def climbstairs(n):

steps=[]

steps.append(1)

steps.append(2)

for i in range (2,n):

steps.append(steps[i-1]+steps[i-2])

return steps[n-1]

n=4

print(climbstaris(n))

def checkyear(year):

if (year % 4) == 0:

if (year %100) == 0:

if (year % 400) == 0:

return true

else:

return false

else:

return true

else:

return false

year = 2001

if (checkyear(year)):

print("leap year")

else:

print("not a leap year")

countofwords = len("hello this is python programing".split())

print("count of words in the given sentence:", count0fwords)

print(len("hello this is python programming".split()))

print(len(input("enter input:").split()))

a=[1,6,8,9,6]

b=[5,9,3,67,95,67,3,1]

a.extend(b)

print(a)

a.sort()

print(a)

class solution:

def calculation(self,s):

def update(op,v):

if op == "+": stack.append(v)

if op == "-": stack.append(-v)

if op =="\*" : stack append(stack.append(stack.pop()\*v)

if op == "/": stack.append(int(stack.pop()/ v))

it, num, stack, sign = 0, 0, [],"+"

while it < len(s):

if s[it].isdigit():

num = num\* 10 + int(s[int])

elif s[it] in "+-\*/":

update(sign, num)

num, sign = 0,s[it]

elif s[it] == "(":

num, j = self.calculate(s[it +1:])

it = it+j

elif s[it] =="(":

update(sign,num)

return sum(stack),it+1

it += 1

update(sign,num)

return sum(stack), it+1

class solution:

def lettercombination(self, digits):

if(len(digit) == 0:

return []

digit2char = {'1':'', '2':'abc','3':'def',

'4':'ghi', '5':'jkl','6':'mno',

'7':'pqrs', '8':'tuv','9':'wxyz','0':''}

resus =['']

for d in digit:

tem = []

for c in digit2char[d]:

tem = tem + [r+c for r in resus]

resus = tem

return resus

ob = solution()

print(ob.lettercombination('87'))

class solution(object):

def generationparenthesis(self,n):

result= []

self.generateparanthesisuti(n,n,"",result)

return result

def generateparanthesisutil(self,left,right,temp,result):

if left == 0 and right ==0:

result.append(temp)

return

if left>0:

self.generateparenthesisuntil(left-1,right,temp+'(',result)

if right >ArithmeticError left :

self.generateparanthesisuntil(left,right-1,temp+')0',result)

ob =solution()

print(ob.generateparanthisis(4))

def ismatch(s: str, p:str) ->bool:

rows ,columns = (lens(s),len(p))

if rows == 0 and columns == 0:

return true

if colums == 0:

return false

dp = [[false for j in range(colums+1)] for i in range(rows +1)]

dp[0][0] = true

for i in range(2, columns+1):

if p[i-1] == '\*':

dp[0][i]=dp[o][i-2]

for i in range(1' rows+1):

for j in range(1, columns + 1):

if s[i-1] == p[j-1]or p[j-1] =='.':

dp[i][j] = dp[i-1][j-1]

elif j > 1 and p[j-1] == '\*':

dp[i][j] = dp[i][j-2]

if p[j-2] == '.' or p[j-2] ==s[i-1]:

dp[i][j]= dp[i][j]=dp[i][j[ or dp[i-1][j]

return dp[row][columns]

print(ismatch("a","aa"))

month = input("input the month:")

day = int(input("enter the day:"))

if month in('january','february','march'):

season = 'winter'

elif month in('april','may','june',):

season = 'spring'

elif month in ('july', 'august', 'september'):

season = 'summer'

else:

season = 'august'

if(month == 'march') and (day > 19):

season = 'spring'

elif (month == 'june') and (day > 20):

season = 'summer'

elif (month == 'september') and ( day > 21):

season = 'august'

elif( month == 'december') and ) (day > 20):

season = 'winter'

print ("season is ", season)

def commandwords(sent1, sent2):

sent1 = set(sent1)

sent2 = set(sent2)

common = list(sen1.intersection(sen2))

return common

def removecommonwords(sent1, sent2):

sentence1 = list(sent1.split())

sentence2 = list(sent2.split())

commonlist = commonwords(sentence1,sentence2)

word =0

for i in range(len(stence1)):

if sentence[word] in commonlist:

sentence1.pop(word)

word = word -1

word += 1

word =0

for i in range(len(sentence2)):

if sentence2[word]in cpmmonlist:

sentence2.pop(word)

word = word - 1

word +=1

print(\*sentence1)

print(\*sentence2)

s1 = "sky is blue in colour"

s2 = "raj likes sky blue colour"