(C) 1. Write a program to generate Symbol table of a two-pass Assembler for the given Assembly

language source code.

INPUT/CODE

START 180

READ M

READ N

LOOP MOVER AREG, M

MOVER BREG, N

COMP BREG, =’200’

BC GT, LOOP

BACK SUB AREG, M

COMP AREG, =’500’

BC LT, BACK

STOP

M DS 1

N DS 1

END  
  
  
def gen\_sym\_table(assym\_code):

sym\_table = {}

address = 0

reserved\_mnemonics = {

'STOP': ('00', 'IS', 0),

'ADD': ('01', 'IS', 2),

'SUB': ('02', 'IS', 2),

'MUL': ('03', 'IS', 2),

'MOVER': ('04', 'IS', 2),

'MOVEM': ('05', 'IS', 2),

'COMP': ('06', 'IS', 2),

'BC': ('07', 'IS', 2),

'DIV': ('08', 'IS', 2),

'READ': ('09', 'IS', 1),

'PRINT': ('10', 'IS', 1),

'START': ('01', 'AD', 1),

'LTORG': ('05', 'AD', 0),

'ORIGIN': ('03', 'AD', 1),

'EQU': ('04', 'AD', 2),

'END': ('AD', 0),

'DS': ('01', 'DL', 1),

'DC': ('02', 'DL', 1),

}

lines = assym\_code.split('\n')

for line in lines:

tokens = line.split()

if len(tokens) == 0:

continue

if tokens[0] in reserved\_mnemonics:

if tokens[0] == 'START':

address = int(tokens[1])

elif tokens[0] == 'END':

break

elif tokens[0] == 'LTORG' or tokens[0] == 'ORIGIN' or tokens[0] == 'EQU':

continue

else:

address += 1

else:

sym\_table[tokens[0]] = address

address += 1

# update the address for the data storage

for line in lines:

tokens = line.split()

if len(tokens) == 0:

continue

if tokens[0] == 'DS':

sym\_table[tokens[1]] = address

address += int(tokens[2])

return sym\_table

asem\_code = """

START 180

READ M

READ N

LOOP MOVER AREG, M

MOVER BREG, N

COMP BREG, ='200'

BC GT, LOOP

BACK SUB AREG, M

COMP AREG, ='500'

BC LT, BACK

STOP

M DS 1

N DS 1

END

"""

sy\_table = gen\_sym\_table(asem\_code)

print("symbol table :")

for lable, address in sy\_table.items():

print(f"{lable}, {address}")