Kundan Kabra

Roll Number:321023

COMP A1

GR: 21810729

**WEEK 3**

Aim: Generate Intermediate code along with previous ST,LT,PT.

CODE:

from tabulate import tabulate

def lit():

global countofstartpt

global count

global LC

global words

global printevt

global ltorgstartlc

wat = 0

temp2 = 1

temp = LC - 1

a = PTABLE[countofstartpt]

countofstartpt+=1

length = len(LTABLE)

for i in range(a,length):

if wat ==0:

printevt = printevt +"(DL,02) (," +list(LTABLE)[i] +")"

dicto[count]=[" ",words,ltorgstartlc,printevt]

ltorgstartlc+=1

wat+=1

else:

temp1= float(str(count) +"."+ str(temp2))

dicto[temp1]=[" "," ",ltorgstartlc,"(DL,02) (C," +list(LTABLE)[i] +")"]

ltorgstartlc+=1

temp+=1

temp2+=1

count+=1

IS = ["STOP","ADD","SUB","MULT",'MOVER',"MOVEM","COMP","BC","DIV","READ","PRINT"]

AD = ["START","END","ORIGIN","EQU","LTORG"]

DL = ["DS","DC"]

RG = ["AREG","BREG","CREG","DREG"]

CC = ["EQ","LT","GT","NE","ANY"]

file = open('asm3.txt', 'r')

Lines = []

STABLE = {}

LTABLE = {}

done = []

PTABLE = {0 : 0}

checked = []

ltablestartitems = []

dsval = 0

ST=0

s = 0

l =0

ok = 0

countofstartpt = 0

watched = 0

same\_pot = 1

pool\_id = 0

count = 1

dicto = {}

ltorgstartlc = 0

for each in file:

each = each.replace(","," ")

each = each.replace("+"," +")

Lines.append(each)

startloc = Lines[0].split()

LC = int(startloc[1]) -2

for j in Lines :

printevt = ""

same\_pot1 = 0

if same\_pot == 2:

LC = LC - 1

same\_pot1 = 1

same\_pot = 1

watched=0

LC=LC+1

split1 = j.split()

leng = len(split1)

words = " "

words0 = " "

for k in range(leng) :

if split1[k] in AD :

words = words +" "+ (split1[k])

x = "(AD,0"+ str(AD.index(split1[k])+1) + ")"

printevt = printevt + x

if split1[k] == "LTORG":

ltablestartitems.append(len(LTABLE))

ltorgstartlc = LC

ok = 0

pool = 0

for key,value in LTABLE.items():

pool += 1

if(ok>0):

LC= LC + 1

if value == 99:

LTABLE[key]=LC

ok += 1

PTABLE[pool\_id]= ( len(LTABLE) - ok )

pool\_id += 1

elif split1[k] == "ORIGIN":

same\_pot=same\_pot+1

elif split1[k] == "END":

LC=LC-1

ok = 0

pool = 0

ltablecontains99 = []

for key,value in LTABLE.items():

ltablecontains99.append(value)

pool += 1

if value == 99:

LC=LC+1

ok+=1

LTABLE[key]=LC

if 99 in ltablecontains99:

ltorgstartlc = LC - ok + 1

PTABLE[pool\_id]= ( len(LTABLE) - ok )

pool\_id += 1

elif split1[k] in IS:

words = words +" "+ (split1[k])

x = "(IS,0"+ str(IS.index(split1[k])) + ")"

printevt = printevt + x

elif split1[k] in DL :

STABLE[split1[k-1]][1]=split1[k+1]

words = words +" "+ (split1[k])

x = "(DL,0"+ str(DL.index(split1[k]) +1) + ")"

printevt = printevt + x

elif split1[k] in RG :

words = words +" "+ (split1[k])

x = "(RG,0"+ str(RG.index(split1[k])+1) + ")"

printevt = printevt + x

elif split1[k] in CC :

words = words +" "+ (split1[k])

x = "(CC,0"+ str(CC.index(split1[k])+1) + ")"

printevt = printevt + x

else :

bin = 0

if ( split1[k].startswith("'") | split1[k].startswith("=")) :

words = words +" "+ (split1[k])

printevt = printevt + "(L,"+str(l)+")"

l+=1

if split1[k] in LTABLE.keys():

while(split1[k] in LTABLE.keys()):

split1[k]=':'+ split1[k]

LTABLE[split1[k]]= 99

else :

LTABLE[split1[k]]= 99

elif (split1[k].startswith("+")):

words = words +" "+ (split1[k])

c = split1[k].replace("+","+ ")

temp = c.split()

for key,value in STABLE.items():

if( key == split1[k-1] ):

LC = value[0] + int(temp[1])

printevt = printevt + "(C,"+str(LC)+")"

elif split1[k].isdigit():

printevt = printevt + " (C,"+(split1[k]) +")"

if split1[k-1] == "ORIGIN":

LC = int(split1[k])

if split1[k-1] in DL :

LC = LC + int(split1[k]) - 1

words = words +" "+ (split1[k])

elif(watched==0) :

done.append(split1[k])

words0 = split1[k]

STABLE[split1[k]]=[LC,1]

ST+=1

s+=1

elif split1[k] not in done :

STABLE[split1[k]]=[LC,1]

printevt = printevt + " (S,"+str(s)+")"

words = words +" "+ (split1[k])

s+=1

else:

words = words +" "+ (split1[k])

watched=1

if words.\_\_contains\_\_("START"):

dicto[count]=["",words," ",printevt]

count+=1

elif words.\_\_contains\_\_("LTORG"):

lit()

elif words.\_\_contains\_\_("END") :

lit()

else:

dicto[count]=[words0,words,LC,printevt]

count+=1

tab = [[ST ,LC]]

headers = [" "," ","Assembly program","LC","Intermediate code"]

print(tabulate([[k,] + v for k,v in dicto.items()], headers = headers,tablefmt="fancy\_grid"))

headers = ["VALUE OF ST", "VALUE OF LC"]

print(tabulate(tab,headers=headers,tablefmt="fancy\_grid"))

headers = ["Symbol", "LC", "Size"]

print(tabulate([[k,] + v for k,v in STABLE.items()], headers = headers,tablefmt="fancy\_grid"))

headers = ["Literal", "LC"]

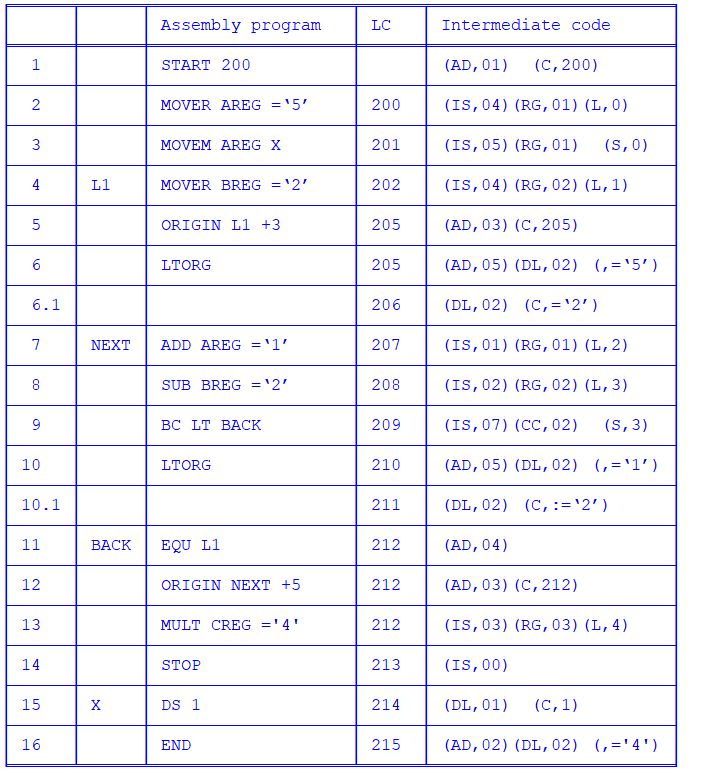
print(tabulate(LTABLE.items(), headers = headers,tablefmt="fancy\_grid"))

headers = ["ID", "Literal\_Number"]

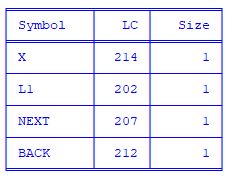
print(tabulate(PTABLE.items(), headers = headers,tablefmt="fancy\_grid"))

**OUTPUT:**

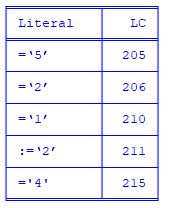
IntermediateCode:



SymbolTable:



LiteralTable:



PoolTable:

