H24081333

# -\*- coding: utf-8 -\*-

"""

Created on Mon Feb 22 20:28:36 2021

@author: Aaron Lin

"""

def multiply(poly\_1,poly\_2): #副程式呼叫

#poly\_1=poly\_n[i]

#poly\_2=poly\_n[i+1]

temp\_sign=[]

for j in range(len(poly\_1)):

for k in range(len(poly\_2)):

sign=0

if poly\_1[j][0]=="+" and poly\_2[k][0]=="+": #判斷+-號

sign+=1

if poly\_1[j][0]=="-" and poly\_2[k][0]=="-":

sign+=1

if poly\_1[j][0]=="+" and poly\_2[k][0]=="-":

sign-=1

if poly\_1[j][0]=="-" and poly\_2[k][0]=="+":

sign-=-1

temp\_sign.append(sign)

#print(temp\_sign) #各多項式的係數,第一輪相乘後出現000000? 注意datatype

multi=[] #save兩個()內poly相乘整理的結果

temp\_poly\_1=[]

print(poly\_1)

for j in range(len(poly\_1)):

space1=[]

space1+=27\*[0] #space

a1=poly\_1[j].find("\*")

a2=poly\_1[j].find("^")

#print(poly\_1[j][1:a1])

space1[0]=int(poly\_1[j][1:a1]) #係數

for k in range(len(poly\_1[j])):

if 65<=ord(poly\_1[j][k])<=90:

space1[ord(poly\_1[j][k])-64]=int(poly\_1[j][k+2])

temp\_poly\_1.append(space1)

#print(temp\_poly\_1)

temp\_poly\_2=[]

print(poly\_2)

for l in range(len(poly\_2)):

space2=[]

space2+=27\*[0]

b1=poly\_2[l].find("\*")

b2=poly\_2[l].find("^")

space2[0]=int(poly\_2[l][1:b1])

for m in range(len(poly\_2[l])):

if 65<=ord(poly\_2[l][m])<=90:

space2[ord(poly\_2[l][m])-64]=int(poly\_2[l][m+2])

temp\_poly\_2.append(space2)

#print(temp\_poly\_2)

n1=0 #處理係數的正負號

n=len(poly\_1)\*len(poly\_2)

for i in range(len(temp\_poly\_1)):

for j in range(len(temp\_poly\_2)):

temp=[]

for k in range(27):

if k==0:

temp.append(temp\_poly\_1[i][k]\*temp\_poly\_2[j][k])

while n1<n:

if temp\_sign[n1]==-1: #係數有負號,記得要補

temp[0]=int("-"+str(temp[0]))

n1+=1

break

else:

n1+=1

break

else:

temp.append(temp\_poly\_1[i][k]+temp\_poly\_2[j][k])

#print(temp) #任兩個()中的string 相乘的結果

multi.append(temp)

print(multi) #二維list,len=poly\_1\*poly\*2,每個multi[i]有27個index(待合併)

multi\_str=[]

for i in range(len(multi)):

temp=""

if multi[i][0]>0:

temp+="+"

temp=temp+(str(multi[i][0]))+"\*"

for j in range(1,len(multi[i])):

if multi[i][j]!=0:

temp=temp+str(chr(j+64))+"^"+str(multi[i][j])

temp+="$"

multi\_str.append(temp)

return multi\_str

poly=input("Please input polynomial:")

poly=poly.strip("(")

poly=poly.strip(")")

poly=poly.split(")(")

print(poly)

i=0

poly\_n=[]

#poly\_n=[0 for i in range(len())]

#for \_ in range(len(poly)):

# poly\_n.append(0)

#print(poly\_n)

for i in range(len(poly)):

poly\_n.append(0) #獲得一個新空間

poly\_n[i]=poly[i].replace("+"," +")

poly\_n[i]=poly\_n[i].replace("-"," -")

poly\_n[i]=poly\_n[i].split(" ")

#print(poly\_n) #製造一個二維list

for i in range(len(poly\_n)):

for j in range(len(poly\_n[i])):

poly\_n[i][j]+="$" #終止字元

if 65<=ord(poly\_n[i][j][0])<=90: #出現A-Z

poly\_n[i][j]="+1\*"+poly\_n[i][j][0:]

if poly\_n[i][j][0]=="+" and 65<=ord(poly\_n[i][j][1])<=90:

poly\_n[i][j]="+1\*"+poly\_n[i][j][1:]

if poly\_n[i][j][0]=="-" and 65<=ord(poly\_n[i][j][1])<=90:

poly\_n[i][j]="-1\*"+poly\_n[i][j][1:]

if 48<=ord(poly\_n[i][j][0])<=57: #出現0-9常數

poly\_n[i][j]="+"+poly\_n[i][j]

if 65<=ord(poly\_n[i][j][-2])<=90:

poly\_n[i][j]=poly\_n[i][j][:-1]+"^1$"

for k in range(len(poly\_n[i][j])):

if 65<=ord(poly\_n[i][j][k])<=90 and 65<=ord(poly\_n[i][j][k+1])<=90:

poly\_n[i][j]=poly\_n[i][j][0:k+1]+"^1"+poly\_n[i][j][k+1:]

#print(poly\_n) #格式化

# \*=star ^=power $=dollar

for i in range(len(poly\_n)-1): #相乘的次數為總括號數-1

poly\_n[i+1]=multiply(poly\_n[i],poly\_n[i+1])

#做出multi後,得到的是各多項式相乘後的係數 以及A-Z的power,要把他重新整合成原本poly\_1(同項合併)

#這樣才能繼續用副程式計算

final\_poly=poly\_n[-1] #為包含多個不等長的string的list

print(final\_poly)

output=""

#output 需要符合:

#1.同變數且同次數可合併 (照原本list index的順序即可)

#2.不同變數之間的\*原本就已忽略

#3.power為1可省略

#4.不同項以+或-連接

for i in range(len(final\_poly)):

final\_poly[i]=final\_poly[i].split("\*")

print(final\_poly)

# 二維list

output=""

for i in range(len(final\_poly)):

temp\_str=final\_poly[i][1]

temp=[final\_poly[i][0],temp\_str]

n=i+1

while n<len(final\_poly):

if temp\_str==final\_poly[n][1]: #有重複的var

temp[0]=str(int(final\_poly[i][0])+int(final\_poly[n][0]))

if int(temp[0])>0:

temp[0]="+"+temp[0]

n=n+1

temp[1]=temp[1].replace("$","")

temp[1]=temp[1].replace("^1","")

if temp[0]=="+1":

temp[0]="+"

if temp[0]=="-1":

temp[0]="-"

if temp[0]=="+" or temp[0]=="-":

output+=temp[0]+temp[1]

else:

output+=temp[0]+"\*"+temp[1]

if output[0]=="+":

output=output[1:]

print("=====================================================================")

print("Output Result",":",output)

"""

if i==0:

temp[0]=temp[0].replace("+","")

if i>0:

temp[0]=temp[0].replace("+1","+")

temp[0]=temp[0].replace("-1","-")

"""

#output+=temp[0]+"\*"+temp[1]

#print(temp)

"""

for i in range(len(final\_poly)):

temp=[]

while n<len(final\_poly):

for j in range(i,len(final\_poly)):

if final\_poly[i][2]==final\_poly[j][2]:

final\_poly[i][1]=str(int(final\_poly[i][1])+int(final\_poly[j][1]))

del final\_poly[j]

print(final\_poly)

"""

"""

var=[] #測試變數

for i in range(len(final\_poly)):

temp=""

#for j in range(len(final\_poly[i])):

a=final\_poly[i].find("\*") #every string has only one "\*" and "$"

b=final\_poly[i].find("$") # so we can use it to test whether the variables are same

var.append(final\_poly[i][a:b])

sub="^1"

for i in range(len(var)):

if sub in var[i]:

var.remove(sub)

print(var) #尚有重複變數

"""

"""

final=[]

for i in range(len(var)):

if var[i] not in final:

final.append(var[i])

#print(0)

else:

#print(1)

n=final.find(var[i]) #重複變數的index

fianl[n]=str(int(fianl[n][0:2])+int(fianl\_poly[i][0:2]))+final[n][2:]

print(final)

"""

"""

"""

"""

#def delvar(var):

#new=[]

#for i in var:

#if i not in new:

#new.append(fianl\_poly[i][0:a])

#new.append(i)

#print(delvar(var))

#return new

"""

"""

"""

"""

for j in range(len(poly\_n[i])):

co=[]

var=[]

power=[]

a1=poly\_n[i][j].find("\*")

a2=poly\_n[i][j].find("^")

a3=poly\_n[i][j].find("$")

for k in range(len(poly\_n[i+1])): #poly\_n[i],poly\_n[i+1]長度不一定相同

co\_temp=[]

var\_temp=[]

power\_temp=[]

b1=poly\_n[i+1][k].find("\*")

b2=poly\_n[i+1][k].find("^")

b3=poly\_n[i+1][k].find("$")

c1=int(poly\_n[i][j][1:a1])\*int(poly\_n[i+1][k][1:b1]) #係數

c2=poly\_n[i][j][a1+1:a2]+poly\_n[i+1][k][b1+1:b2] #未知數

c3=int(poly\_n[i][j][a2+1:a3])+int(poly\_n[i+1][k][b2+1:b3]) #次方

co\_temp.append(c1)

var\_temp.append(c2)

power\_temp.append(c3)

print(co\_temp)

print(var\_temp)

print(power\_temp)

"""