Team 12 - ARROW

Divya Yadamreddi

Harshitha Katpally

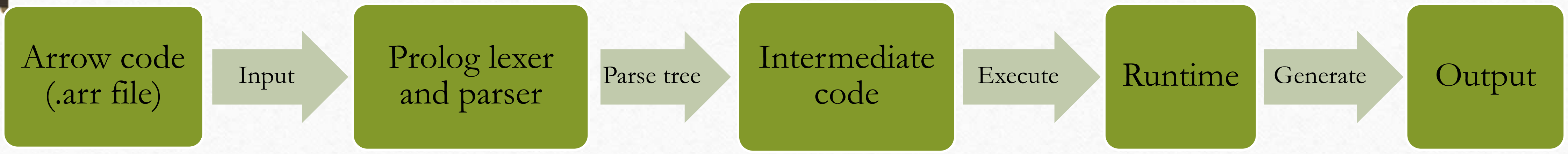
Venkata Akhil Madiraju

Venkata Sai Shirisha Kakarla

Key features of the language

- Imperative language - Inspiration from C language
- Statically typed
- Data types – int, bool, string
- Decision control statement – if-else, nested if-else
- Looping construct – while
- Developed completely in Prolog.

Language Design Flow



Lexical Analyzer and Parser

- Written in Prolog DCG.
- Lexer generates a list of tokens eliminating spaces, tabs and new lines.
- Parser generates a parse tree using the grammar of the language.
- Parses in a top-down fashion.

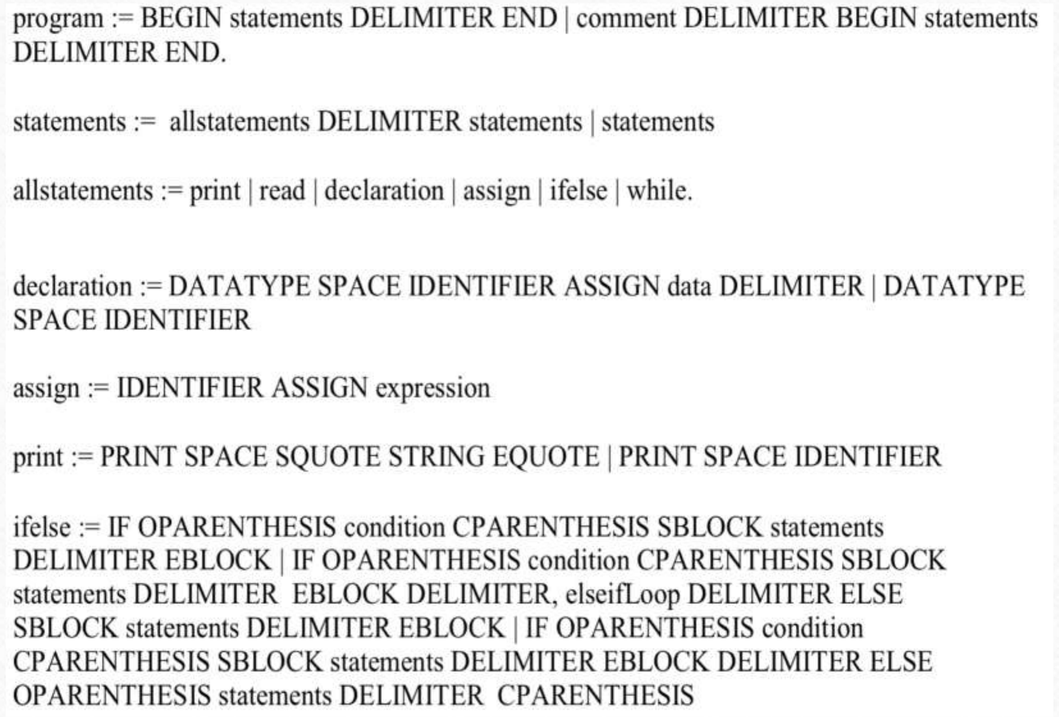
Intermediate Code

- Parse tree is the intermediate code.
- Output to a separate file.
- Easily generated using DCG.
- Used by the interpreter to evaluate the code.

Runtime

- Evaluated using Prolog.
- Environment is a list of tuples.
- Used reduction rules to evaluate the code.

Language Grammar



program := BEGIN statements DELIMITER END | comment DELIMITER BEGIN statements DELIMITER END.

statements := allstatements DELIMITER statements | statements

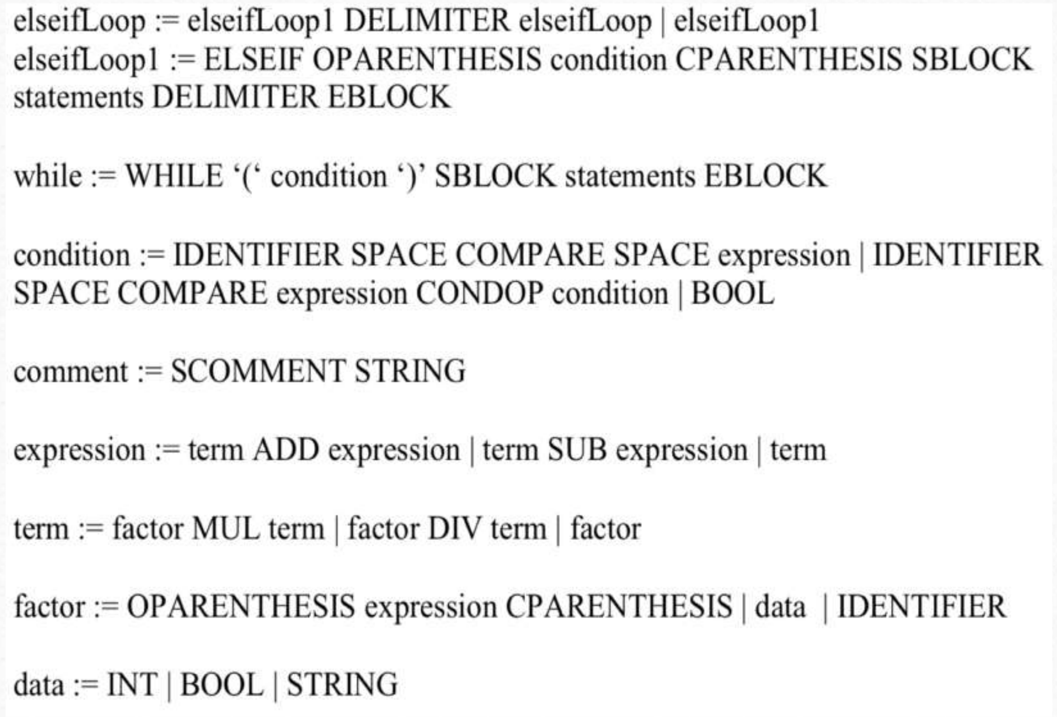
allstatements := print | read | declaration | assign | ifelse | while.

declaration := DATATYPE SPACE IDENTIFIER ASSIGN data DELIMITER | DATATYPE SPACE IDENTIFIER

assign := IDENTIFIER ASSIGN expression

print := PRINT SPACE SQUOTE STRING EQUOTE | PRINT SPACE IDENTIFIER

ifelse := IF OPARENTHESIS condition CPARENTHESIS SBLOCK statements DELIMITER EBLOCK | IF OPARENTHESIS condition CPARENTHESIS SBLOCK statements DELIMITER EBLOCK DELIMITER, elseifLoop DELIMITER ELSE SBLOCK statements DELIMITER EBLOCK | IF OPARENTHESIS condition CPARENTHESIS SBLOCK statements DELIMITER EBLOCK DELIMITER ELSE OPARENTHESIS statements DELIMITER CPARENTHESIS



elseifLoop := elseifLoop1 DELIMITER elseifLoop | elseifLoop1
elseifLoop1 := ELSEIF OPARENTHESIS condition CPARENTHESIS SBLOCK statements DELIMITER EBLOCK

while := WHILE '(' condition ')' SBLOCK statements EBLOCK

condition := IDENTIFIER SPACE COMPARE SPACE expression | IDENTIFIER SPACE COMPARE expression CONDOP condition | BOOL

comment := SCOMMENT STRING

expression := term ADD expression | term SUB expression | term

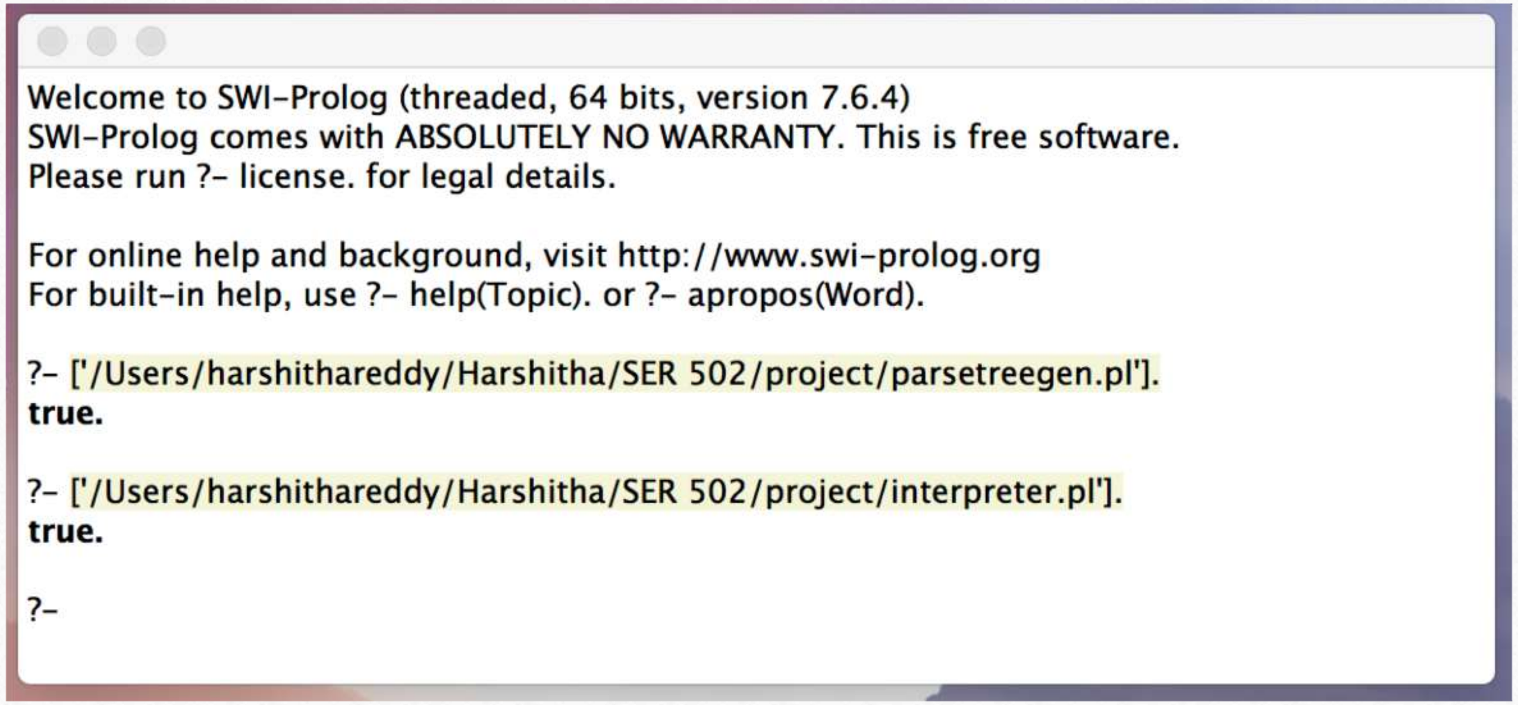
term := factor MUL term | factor DIV term | factor

factor := OPARENTHESIS expression CPARENTHESIS | data | IDENTIFIER

data := INT | BOOL | STRING

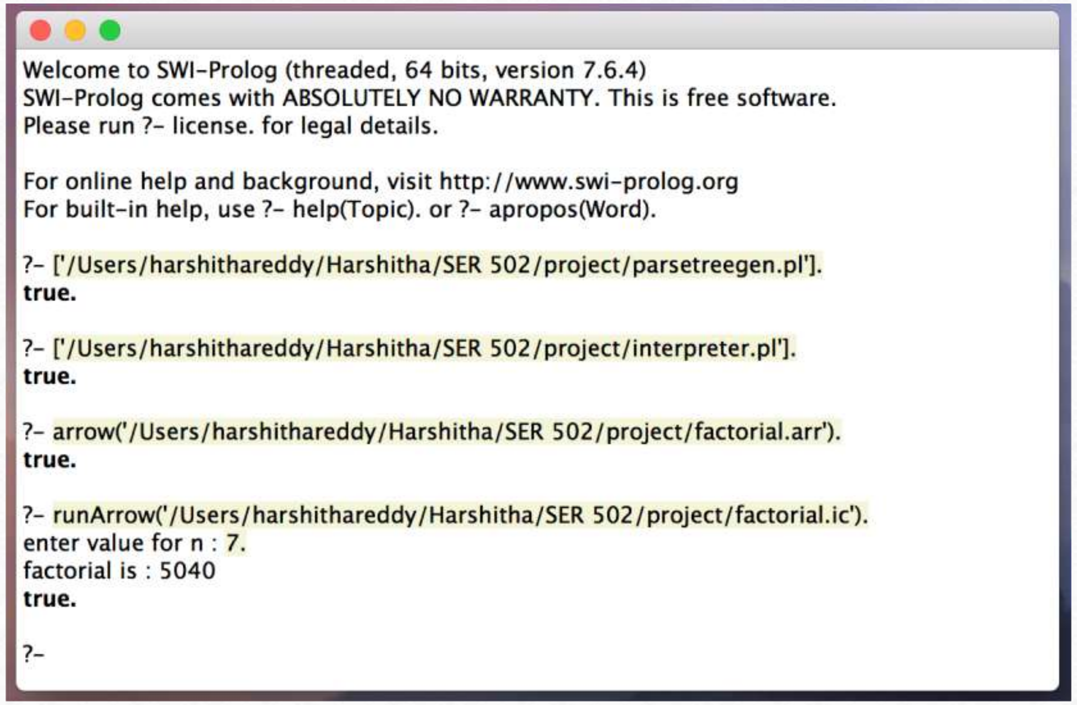
Language Demonstration

Executing Prolog Files



```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- ['/Users/harshithareddy/Harshitha/SER 502/project/parsetreegen.pl'].
true.

?- ['/Users/harshithareddy/Harshitha/SER 502/project/interpreter.pl'].
true.

?-
```

Runtime



```
Welcome to SWI-Prolog (threaded, 64 bits, version 7.6.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit http://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?- ['/Users/harshithareddy/Harshitha/SER 502/project/parsetreegen.pl'].
true.

?- ['/Users/harshithareddy/Harshitha/SER 502/project/interpreter.pl'].
true.

?- arrow('/Users/harshithareddy/Harshitha/SER 502/project/factorial.arr').
true.

?- runArrow('/Users/harshithareddy/Harshitha/SER 502/project/factorial.ic').
enter value for n : 7.
factorial is : 5040
true.

?-
```

Thank You