Name: - Dev Chhawachharia
Roll Number:- 322014
PRN:- 22010381
TY B

Assignment 2

AIM: - Implement second pass of a two-pass Assembler and generate machine language code for the given intermediate code.

Source code:

```
import pandas as pd

emot_table = [['STOP', '01', '00'],
              ['ADD', '01', '01'],
              ['SUB', '01', '02'],
              ['MULT', '01', '03'],
              ['MOVER', '01', '04'],
              ['MOVEM', '01', '05'],
              ['COMP', '01', '06'],
              ['BC', '01', '07'],
              ['DIV', '01', '08'],
              ['READ', '01', '09'],
              ['PRINT', '01', '10'],
              ['START', '03', '01'],
              ['END', '03', '02'],
              ['ORIGIN', '03', '03'],
              ['EQU', '03', '04'],
              ['LTORG', '03', '05'],
              ['DS', '02', '01'],
              ['DC', '02', '02'],
              ['AREG', '04', '01'],
              ['BREG', '04', '02'],
              ['CREG', '04', '03'],
              ['EQ', '05', '01'],
              ['LT', '05', '02'],
              ['GT', '05', '03'],
              ['NE', '05', '04'],
              ['LE', '05', '05'],
              ['GT', '05', '06'],
              ['ANY', '05', '07']]
emot_table_df = pd.DataFrame(emot_table, columns=['Mnemonic', 'Class',
'Opcode'])
# print(emot_table_df)

def check(str1):
    pass


machine_code = pd.DataFrame(columns=['LC', 'MCODE', 'OP1', 'OP2'])
```

```
f2 = open("symbol_table.txt",mode='rt')
symbol_table_contents = f2.read().split()
symbols=[symbol_table_contents[2*i] for i in
range(int(len(symbol_table_contents)/2))]
address = [symbol_table_contents[2*i+1] for i in
range(int(len(symbol_table_contents)/2))]
# print(symbols,address)

f3 = open("literal_table.txt",mode='rt')
literal_table_contents = f3.read().split()
lit_num = [literal_table_contents[3*i] for i in
range(int(len(literal_table_contents)/3))]
lit_value = [literal_table_contents[3*i+1] for i in
range(int(len(literal_table_contents)/3))]
lit_add = [literal_table_contents[3*i+2] for i in
range(int(len(literal_table_contents)/3))]
# print(lit_num,lit_value,lit_add)

f1 = open('intermediate_code.txt')
temp = f1.readline().split()[5].split()
ind = temp[0].index(')')
lc = int(temp[0][3:ind])
id=0
# print(lc)
for x in f1:
    line = x.split()
    # print((line))
    entry1 = lc
    mnemonic = line[1]
    code = line[3]
    # print(code)
    entry2 = code
    if mnemonic == 'AD':
        entry2=00
        entry3=00
        if code == '02':
            leng = len(lit_value)
            for i in range(leng):
                entry1 = lc
                entry4 = lit_value[i]
                if i!=leng-2:
                    lc+=1
        # if code == '05':
        #     pass
    if mnemonic == 'IS' or mnemonic== 'DL':
        if mnemonic == 'IS':
            if code == '00':
                entry3= 00
                entry4 = 00
            reg = line[5]
            opcode = emot_table_df.loc[emot_table_df['Mnemonic'] == reg,
'Opcode'].values[0]
            entry3 = opcode
        token = line[len(line)-1]
        # print(token[0])
        if mnemonic == 'DL':
            entry3 = 00
            if code == '01':
                inc = token[3]
                for i in range(int(inc)):
```

```
                    entry1 = lc
                    entry2 = 00
                    entry3 = 00
                    entry4 = 00
                    if i!=int(inc)-1:
                        lc+=1
                    entry = [entry1,entry2,entry3,entry4]
                    machine_code.loc[id] = entry
                    id+=1
        if token[0] != 'C' and token[0] != 'L' and token.isalpha():
            entry4 = address[symbols.index(token)]
            # print(entry4)
        elif token[1]=='C' and (mnemonic!= 'Dl' and code != '01'):
            entry4 = token[3]
            # print(entry4)
        elif token[1]=='L':
            entry4 = lit_add[lit_num.index(token[3])]
            # print(entry4)
    lc=lc+1
    # print(entry1,entry2,entry3,entry4)
    entry = [entry1, entry2, entry3, entry4]
    machine_code.loc[id] = entry
    id += 1
# print(lc)


f4=open("machine_code.txt",mode="wt")
machine_code = machine_code.drop_duplicates()
dfasString =machine_code.to_string(index = False)
f4.write(dfasString)

# print(machine_code)
```

Outputs: -

Input:

```
START 200
MOVER AREG X
ADD AREG '=3'
MOVEM AREG Y
X DC 1
Y DS 1
END
```

Machine Code:

LC	MCODE	OP1	OP2
200	04	01	203
201	01	01	205
202	05	01	204
203	02	0	1
204	0	0	0
205	0	0	'=3'

Note:- Not considered advanced assembler directives.