LPCC Assignment 2-c

Name: Digvijay Pawar
Class: TY.Btech Comp B2
Gr.No: 21810344
Roll No: 322043

Aim: Design suitable data structures & implement pass-I for a nested macro.

Objective: To understand concepts of Nested macro and Nested macro call.

Theory: Writing a macro is another way of ensuring modular programming in assembly language. A
macro is a sequence of instructions, assigned by a name and could be used anywhere in the program. In NASM, macros are defined with %macro and %endmacro directives.

Code:

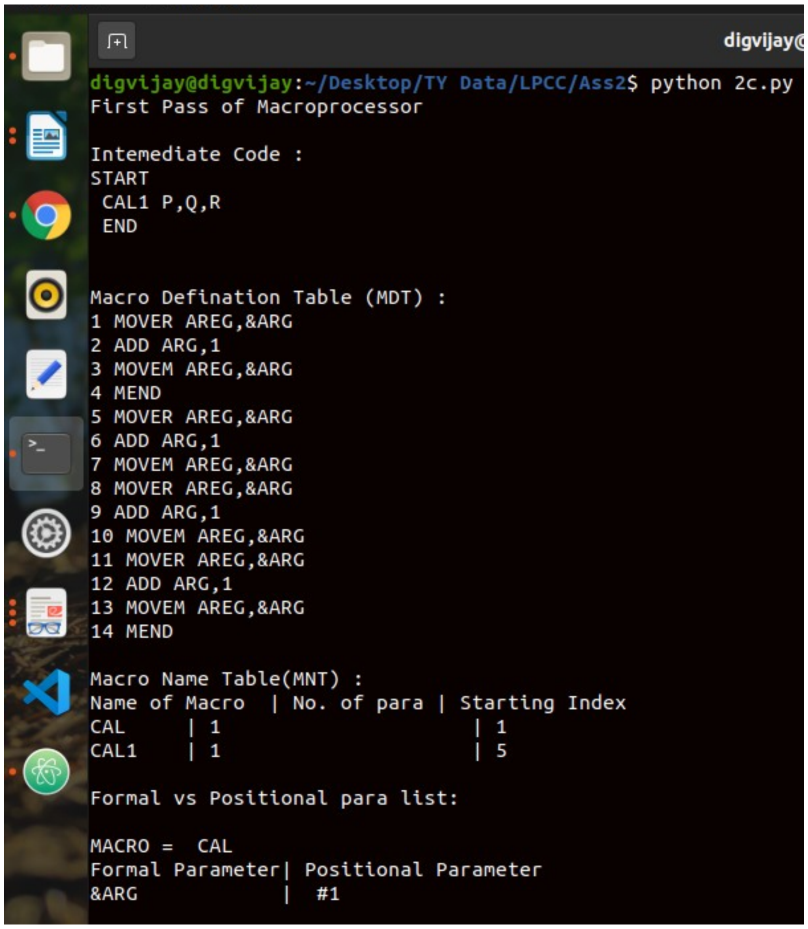
2C.py :

```
fhand = open('input3.txt', 'r')

curr_mac = "NULL"
code = {}
para = {}
output = []

for line in fhand:
    line.strip()
    dup_line = line
    words=line.split()

    if words[0] == "MACRO":
        curr_mac = words[1]
        param = []
        for y in words[2:]:
            param.append(y)
        code[words[1]] = []
        para[words[1]] = param
    elif words[0]!="MACRO" and curr_mac=="NULL":
        output.append(dup_line)
    elif words[0] == "MEND":
        code[curr_mac].append(words)
        curr_mac = "NULL"
    elif words[0] != "MACRO" and curr_mac != "NULL":
        code[curr_mac].append(words)

mdt = []
start = {}
i = 1
actual_pram = {}
def MACRO_expansion(key,lst):
    global i,actual_pram
    values = {}
    k = 0
    for y in para[key]:
        values[y] = lst[k]
        k = k + 1
    for x in code[key]:
        if x[0] not in code.keys() and x[0] != "MEND":
            n = 0
            st1 = x[:]
            for element in st1:
                if element in para[key]:
                    st1[n] = values[element]
                n = n + 1
            temp = [i,st1]
            mdt.append(temp)
            i = i+1
        elif x[0] in code.keys():
            temp = []
            for y in x[1:]:
                temp.append(y)
            if x[0] not in actual_pram.keys():
                actual_pram[x[0]] = []
            actual_pram[x[0]].append(temp)
            MACRO_expansion(x[0],temp)

for key in code.keys():
    loop = 1
    values = {}
    for x in para[key]:
        values[x] = "#" + str(loop)
        loop = loop+1
    start[key] = i
    for x in code[key]:
        if x[0] not in code.keys():
            n = 0
            stmt = x[:]
            for element in stmt:
                if element in para[key]:
                    stmt[n] = values[element]
                n = n + 1
            temp = [i,stmt]
            mdt.append(temp)
            i = i + 1
        elif x[0] in code.keys():
            temp = []
            for y in x[1:]:
                temp.append(y)
            if x[0] not in actual_pram.keys():
                actual_pram[x[0]] = []
            actual_pram[x[0]].append(temp)
            MACRO_expansion(x[0],temp)

for line in output:
    line = line.replace(","," ")
    words = line.split()
    if words[0] in para.keys():
        temp = []
        for y in words[1:]:
            temp.append(y)
        if words[0] not in actual_pram.keys():
            actual_pram[words[0]] = []
        actual_pram[words[0]].append(temp)

print("First Pass of Macroprocessor")
print()
print("Intemediate Code : ")
for x in output:
    print(x, end=" ")
print()
print("\nMacro Defination Table (MDT) : ")
for x in mdt:
    print(x[0],end = " ")
    for y in x[1]:
        print(y,end = " ")
    print()
print()
print("Macro Name Table(MNT) : ")
print("Name of Macro  | No. of para | Starting Index")
for x in para.keys():
    print(x,"\t|",len(para[x]),"\t\t\t|",start[x])

print("\nFormal vs Positional para list: \n")
for key in para.keys():
    if len(para[key]) > 0:
        print("MACRO = ",key)
        print("Formal Parameter| Positional Parameter")
        k = 1
        for x in para[key]:
            print(x,"\t\t| ","#"+str(k))
            k = k + 1
        print()
print("\nActual vs Positional para list: \n")
for key in actual_pram.keys():
    if len(para[key]) > 0:
        print("MACRO = ",key)
        for x in actual_pram[key]:
            k = 1
            print("Actual Parameter| Positional Parameter")
            for element in x:
                print(element,"\t\t| ","#"+str(k))
                k = k + 1
            print()

fhand.close()
```

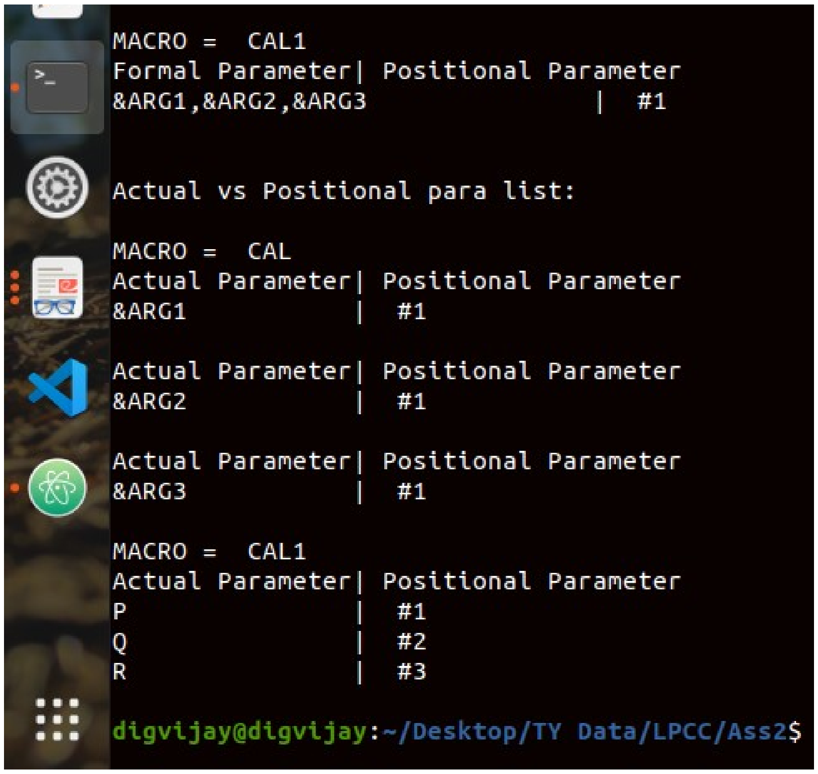
task.txt:

```
START
MACRO CAL &ARG
MOVER AREG,&ARG
ADD ARG,1
MOVEM AREG,&ARG
MEND
MACRO CAL1 &ARG1,&ARG2,&ARG3
CAL &ARG1
CAL &ARG2
CAL &ARG3
MEND
CAL1 P,Q,R
END
```

Output:



```
digvijay@digvijay:~/Desktop/TY Data/LPCC/Ass2$ python 2c.py
First Pass of Macroprocessor

Intemediate Code :
START
 CAL1 P,Q,R
 END


Macro Defination Table (MDT) :
1 MOVER AREG,&ARG
2 ADD ARG,1
3 MOVEM AREG,&ARG
4 MEND
5 MOVER AREG,&ARG
6 ADD ARG,1
7 MOVEM AREG,&ARG
8 MOVER AREG,&ARG
9 ADD ARG,1
10 MOVEM AREG,&ARG
11 MOVER AREG,&ARG
12 ADD ARG,1
13 MOVEM AREG,&ARG
14 MEND

Macro Name Table(MNT) :
Name of Macro   | No. of para | Starting Index
CAL      | 1                        | 1
CAL1     | 1                        | 5

Formal vs Positional para list:

MACRO =  CAL
Formal Parameter| Positional Parameter
&ARG               |  #1
```



```
MACRO =  CAL1
Formal Parameter| Positional Parameter
&ARG1,&ARG2,&ARG3                |  #1


Actual vs Positional para list:

MACRO =  CAL
Actual Parameter| Positional Parameter
&ARG1               |  #1

Actual Parameter| Positional Parameter
&ARG2               |  #1

Actual Parameter| Positional Parameter
&ARG3               |  #1

MACRO =  CAL1
Actual Parameter| Positional Parameter
P                   |  #1
Q                   |  #2
R                   |  #3

digvijay@digvijay:~/Desktop/TY Data/LPCC/Ass2$
```