Pyprog Language Writeup

Changes I made in part 2:

I modified the way the if statement works. By default, the if statement worked like this:

```
# <if> -> "if" <expr> "then" <stmt_list> "else" <stmt_list> "end"
if x > y
then
    print x;
else
    print y;
end;
```

I added an option for an if statement without an "else":

```
# <if> -> "if" <expr> "then" <stmt_list> "else" <stmt_list> "end"
#         |"if" <expr> "then" <stmt_list> "end"
if x == y
then
    print x + y;
end;
```

For the parser, this meant checking for "end" as well as "else" after evaluating the statement list under "then."

No changes were made to the lexer.

Problems:

Sometimes, the line number for error messages will be wrong. This is because for some reason, the lexer is incrementing the line more than it should when it is only supposed to call next_line() if the current lexeme is '\n'. I tried running through lex() step by step, and one cause I ran into is in the situation where your program is one line long, for example:

```
print x
```

When the lexer gets to the '\n' at the end of the line, it calls next_line() which increments line from 1 to 2, so then when the parser realizes there is no semi colon and calls error(), it prints the line number as 2 even though there is only 1 line in the program.

I think there needs to be a change to the if statement that tells the lexer to call next_line(). If I had the time to resolve this issue, I would find situations in which the line number was wrong and then use break points to step through the code line by line and determine what changes need to be made to the lexer in order for the line number to be updated accurately.