Lexer implementation

- Will be implementing a very basic lexer
- Iterate through source code, character by character
  - Character options: LPAREN, RPAREN, DIGIT, OPPLUS, OPMINUS, OPMUL, OPDIV
  - Ignore whitespace (tabs, spaces, newlines)
  - If character is a digit, then combine with adjacent characters to form multi-digit number as a single lexeme

Lexer code

```
def lex(source_code):
    lexemes = []
    i = 0
    while i < len(source_code):
        # returns next non-whitespace index
        i = skip_whitespace(source_code, i)
        if source_code[i] in constants:
            lexemes.append(source_code[i])
        elif source_code[i].isdigit():
            num = source_code[i]
            i += 1
            while i < len(source_code) and source_code[i].isdigit():
                num += source_code[i]
                i += 1
            lexemes.append(num)
            i -= 1 # undo last increment because now not pointing to integer
        else:
            raise SyntaxError("Unknown character: {}".format(source_code[i]))
        i += 1
    return lexemes
```

```
def skip_whitespace(s, idx):
    while s[idx] in ' \t\r\n':
        idx += 1
    return idx
```

```
LPAREN = r"("
RPAREN = r")"
OPPLUS = r"+"
OPMINUS = r"-"
OPMUL = r"*"
OPDIV = r"/"

constants = [
    LPAREN, RPAREN,
    OPPLUS, OPMINUS,
    OPMUL, OPDIV
]
```