Solving Procedural Programming Problems with Python

Emmanuel Ameh

**Python Code for filtering odd numbers**

#defining list as lst containing numbers 1 to 9

lst = [1,2,3,4,5,6,7,8,9]

#def filter(lst):

# iterating each number in list with for loop declaration

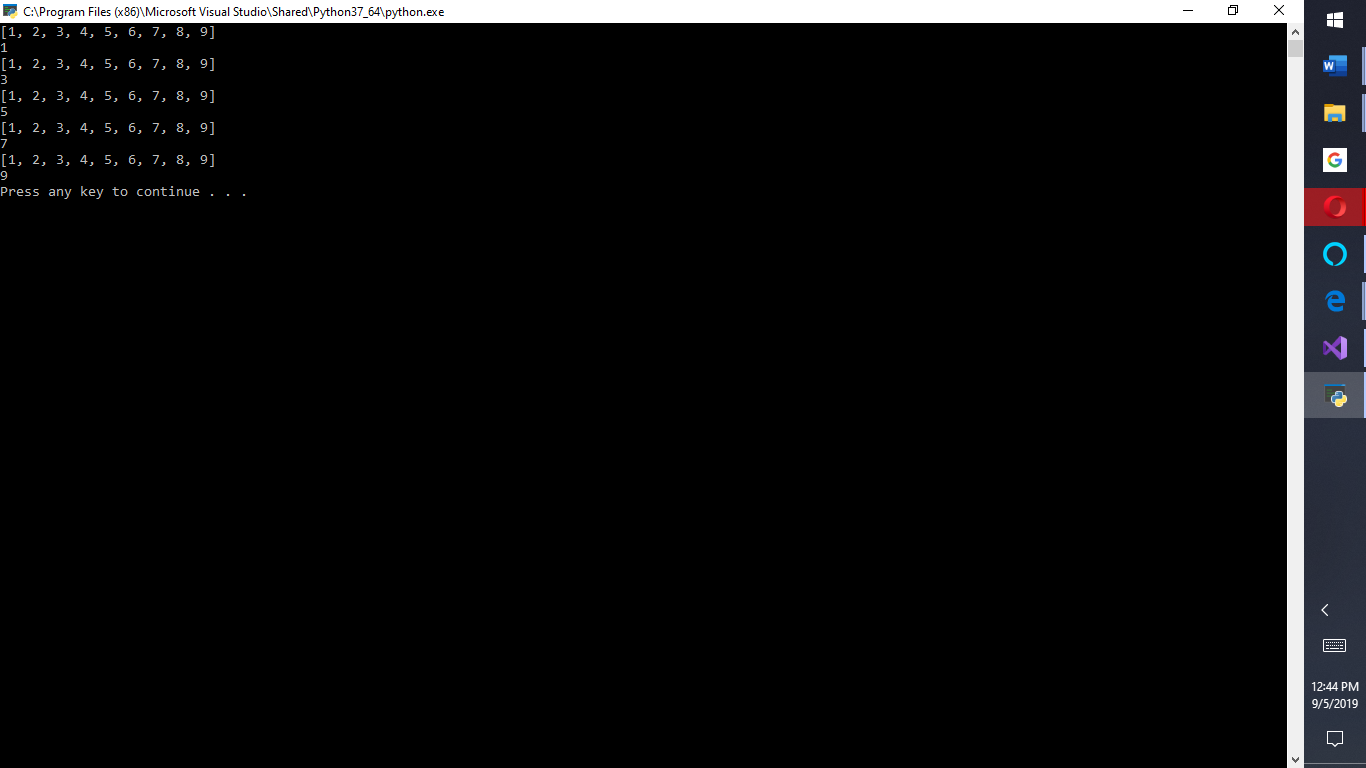
for num in lst:

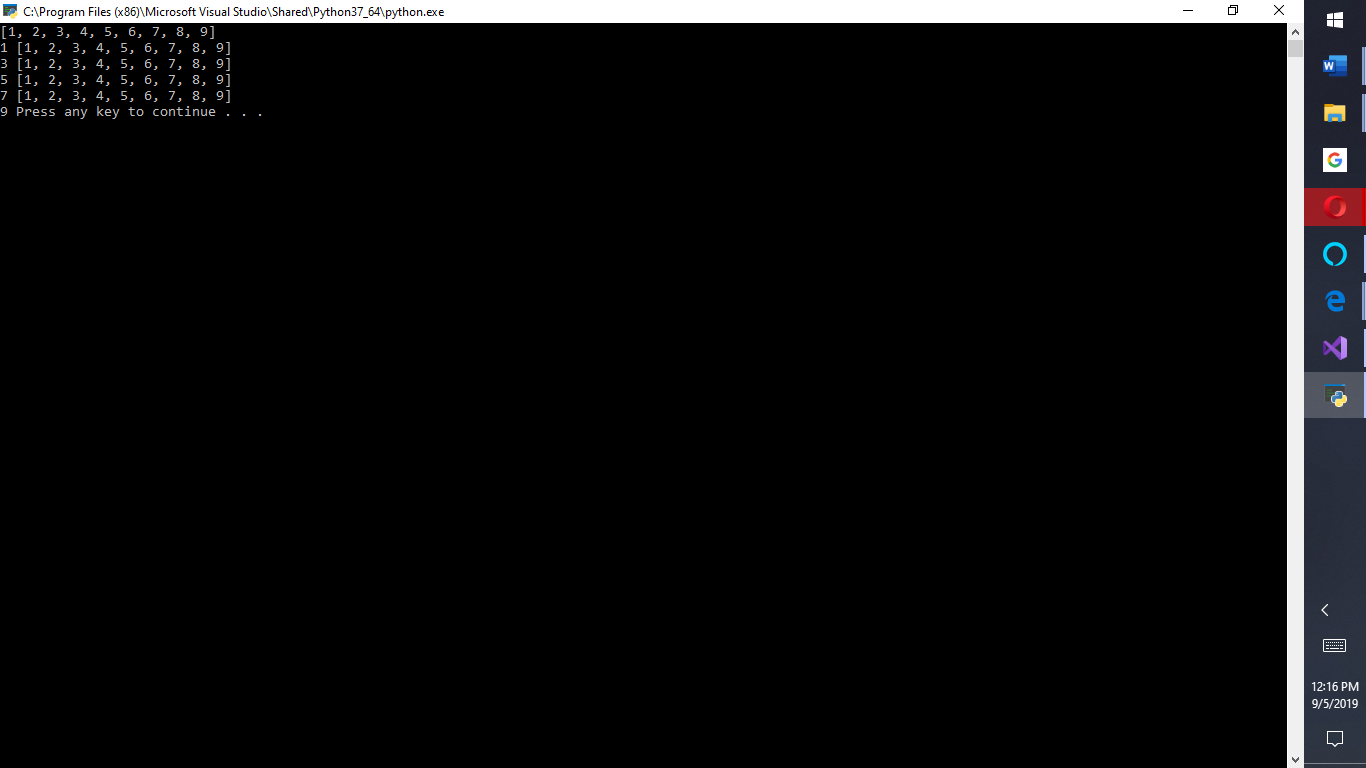
if num % 2 != 0:

#lst.append(num)

print(lst)

print(num, end = " ")



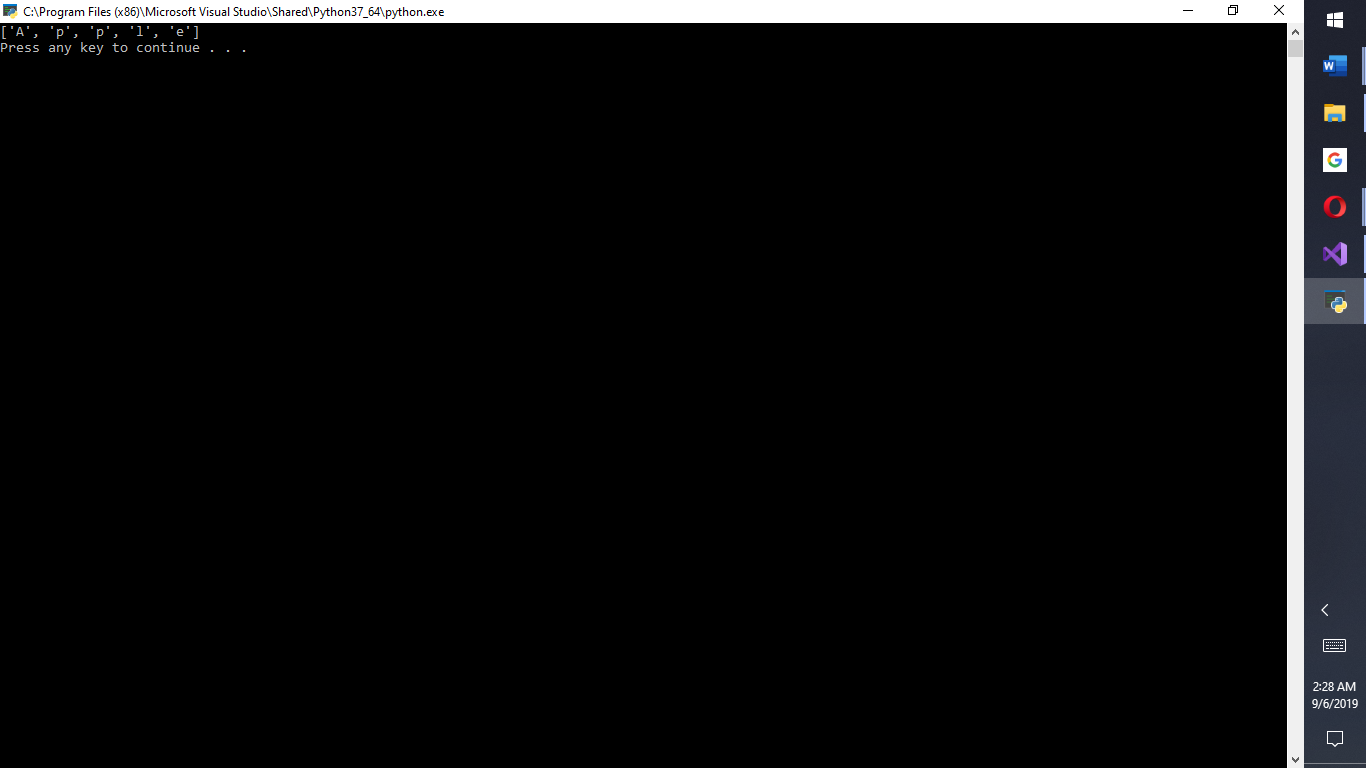


**Python Code for splitting letters in a word into a list of individual symbols**

word = 'Apple'

# Splitting at 1

print([word[i:i+1] for i in range(0, len(word), 1)])



**3.**

**Python code for joining letters into one string word**

# Demonstrating use of join function to join list

# elements without any separator.

# Joining with empty separator

list1 = ['a','p','p','l', 'e']

print("".join(list1))

list2 = ['N','C','U']

print("".join(list2))



**4. Python Code for Tribbles, an implementation of the Pascal’s triangle also implemented in Fibonacci**

# calculate one generation of tribbles with 12-hour intervals: index i, index j --> pas[i,j]

def calculatetribbles(i,j):

if (i == 0 or j == 0):

return 1

elif( i == j):

return 1

else:

return (calculatetribbles(i-1,j)+ calculatetribbles(i-1,j-1))

# calculate a given generation tribbles (equivalent of pascal triangle : number of row --> row as array

def calculaterow(row):

result = []

for i in range(row + 1):

result.append(calculatetribbles(row,i))

return result

# convert a given generation of tribbles to power of 11 : number of row -> intarray

def summrow(row):

trans = 0

calcrow = calculaterow(row)

length = len(calcrow)

for i in range((length)):

item = calcrow[length - i - 1] + trans

calcrow[length - i - 1] = (item % 10)

trans = item // 10

return calcrow

## further function: create number of tribbles as array: number of rows --> arr[][] of pascal

def arrpas(rows):

arr = [ [0 for i in range(j + 1)] for j in range(rows + 1)]

for k in range(rows + 1):

for l in range(k+1):

arr[k][l] = calculatetribbles(k,l)

return(arr)

## further function: create array of powers of 11: number of rows --> pow 11 as stinrgarr

def arrpassum(rows):

arrayout = []

for i in range(rows):

arrayout.append(summrow(i))

return arrayout

if \_\_name\_\_ == '\_\_main\_\_':

mypow = input("Which generarion of tribbles is to be calculated? ")

print()

print(" The " + mypow + "th generation of tribbles has this number of tribbles:")

output = "".join(map(lambda x: str(x),summrow(int(mypow))))

print(output)

#print(summrow(int(mypow)))

for line in (arrpas(int(mypow))):

print(line)

print("----------------")

print(" Powers of 11 from pascal triangle is equivalent to this number of tribbles " + mypow)

for line in (arrpassum(int(mypow)+1)):

print(line)

**Screen Output for number of tribbles in a given generation**

