堆疊

stack = []

def pushit():

stack.append(input('Enter New String: ').strip())

def popit():

if len(stack) == 0:

print("Cannot pop from an empty stack!")

else:

print("Remove [",stack.pop(),"]")

def viewstack():

print(stack)

CMDs = {'u':pushit, 'o':popit, 'v':viewstack}

def showmenu():

pr = """

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: """

while True:

while True:

try:

#先用strip()去掉空格,再把第一個字元轉換成小寫的

choice = input(pr).strip()[0].lower()

except (EOFError, KeyboardInterrupt, IndexError):

choice = 'q'

print("\nYou picked: [",choice,"]")

if choice not in 'uovq':

print("Invalid option, try again")

else:

break

if choice == 'q':

break

CMDs[choice]()

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

showmenu()

**Result:**

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: v

You picked: [v]

[]

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: u

You picked: [u]

Enter New String: Test1

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: v

You picked: [v]

['Test1']

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: u

You picked: [u]

Enter New String: Test2

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: U

You picked: [u]

Enter New String: test3

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: v

You picked: [v]

['Test1', 'Test2', 'test3']

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: o

You picked: [o]

Remove [ 'test3' ]

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: v

You picked: [v]

['Test1', 'Test2']

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: U

You picked: [u]

Enter New String: Test4

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: v

You picked: [v]

['Test1', 'Test2', 'Test4']

p(U)sh

p(O)p

(V)iew

(Q)uit

Enter choice: q

You picked: [q]

佇列

#!/use/bin/env python

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

from collections import deque

def yanghui(k):

"""

:param k: 楊輝三角中第幾層

:return: 第K層的係數

"""

q = deque([1]) # 建立一個佇列，預設從1開始

for i in range(k): # 迭代要查詢的層數

for \_ in range(i): # 迴圈需要出隊多少次

q.append(q.popleft() + q[0]) # 第一個數加上佇列中第二個數並賦值到佇列末尾

q.append(1) # 每次查詢結束後都需要在佇列最右邊新增個1

return list(q)

result = yanghui(3)

print(result)

**Result:**

[1, 3, 3, 1]