```
def IS(word):
  answer="("
  if(word=="STOP"):
    answer+=("IS,00)")
  elif(word=="ADD"):
    answer+=("IS,01)")
  elif(word=="SUB"):
    answer+=("IS,02)")
  elif(word=="MUL"):
    answer+=("IS,03)")
  elif(word=="MOVER"):
    answer+=("IS,04)")
  elif(word=="MOVEM"):
    answer+=("IS,05)")
  elif(word=="COMP"):
    answer+=("IS,06)")
  elif(word=="BC"):
    answer+=("IS,07)")
  elif(word=="DIV"):
    answer+=("IS,08)")
  elif(word=="READ"):
    answer+=("IS,09)")
  elif(word=="PRINT"):
    answer+=("IS,10)")
  else:
    return DL(word)
  return answer

def DL(word):
  answer="("
  if(word=="DS"):
    answer+=("DL,01)")
  elif(word=="DC"):
    answer+=("DL,02)")
  else:
    return ""
  return answer

def AD(word):
  answer="("
  if(word=="START"):
    answer+=("AD,01)")
  elif(word=="END"):
    answer+=("AD,01)")
  elif(word=="ORIGIN"):
    answer+=("AD,01)")
  elif(word=="EQU"):
    answer+=("AD,01)")
  elif(word=="LTORG"):
    answer+=("AD,01)")
  return answer

def RG(word):
  answer="(RG,"
  if(word=="AREG,"):
    answer+=("1)")
  elif(word=="BREG,"):
    answer+=("2)")
  elif(word=="CREG,"):
    answer+=("3)")
  else:
    return CC(word)
  return answer

def CC(word):
  answer=""
  if(word=="EQ,"):
    answer+=("(CC,1)")
  elif(word=="LT,"):
    answer+=("(CC,2)")
  elif(word=="GT,"):
    answer+=("(CC,3)")
  elif(word=="NE,"):
    answer+=("(CC,4)")
  elif(word=="LE,"):
    answer+=("(CC,5)")
```

```
    elif(word=="GE,"):
      answer+=("(CC,6)")
    elif(word=="ANY,"):
      answer+=("(CC,7)")

    return answer

  def IC(assembly_code):
    ST=[]
    LT=[]
    PT=0
    for line in assembly_code:
      parts=line.split()
      answer=""
      if(len(parts)==1):
          if(parts[0]=="STOP"):
            answer+=(IS(parts[0]))
          else:
            answer+=(AD(parts[0]))
      elif(len(parts)==2):
            if(parts[0]=="START"):
              answer+=(DL(parts[0]))
              answer+=("(C,")
              answer+=(parts[1])
              answer+=(")")
              ADD=int(parts[1])
            else:
              answer+=(IS(parts[0]))
              if parts[1] in ST:
                answer+=("(S,")
                answer+=(ST.index(parts[1]))
              else:
                answer+=("(")
                answer+=(parts[1])
                ST+=(parts[1])
                answer+=(")")
      elif(len(parts)==3):
           answer+=(IS(parts[0]))
           answer+=(RG(parts[1]))
           if(parts[2]=="AREG" or parts[2]=="BREG" or parts[2]=="CREG"):
            parts[2]+=","
            answer+=RG(parts[2])
           else:
            if(parts[2].startswith("='")):
             if( parts[2] in LT):
                answer+=("(L,")
                answer+=(LT.index(parts[2]))
                answer+=(")")
             else:
                LT+=(parts[2])
                PT+=1
                answer+=("(L,")
                answer+=(str(PT))
                answer+=(")")
            elif(parts[1]=="DS" or parts[1]=="DC"):
                answer+=("(S,")
                answer+=(str(ST.index(parts[0])))
                answer+=(")")
                answer+=(DL(parts[1]))
                answer+=("(C,")
                answer+=(parts[2])
                answer+=(")")
            else:
             if parts[2] in ST:
                answer+=("(S,")
                answer+=(str(ST.index(parts[2])))
                answer+=(")")
             else:
                answer+=("(")
                answer+=(parts[2])
                ST.append(parts[2])
                answer+=(")")

      elif(len(parts)==4):
           ST.append(parts[0])
           answer+=(IS(parts[1]))
           answer+=(RG(parts[2]))
           if(parts[3].startswith("='")):
```

```
            if(parts[3].startswith( =  )):
              if( parts[3] in LT):
                 answer+=("(L,")
                 answer+=(LT.index(parts[3]))
                 answer+=(")")
              else:
                 LT.append(parts[3])
                 PT+=1
                 answer+=("(L,")
                 answer+=(str(PT))
                 answer+=(")")
            else:
              if parts[3] in ST:
                 answer+=("(S,")
                 answer+=(str(ST.index(parts[3])))
              else:
                 answer+=("(S,")
                 answer+=(parts[3])
                 ST.append(parts[3])
            answer+=(")")
        print(answer)



    assembly_code = [
        "START 200",
        "READ X",
        "READ Y",
        "MOVER AREG, ='5'",
        "MOVER BREG, ='6'",
        "ADD AREG, BREG",
        "LOOP MOVER CREG, X",
        "ADD CREG, ='1'",
        "COMP CREG, Y",
        "BC LT, LOOP",
        "LTORG",
        "NEXT SUB AREG, ='1'",
        "COMP AREG, Y",
        "BC GT, NEXT",
        "STOP",
        "X DS 1",
        "Y DS 1",
        "END"
    ]

    # Generate literal table
    IC(assembly_code)
```

```
(C,200)
(IS,09)(X)
(IS,09)(Y)
(IS,04)(RG,1)(L,1)
(IS,04)(RG,2)(L,2)
(IS,01)(RG,1)(RG,2)
(IS,04)(RG,3)(S,0)
(IS,01)(RG,3)(L,3)
(IS,06)(RG,3)(S,1)
(IS,07)(CC,2)(S,2)
(AD,01)
(IS,02)(RG,1)(L,4))
(IS,06)(RG,1)(S,1)
(IS,07)(CC,3)(S,3)
(IS,00)
(S,0)(DL,01)(C,1)
(S,1)(DL,01)(C,1)
(AD,01)
```