**Name:** Atharva Jagtap

**Roll No:** 312018

**PRN No:**22111311

**Div:**B

**Experiment No. 03:**

# Design suitable data structures & implement first pass of a two-pass Macro processor

**Code:**

import re

MDTC = 1

MNTC = 1

ntHead = None

dtHead = None

alHead = None

alIndex = 1

class DefinitionTable:

def \_\_init\_\_(self):

self.index = 0

self.definition = ""

self.arg = []

self.next = None

class NameTable:

def \_\_init\_\_(self):

self.index = 0

self.name = ""

self.dtIndex = None

self.next = None

class ArgumentListArray:

def \_\_init\_\_(self):

self.index = 0

self.arg = ""

self.next = None

def findArgIndex(arg):

temp = alHead

while temp is not None:

if temp.arg == arg:

return temp

temp = temp.next

return None

def findName(name):

temp = ntHead

while temp is not None:

if temp.name == name:

return temp.dtIndex

temp = temp.next

return None

def pass1(input\_file):

global MDTC, MNTC, ntHead, dtHead, alHead, alIndex

lines = input\_file.readlines()

is\_processing\_macro\_def = False

for line in lines:

if "MACRO" in line:

tokens = re.split(r'\s+', line)

macro\_name = tokens[1]

print(f"MACRO {macro\_name} Detected...")

if ntHead is None:

ntHead = NameTable()

ntTemp = ntHead

else:

ntTemp.next = NameTable()

ntTemp = ntTemp.next

ntTemp.index = MNTC

MNTC += 1

ntTemp.name = macro\_name

print(f"{macro\_name} added into Name Table")

tokens = tokens[2:]

for token in tokens:

if token != "MACRO" and token != "\n":

if alHead is None:

alHead = ArgumentListArray()

alTemp = alHead

else:

alTemp.next = ArgumentListArray()

alTemp = alTemp.next

alTemp.index = alIndex

alIndex += 1

alTemp.arg = token

print(f"Argument {alTemp.arg} added into argument list array")

if dtHead is None:

dtHead = DefinitionTable()

dtTemp = dtHead

else:

dtTemp.next = DefinitionTable()

dtTemp = dtTemp.next

dtTemp.definition = macro\_name

dtTemp.arg = []

print(f"Definition table entry created for {macro\_name}")

ntTemp.dtIndex = dtTemp

is\_processing\_macro\_def = True

elif is\_processing\_macro\_def:

tokens = re.split(r'\s+', line)

isArg = 0

for token in tokens:

if isArg == 0:

if dtHead is None:

dtHead = DefinitionTable()

dtTemp = dtHead

else:

dtTemp.next = DefinitionTable()

dtTemp = dtTemp.next

dtTemp.index = MDTC

MDTC += 1

dtTemp.definition = token

print(f"Entry appended for {dtTemp.definition} at index {dtTemp.index}")

isArg = 1

else:

arg\_index = findArgIndex(token)

if arg\_index is None:

if alHead is None:

alHead = ArgumentListArray()

alTemp = alHead

else:

alTemp.next = ArgumentListArray()

alTemp = alTemp.next

alTemp.index = alIndex

alIndex += 1

alTemp.arg = token

else:

arg\_index = findArgIndex(token)

dtTemp.arg.append(arg\_index)

if "MEND" in line:

is\_processing\_macro\_def = False

def pass2(input\_file):

with open("output.txt", "a") as output\_file:

for line in input\_file:

print(line)

temp = findName(line.strip())

if temp is not None:

while temp.definition != "MEND":

output\_file.write(f"-\t{temp.definition}\t{temp.arg[0]}\t{temp.arg[1]}\n")

temp = temp.next

if \_\_name\_\_ == "\_\_main\_\_":

input\_file = open("input.asm", "r")

if input\_file is None:

print("Failed to open the assembly file!")

exit(0)

print("Pass 1 in progress")

pass1(input\_file)

input\_file.close()

input\_file = open("input.asm", "r")

input\_file.seek(0)

print("Pass 2 in progress")

pass2(input\_file)

input\_file.close()

print("Output with expanded code. is as above, Pass 2 Complete!")

**input.asm:**

INCR MACRO X

MOV R 1

ADD X R

MEND

.CODE

MOV AX, 5

INCR AX

CLC

END

**Output:**

A screenshot of a computer

Description automatically generated