stackVar.push(1);

print "push element onto stack";

stackVar.push(2);

print "push element onto stack";

stackVar.push(3);

print "push element onto stack";

stackVar.push(4);

print "push element onto stack";

stackVar.push(5);

print "push element onto stack";

stackVar.push(6);

print "push element onto stack";

stackVar.push(7);

print "push element onto stack";

stackVar.push(8);

print "push element onto stack";

stackVar.push(9);

print "push element onto stack";

stackVar.push(10);

print "element popped";

popEle = stackVar.pop();

print popEle;

print "element popped";

popEle = stackVar.pop();

print popEle;

print "element popped";

popEle = stackVar.pop();

print popEle;

print "element popped";

popEle = stackVar.pop();

print popEle;

print "element popped";

popEle = stackVar.pop();

print popEle;

print "element popped";

popEle = stackVar.pop();

print popEle;

print "element popped";

popEle = stackVar.pop();

print popEle;

print "element popped";

popEle = stackVar.pop();

print popEle;

print "element popped";

popEle = stackVar.pop();

print popEle;

print "element popped";

popEle = stackVar.pop();

print popEle;

1. Recursive Program:

var ct = 0;

function recurse\_func(x){

if(ct < x){

print x;

x = x - 1;

call recurse\_func(x);

}else{

print "executed";

}

}

var a = 10;

call recurse\_func(a);