LPCC Assignment 1-B

Name: Digvijay Pawar
Class: T.Y Btech Comp B
GR no.: 21810344
Roll no.: 322043

Aim: Generate Symbol table, Literal table, Pool table & Intermediate code of a two-pass Assembler for the given source code.
1-b: Generate literal table from given assembly code.

Objective:

1. To generate literal table
2. To understand the working of two-pass Assembler

Theory:

Literal Table:

- A literal table is created for the literals which are used in the program.
- The literal table contains the literal name, operand value and length.
- The literal table is usually created as a hash table on the literal name.

Program:

1B.py :

```
import pandas as pd
import re

tfile = open('Task.txt','r')
literal = dict()
var1 = list()
symbol = dict()
LocCount = 0

re_lit = re.compile(r'=[0-9]')
```

```
for line in tfile:
    line.strip()
    words = line.split()

    if line.startswith('START'):
        LocCount = int(words[-1])
        continue

    if len(words)>3 :
        symbol[str(words[0])] = LocCount

    if 'DC' in line:
        symbol[str(words[0])] = LocCount

    if re_lit.search(line):
        var1.append(str(words[-1]))
        literal[str(words[-1])] = 0

    if line.startswith('END'):
        for w in var1:
            if literal.get(w)==0:
                literal[w] = LocCount
                LocCount += 1

    if 'DS' in line:
        LocCount += int(words[-1])
        symbol[str(words[0])] = LocCount
        continue

    if line.startswith('ORIGIN'):
        sub = words[-1].split('+')
        LocCount = symbol[str(sub[0])] + int(sub[1])
        continue

    if 'EQU' in line:
        if words[0] not in symbol.keys():
            symbol[str(words[0])] = symbol[str(words[-1])]

    if 'LTORG' in line:
        for w in var1:
            literal[w] = LocCount
            LocCount += 1

    LocCount += 1
```

```
literal_tb = pd.DataFrame(list(literal.items()),columns=['Literal','Address'])
print(literal_tb)
```

Input File :

Task.txt:

```
START 200
MOVER AREG =6
MOVER BREG X
L1 MOVER BREG =2
LTORG
NEXT ADD AREG =3
X DS 1
END
```

Output: