**Compiler course work 1 report**

Task I.1:

Aim is to extend MiniTriangle with a repeat-loop.

Should add extra token, extend AST have new print method and should let scanner and parser can identify new token.

First extend grammar:

Lexical Syntax:

*Keyword* repeat | until

Context-Free Syntax:

*Command* |repeat *command* until *expression*

Abstract Syntax:

Command |repeat command until expression CmdRepeat

Coding part:

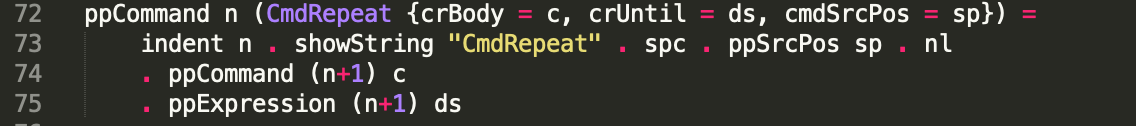
Extend ‘repeat’, ‘until’ token to token.hs



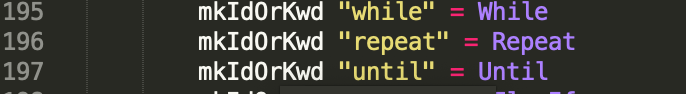
Extend command in AST.hs



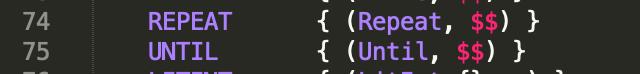
Extend ppcommand in PPAST.hs to make parser print properly



Extend mkIdOrKwd in scanner.hs to let compiler can properly identify the new token.



Extend token and command in parser.y





Task I.2

Should add extra token, extend AST have new print method and should let scanner and parser can identify new token.

First extend grammar:

Context-Free Syntax:

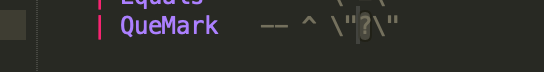
*Expression* | expression ? expression : expression

Abstract Syntax:

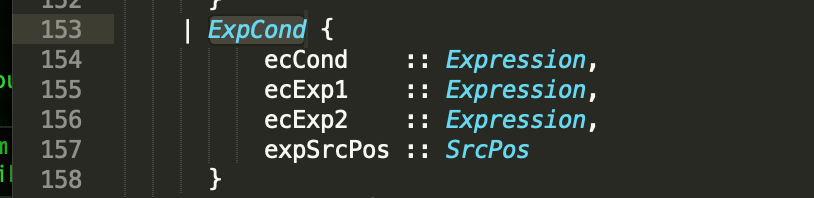
*Expression* | expression ? expression : expression ExpCond

Coding part:

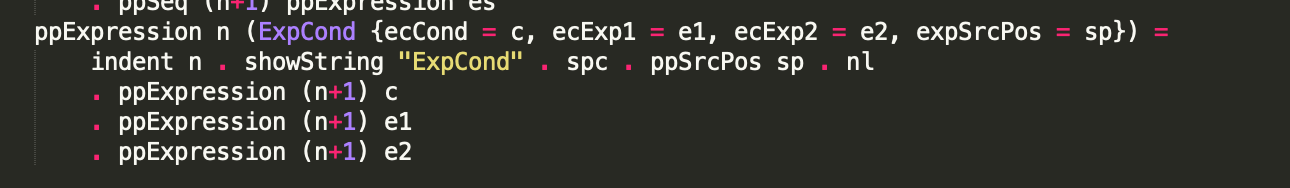
Extend ‘?’ token to token.hs since ‘?’ is already there



Exrend AST.hs:



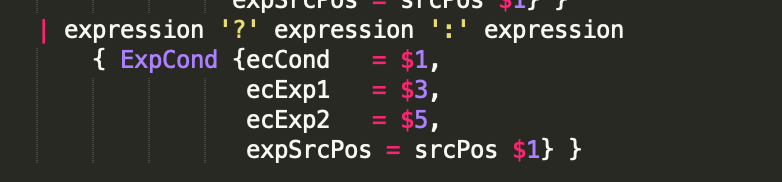
Extend ppexpression to PPAST:



Scanner.hs:



Parser.y

Task I.3:

This task should use maybe keyword to allow the condition command exist, and use the ppSeq to allow multiple Elseif command.

First extend grammar:

Lexical Syntax:

*Keyword* elseif

Context-Free Syntax:

*Command* | if expression then command

| if expression then command elseifcommands else command

| if expression then command elseifcommands

*Elseifcommands* -> Elseifcommand | elseifcommand elseifcommands

*Elseifcommand* -> elseif expression then command

Abstract Syntax:

*Command* | if expression then command CmdIf

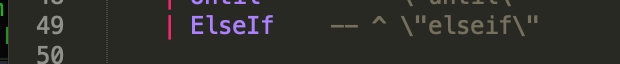
| if expression then command elseifcommand\* else command CmdIf

| if expression then command elseifcommand\* CmdIf

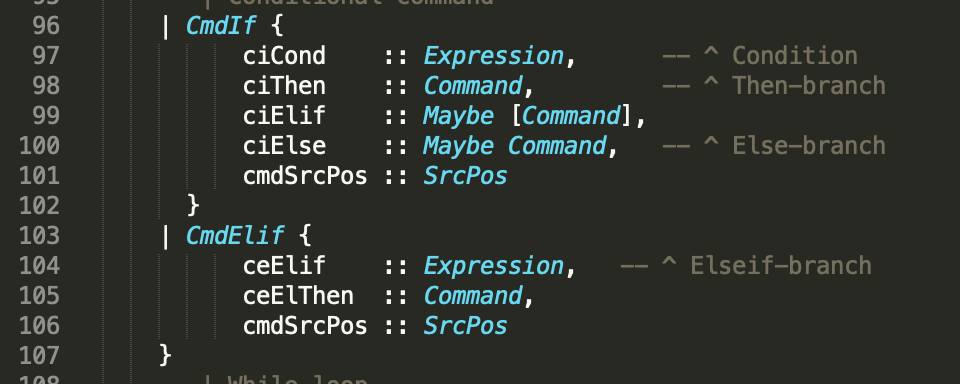
*Elseifcommand* -> elseif expression then command CmdElIf

Coding part:

Token.hs:



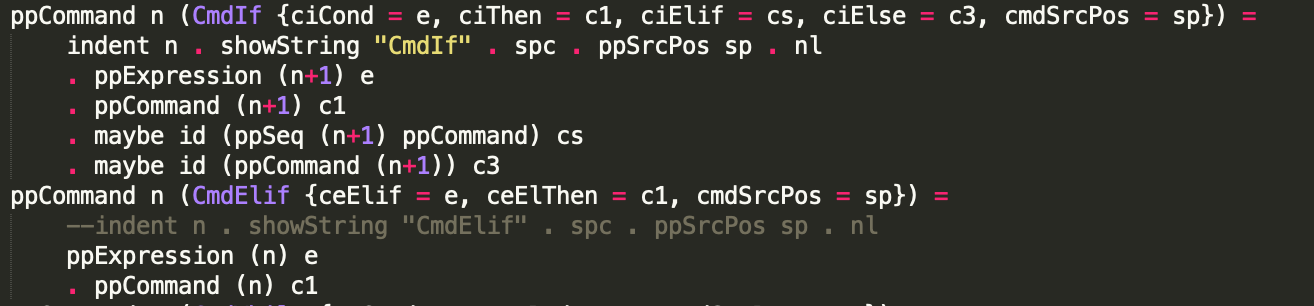
AST.hs:



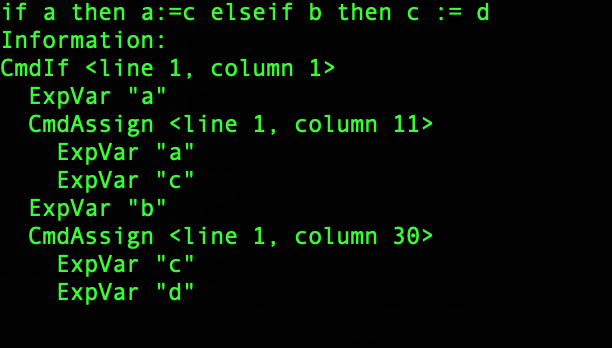
Add maybe [command] means this is sequence of command and it’s conditional.

And add CmdElif is that I consider else if is a kind of command but it’s following by CmdIf in parser.y do not have a command start with elseif it prevent error grammar like “elseif a then b” without if command.

PPAST.hs:



The ppcommand of cmdElif is a bit different to other command since I should make the command indent properly.



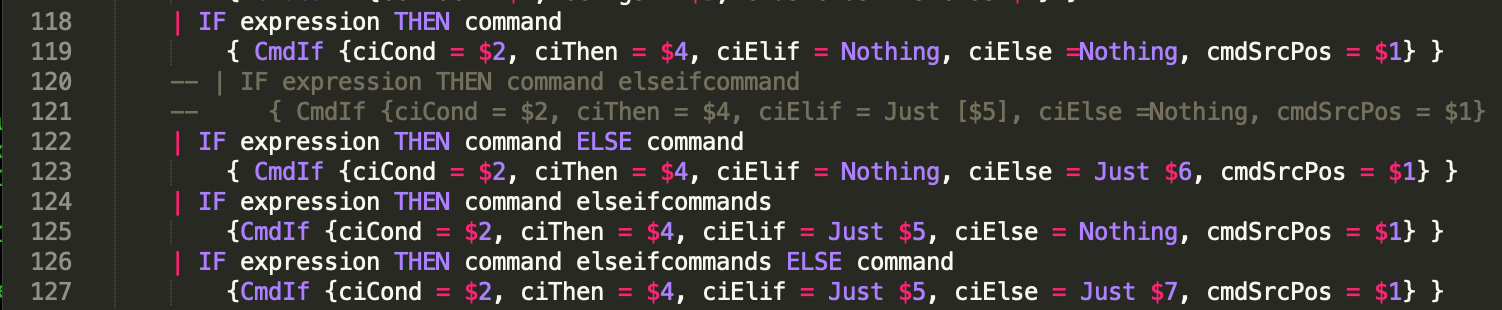
Two CmdAssign have same indent.

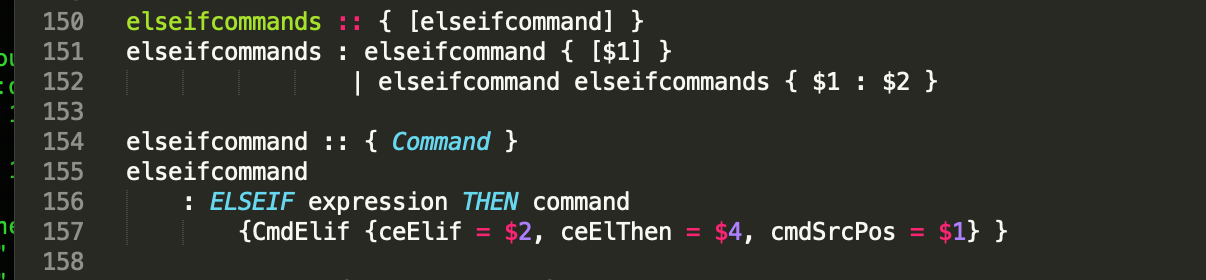
Scanner.hs:



Simply add one token to it like task I.1

Parser.y:



should add extra command to it just follow the extended grammar.

Task I.4

This part should focus on extend the scanner, make scanner can identify the new grammar.

First extend grammar:

Lexical Syntax:

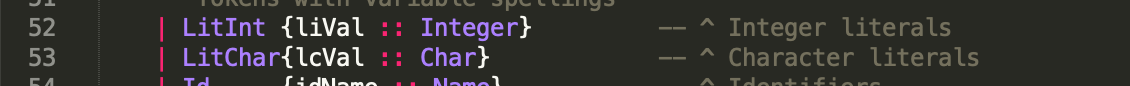
CharacterLiteral -> Graphic | EscapeCharacter

Graphic -> non-contril character except ‘ and \

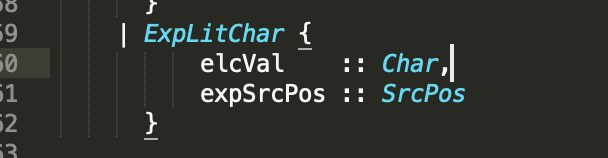
EscapeCharecter -> \ (n | r | t | \ | ’)

Coding part:

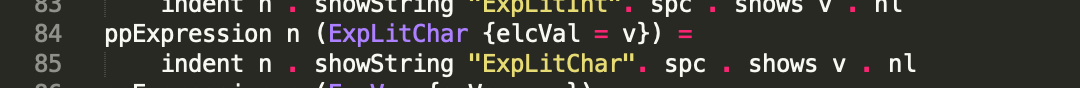
Token.hs:



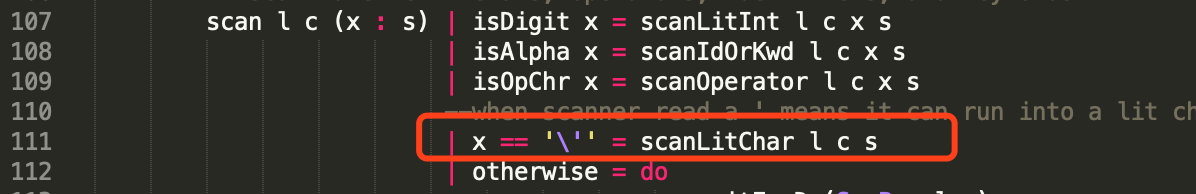
AST.hs:



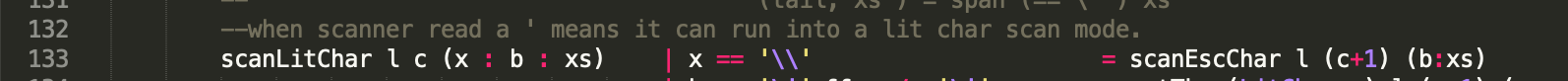
PPAST.hs:

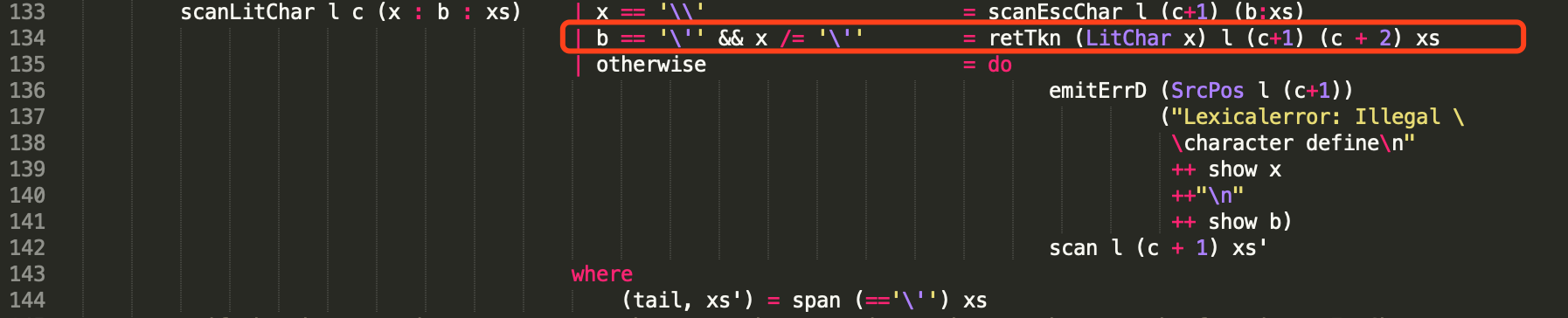


Scanner.hs:

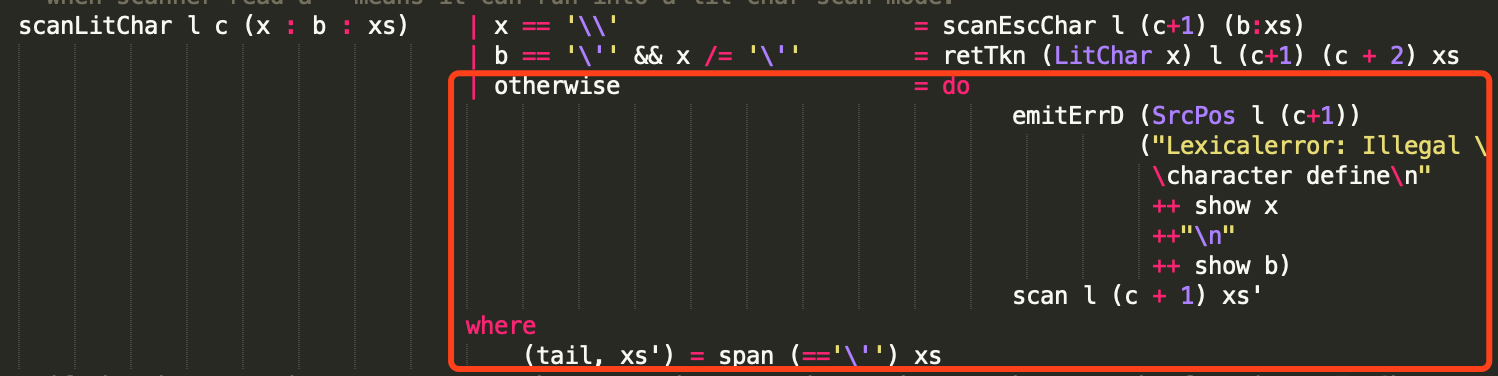


When reach a ‘ start scanlitchar



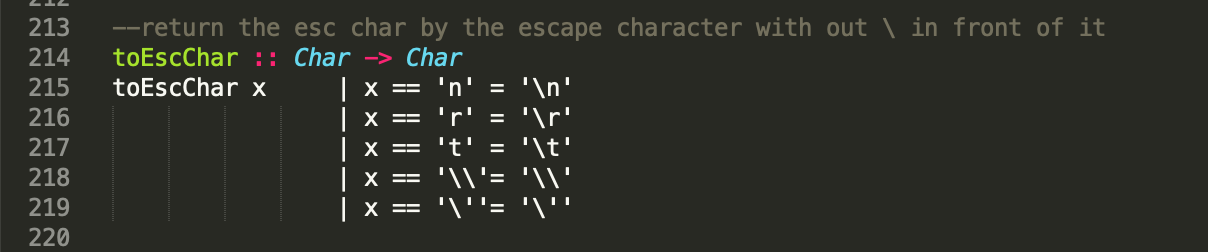
When reach a \ start scanescchar

Return literal character with source position.

print error message properly.



When reach escape character return litchar, using a helper function toEscChar for change the single escape character to full escape charcter etc. n -> \n



Parser.y:



