STRIGER

SER 502-EMERGIN LANGUAGES AND PROGRAMMING PARADIGMS PROJECT GROUP 9

Github: https://github.com/ntarunasy/SER502-Striger-Team9/tree/main

TEAM MEMBERS

- VENKATA SAI TARUN NUKA
- SRI VIKAS GANUGU
- NITIN SURYA MOTURU

Striger Features

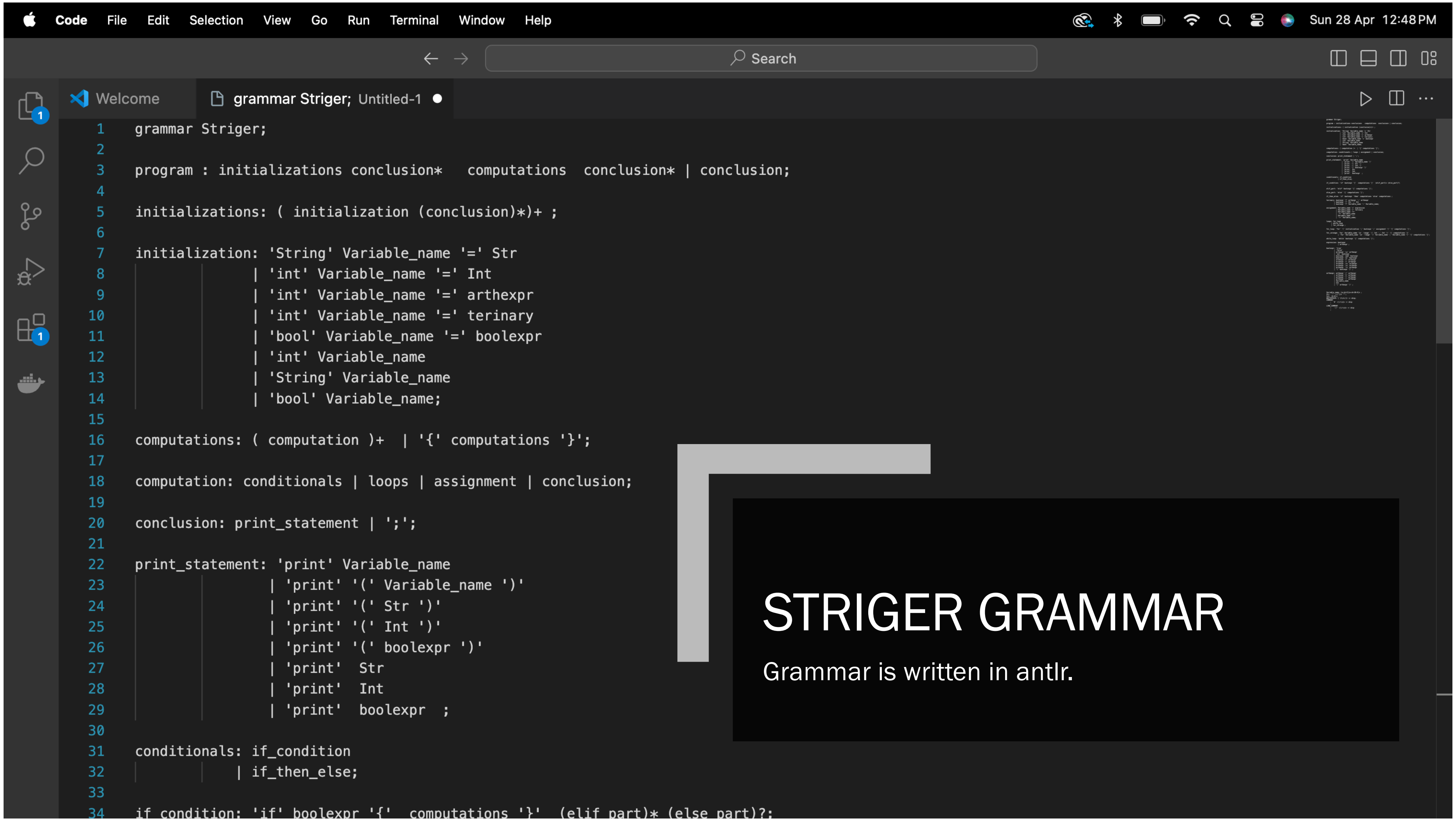
- It supports three types of data types: Integer, String and Boolean.
- It supports two types of printing of variables: print(variable) and print variable.
- It has two types of conditional statements:

1. if_condition: a normal if else condition where add ‘:’ at the end of both the keywords to increase readability of the code.
2. If_then_else:It is also similar to above one, but here there won’t be elif and ‘:’.

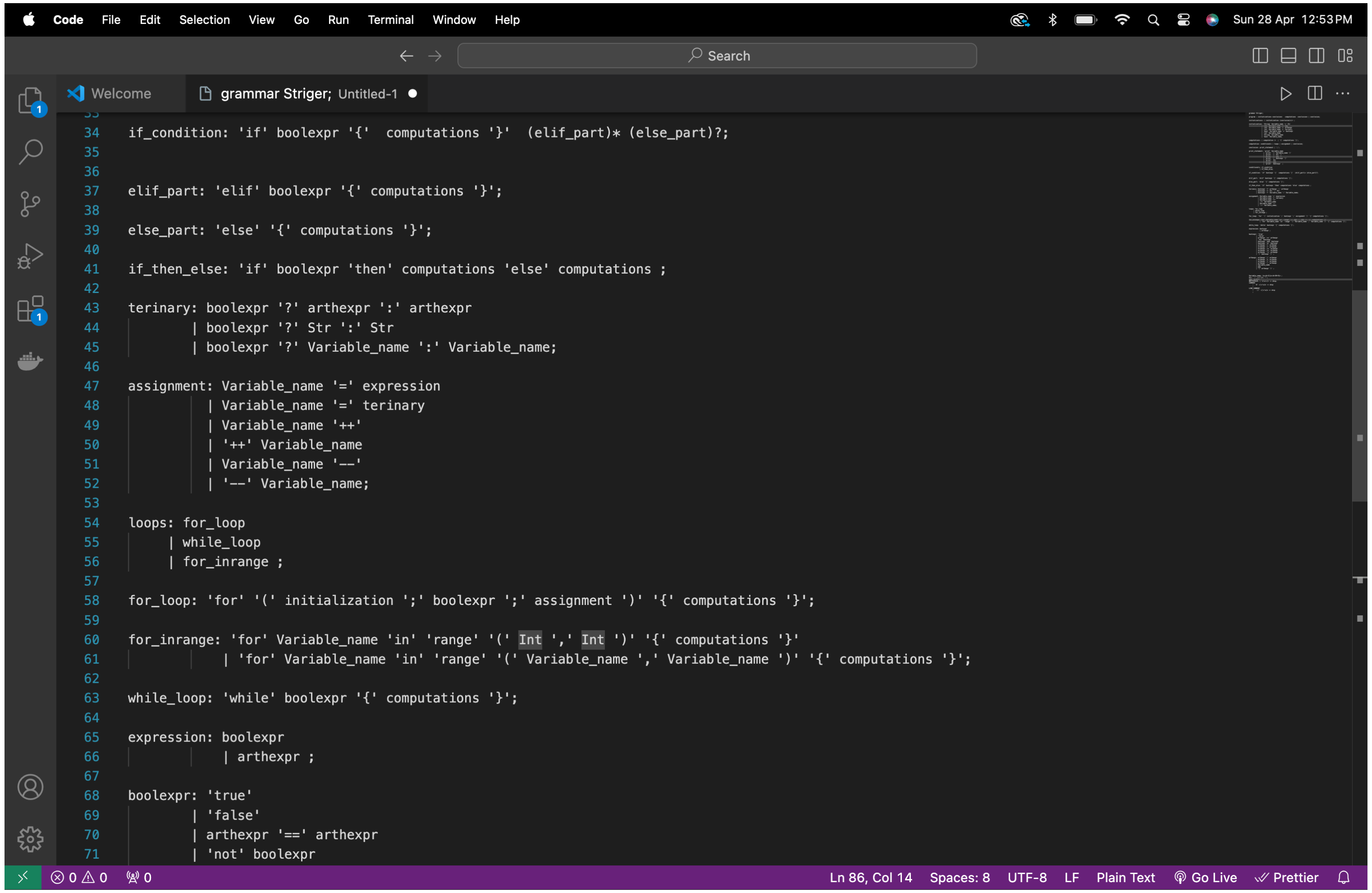
- It has three types of loops: for_loop, while_loop and for_inrange.
- The arithmetic operations it support are addition, subtraction, multiplication, division.
- The Boolean operations it support are ‘true’, false, ==, not, and >,<,>=,<=,!=‘.
- Variables name contain small and capital letters, along with numbers. But variable names shouldn’t start with number
- The language basically supports all the basic features of an advanced language.

STRIGER GRAMMAR

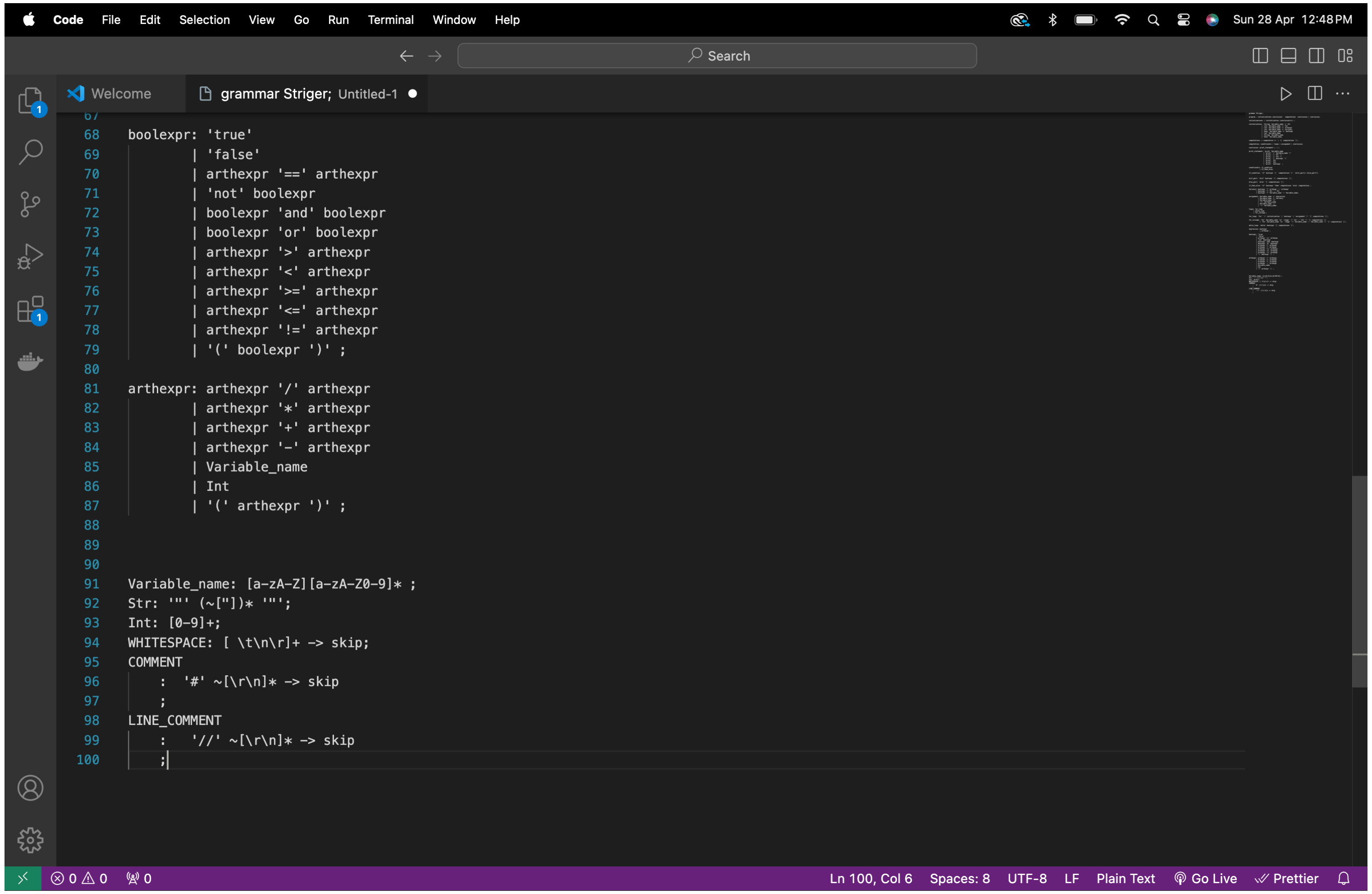
Grammar is written in antlr.


Code File Edit Selection View Go Run Terminal Window Help
Sun 28 Apr 12:48 PM
Search
Welcome
grammar Striger; Untitled-1

```
grammar Striger;

program : initializations conclusion*   computations  conclusion* | conclusion;

initializations: ( initialization (conclusion)*)+ ;

initialization: 'String' Variable_name '=' Str
              | 'int' Variable_name '=' Int
              | 'int' Variable_name '=' arthexpr
              | 'int' Variable_name '=' terinary
              | 'bool' Variable_name '=' boolexpr
              | 'int' Variable_name
              | 'String' Variable_name
              | 'bool' Variable_name;

computations: ( computation )+  | '{' computations '}';

computation: conditionals | loops | assignment | conclusion;

conclusion: print_statement | ';';

print_statement: 'print' Variable_name
                | 'print' '(' Variable_name ')'
                | 'print' '(' Str ')'
                | 'print' '(' Int ')'
                | 'print' '(' boolexpr ')'
                | 'print'  Str
                | 'print'  Int
                | 'print'  boolexpr  ;

conditionals: if_condition
            | if_then_else;

if_condition: 'if' boolexpr '{'  computations '}'  (elif_part)* (else_part)?;
```


Code File Edit Selection View Go Run Terminal Window Help
Sun 28 Apr 12:53 PM
Search
Welcome
grammar Striger; Untitled-1
Ln 86, Col 14 Spaces: 8 UTF-8 LF Plain Text Go Live Prettier

```
if_condition: 'if' boolexpr '{'  computations '}'  (elif_part)* (else_part)?;


elif_part: 'elif' boolexpr '{' computations '}';

else_part: 'else' '{' computations '}';

if_then_else: 'if' boolexpr 'then' computations 'else' computations ;

terinary: boolexpr '?' arthexpr ':' arthexpr
        | boolexpr '?' Str ':' Str
        | boolexpr '?' Variable_name ':' Variable_name;

assignment: Variable_name '=' expression
          | Variable_name '=' terinary
          | Variable_name '++'
          | '++' Variable_name
          | Variable_name '--'
          | '--' Variable_name;

loops: for_loop
     | while_loop
     | for_inrange ;

for_loop: 'for' '(' initialization ';' boolexpr ';' assignment ')' '{' computations '}';

for_inrange: 'for' Variable_name 'in' 'range' '(' Int ',' Int ')' '{' computations '}'
            | 'for' Variable_name 'in' 'range' '(' Variable_name ',' Variable_name ')' '{' computations '}';

while_loop: 'while' boolexpr '{' computations '}';

expression: boolexpr
            | arthexpr ;

boolexpr: 'true'
        | 'false'
        | arthexpr '==' arthexpr
        | 'not' boolexpr
```



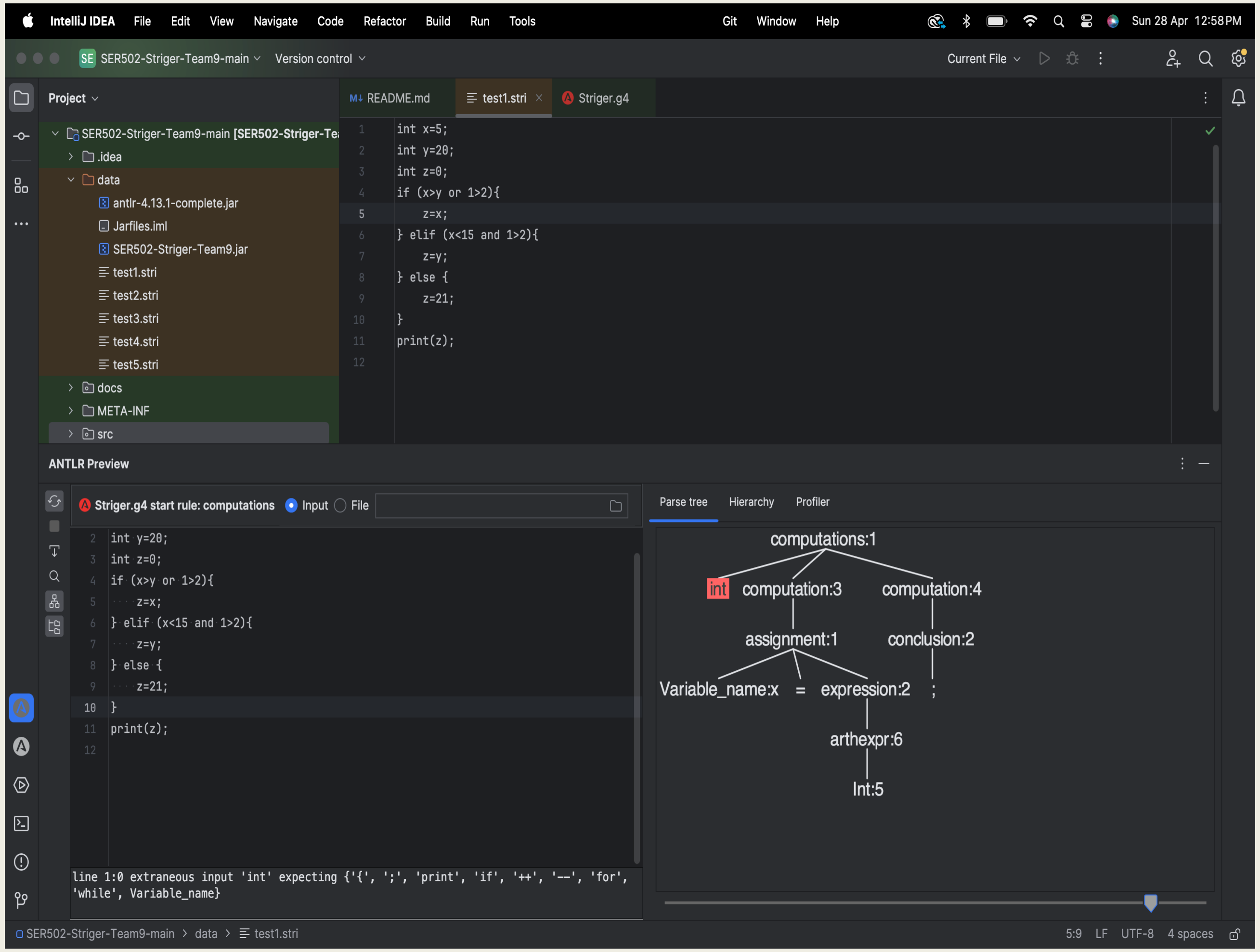
Code File Edit Selection View Go Run Terminal Window Help
Sun 28 Apr 12:48 PM
Search
Welcome
grammar Striger; Untitled-1
Ln 100, Col 6 Spaces: 8 UTF-8 LF Plain Text Go Live Prettier

```
boolexpr: 'true'
        | 'false'
        | arthexpr '==' arthexpr
        | 'not' boolexpr
        | boolexpr 'and' boolexpr
        | boolexpr 'or' boolexpr
        | arthexpr '>' arthexpr
        | arthexpr '<' arthexpr
        | arthexpr '>=' arthexpr
        | arthexpr '<=' arthexpr
        | arthexpr '!=' arthexpr
        | '(' boolexpr ')' ;

arthexpr: arthexpr '/' arthexpr
        | arthexpr '*' arthexpr
        | arthexpr '+' arthexpr
        | arthexpr '-' arthexpr
        | Variable_name
        | Int
        | '(' arthexpr ')' ;



Variable_name: [a-zA-Z][a-zA-Z0-9]* ;
Str: '"' (~["])* '"';
Int: [0-9]+;
WHITESPACE: [ \t\n\r]+ -> skip;
COMMENT
    :  '#' ~[\r\n]* -> skip
    ;
LINE_COMMENT
    :   '//' ~[\r\n]* -> skip
    ;
```

Sample Code

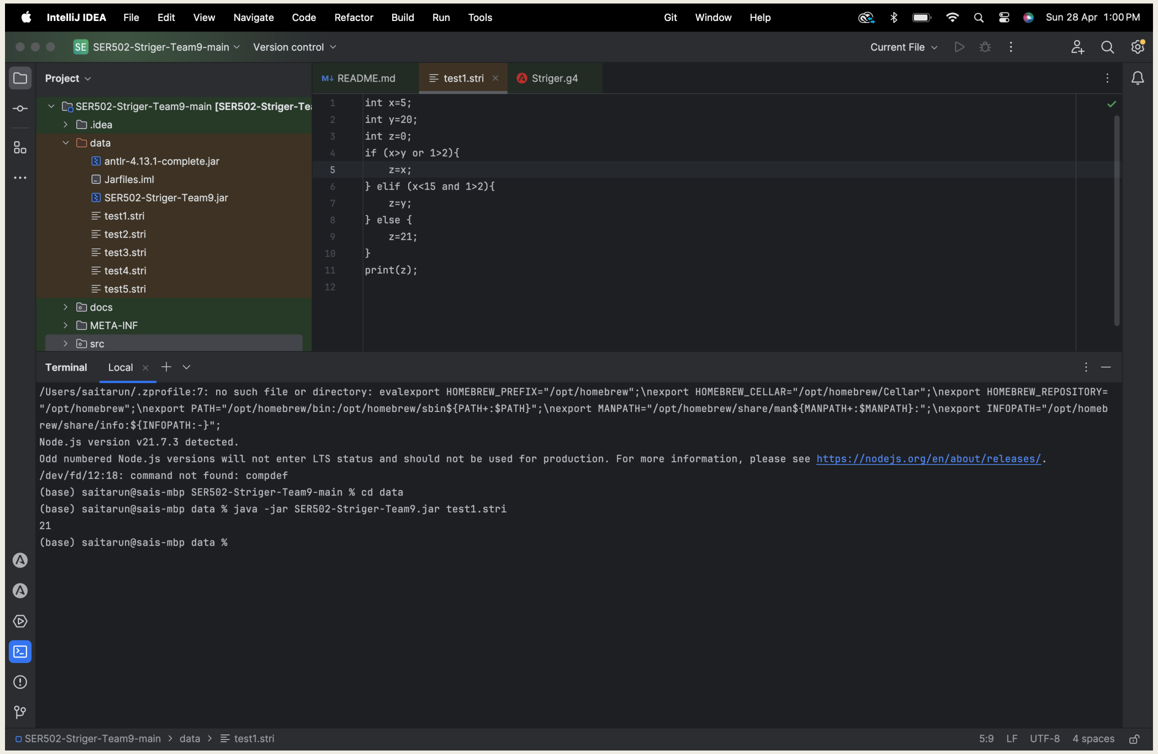
```
int x=5;
int y=20;
int z=0;
if (x>y or 1>2){
    z=x;
} elif (x<15 and 1>2){
    z=y;
} else {
    z=21;
}
print(z);
```

PARSE TREE



```
int x=5;
int y=20;
int z=0;
if (x>y or 1>2){
    z=x;
} elif (x<15 and 1>2){
    z=y;
} else {
    z=21;
}
print(z);
```

```
line 1:0 extraneous input 'int' expecting {'{', ';', 'print', 'if', '++', '--', 'for',
'while', Variable_name}
```

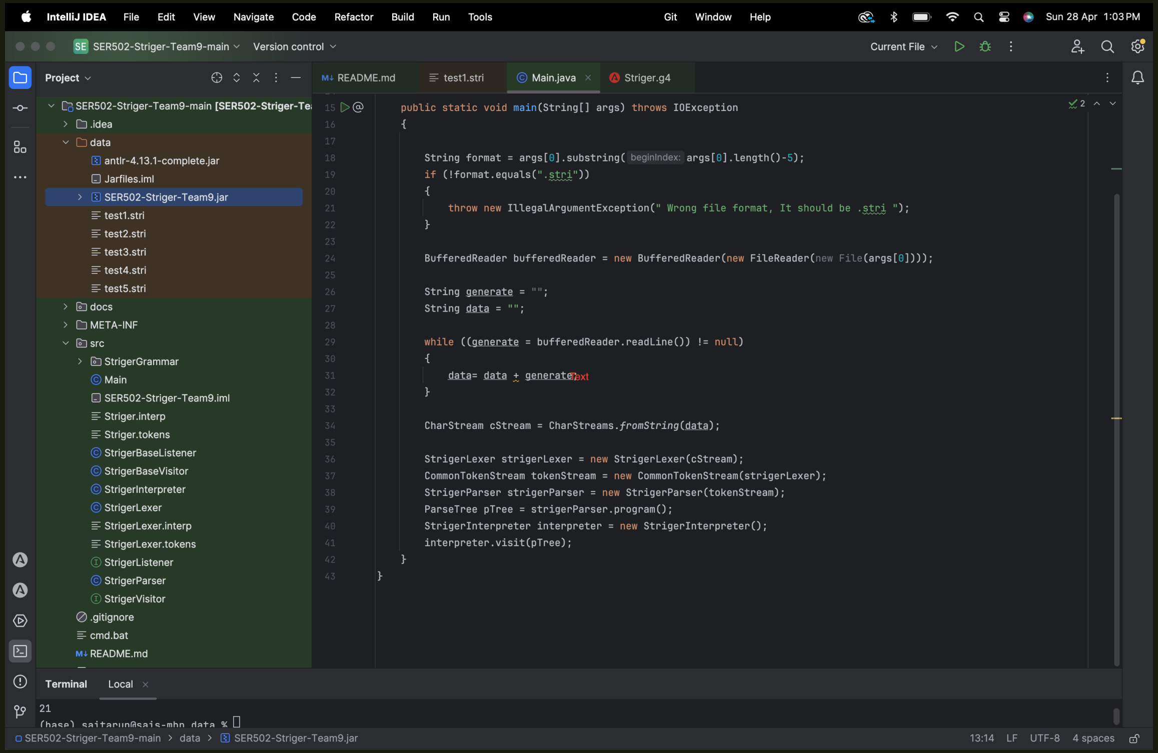


```
int x=5;
int y=20;
int z=0;
if (x>y or 1>2){
    z=x;
} elif (x<15 and 1>2){
    z=y;
} else {
    z=21;
}
print(z);
```

```
/Users/saitarun/.zprofile:7: no such file or directory: evalexport HOMEBREW_PREFIX="/opt/homebrew";\nexport HOMEBREW_CELLAR="/opt/homebrew/Cellar";\nexport HOMEBREW_REPOSITORY=
"/opt/homebrew";\nexport PATH="/opt/homebrew/bin:/opt/homebrew/sbin${PATH+:$PATH}";\nexport MANPATH="/opt/homebrew/share/man${MANPATH+:$MANPATH}:";\nexport INFOPATH="/opt/homeb
rew/share/info:${INFOPATH:-}";
Node.js version v21.7.3 detected.
Odd numbered Node.js versions will not enter LTS status and should not be used for production. For more information, please see https://nodejs.org/en/about/releases/.
/dev/fd/12:18: command not found: compdef
(base) saitarun@sais-mbp SER502-Striger-Team9-main % cd data
(base) saitarun@sais-mbp data % java -jar SER502-Striger-Team9.jar test1.stri
21
(base) saitarun@sais-mbp data %
```

Sample Run and Output

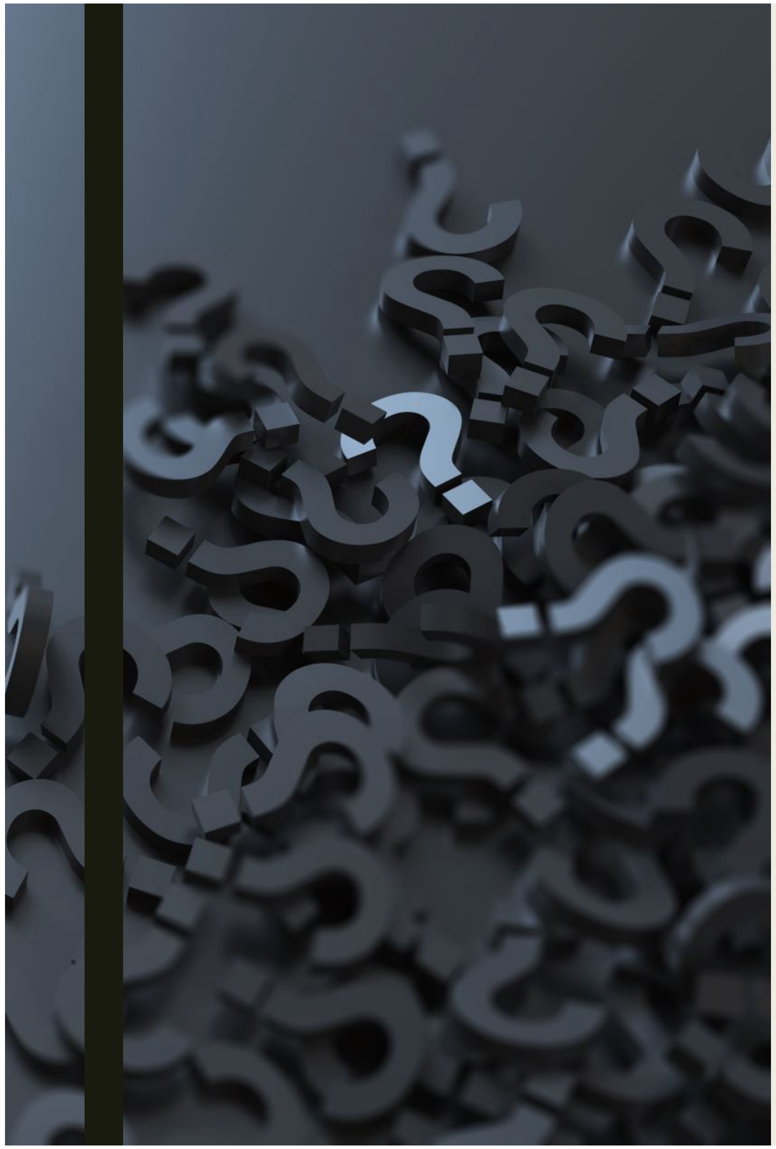
To run the code, the command is
java -jar SER502-Striger-Team9.jar file_name.stri



```java
    public static void main(String[] args) throws IOException
    {

        String format = args[0].substring( beginIndex: args[0].length()-5);
        if (!format.equals(".stri"))
        {
            throw new IllegalArgumentException(" Wrong file format, It should be .stri ");
        }

        BufferedReader bufferedReader = new BufferedReader(new FileReader(new File(args[0])));

        String generate = "";
        String data = "";

        while ((generate = bufferedReader.readLine()) != null)
        {
            data= data + generateText
        }

        CharStream cStream = CharStreams.fromString(data);

        StrigerLexer strigerLexer = new StrigerLexer(cStream);
        CommonTokenStream tokenStream = new CommonTokenStream(strigerLexer);
        StrigerParser strigerParser = new StrigerParser(tokenStream);
        ParseTree pTree = strigerParser.program();
        StrigerInterpreter interpreter = new StrigerInterpreter();
        interpreter.visit(pTree);
    }
}
```

```
21
(base) saitarun@sais-mbp data %
```

INTERPRETER ARCHITECTURE



Future Implementation

- Additional large datatypes like double and long.
- Providing input during run time
- Arrays and lists
- Functions

THANK YOU