קורס

Basic Python

תרגולים

\*כל הזכויות שמורות למכללת NET4U\*

תוכן עניינים:

תרגולים.............................................3-26

**תרגולים מעשיים- פתרון**



import datetime

now = datetime.datetime.now()

print ("Current date and time : ")

print (now.strftime("%Y-%m-%d %H:%M:%S"))

print("Net4U, is the best palce \n\...in the world")



filename = input("Input the Filename: ")

f\_extns = filename.split(".")

print ("The extension of the file is : " + repr(f\_extns[-1]))

fname = input("Input your First Name : ")

lname = input("Input your Last Name : ")

print ("Hello " + lname + " " + fname)



a = int(input("Input an integer : "))

n1 = int( "%s" % a )

n2 = int( "%s%s" % (a,a) )

n3 = int( "%s%s%s" % (a,a,a) )

print (n1+n2+n3)



def difference(n):

if n <= 17:

return 17 - n

else:

return (n - 17) \* 2

print(difference(22))

print(difference(14))



num = int(input("Enter a number: "))

mod = num % 2

if mod > 0:

print("This is an odd number.")

else:

print("This is an even number.")



numbers = [

386, 462, 47, 418, 907, 344, 236, 375, 823, 566, 597, 978, 328, 615, 953, 345,

399, 162, 758, 219, 918, 237, 412, 566, 826, 248, 866, 950, 626, 949, 687, 217,

815, 67, 104, 58, 512, 24, 892, 894, 767, 553, 81, 379, 843, 831, 445, 742, 717,

958,743, 527

]

for x in numbers:

if x == 237:

print(x)

break;

elif x % 2 == 0:

print(x)



def add\_numbers(a, b):

if not (isinstance(a, int) and isinstance(b, int)):

raise TypeError("Inputs must be integers")

return a + b

print(add\_numbers(10, 20))



import site;

print(site.getsitepackages())



print("Input your height: ")

h\_ft = int(input("Feet: "))

h\_inch = int(input("Inches: "))

h\_inch += h\_ft \* 12

h\_cm = round(h\_inch \* 2.54, 1)

print("Your height is : %d cm." % h\_cm)

import os

# Access all environment variables

print('\*----------------------------------\*')

print(os.environ)

print('\*----------------------------------\*')

# Access a particular environment variable

print(os.environ['HOME'])

print('\*----------------------------------\*')

print(os.environ['PATH'])

print('\*----------------------------------\*')



time = float(input("Input time in seconds: "))

day = time // (24 \* 3600)

time = time % (24 \* 3600)

hour = time // 3600

time %= 3600

minutes = time // 60

time %= 60

seconds = time

print("d:h:m:s-> %d:%d:%d:%d" % (day, hour, minutes, seconds))



n = 20

d = {"x":200}

l = [1,3,5]

t= (5,7,8)

print(type(n)())

print(type(d)())

print(type(l)())

print(type(t)())



def test\_distinct(data):

if len(data) == len(set(data)):

return True

else:

return False;

print(test\_distinct([1,5,7,9]))

print(test\_distinct([2,4,5,5,7,9]))



print("Input the value of a, b, c, d, e, f:")

a, b, c, d, e, f = map(float, input().split())

n = a\*e - b\*d

print("Values of x and y:")

if n != 0:

x = (c\*e - b\*f) / n

y = (a\*f - c\*d) / n

print('{:.3f} {:.3f}'.format(x+0, y+0))

import bs4

from bs4 import BeautifulSoup as soup

from urllib.request import urlopen

news\_url="https://news.google.com/news/rss"

Client=urlopen(news\_url)

xml\_page=Client.read()

Client.close()

soup\_page=soup(xml\_page,"xml")

news\_list=soup\_page.findAll("item")

# Print news title, url and publish date

for news in news\_list:

print(news.title.text)

print(news.link.text)

print(news.pubDate.text)

print("-"\*60)



while True:

print("Input number of rows/columns (0 to exit)")

n = int(input())

if n == 0:

break

print("Input cell value:")

x = []

for i in range(n):

x.append([int(num) for num in input().split()])

for i in range(n):

sum = 0

for j in range(n):

sum += x[i][j]

x[i].append(sum)

x.append([])

for i in range(n + 1):

sum = 0

for j in range(n):

sum += x[j][i]

x[n].append(sum)

print("Result:")

for i in range(n + 1):

for j in range(n + 1):

print('{0:>5}'.format(x[i][j]), end="")

print()



def char\_frequency(str1):

dict = {}

for n in str1:

keys = dict.keys()

if n in keys:

dict[n] += 1

else:

dict[n] = 1

return dict

print(char\_frequency('google.com'))

20.

def not\_poor(str1):

snot = str1.find('not')

spoor = str1.find('poor')

if spoor > snot and snot>0 and spoor>0:

str1 = str1.replace(str1[snot:(spoor+4)], 'good')

return str1

else:

return str1

print(not\_poor('The lyrics is not that poor!'))

print(not\_poor('The lyrics is poor!'))

21.

def insert\_end(str):

sub\_str = str[-2:]

return sub\_str \* 4

print(insert\_end('Python'))

print(insert\_end('Exercises'))

22.

x = 3

y = 123

print("\nOriginal Number: ", x)

print("Formatted Number(left padding, width 2): "+"{:0>2d}".format(x));

print("Original Number: ", y)

print("Formatted Number(left padding, width 6): "+"{:0>6d}".format(y));

print()

def remove\_spaces(str1):

str1 = str1.replace(' ','')

return str1

print(remove\_spaces("w 3 res ou r ce"))

print(remove\_spaces("a b c"))

23.  
  
  
  
  
  
  
  
  
  
  
  
24.

25.   
  
  
  
  
  
26.  
  
  
  
  
  
27.  
  
  
  
28.  
  
  
  
  
  
  
29.

#Create a set

seta = set([5, 10, 3, 15, 2, 20])

#Find the length use len()

print(len(seta))

#Create a set

seta = set([5, 10, 3, 15, 2, 20])

#Find maximum value

print(max(seta))

#Find minimum value

print(min(seta))

price = [('item1', '12.20'), ('item2', '15.10'), ('item3', '24.5')]

print( sorted(price, key=lambda x: float(x[1]), reverse=True))

x = {'key1': 1, 'key2': 3, 'key3': 2}

y = {'key1': 1, 'key2': 2}

for (key, value) in set(x.items()) & set(y.items()):

print('%s: %s is present in both x and y' % (key, value))

num1 = [1, 3, 5, 7, 9, 10]

num2 = [2, 4, 6, 8]

num1[-1:] = num2

print(num1)

def remove\_zeros\_from\_ip(ip\_add):

new\_ip\_add = ".".join([str(int(i)) for i in ip\_add.split(".")])

return new\_ip\_add ;

print(remove\_zeros\_from\_ip("255.024.01.01"))

print(remove\_zeros\_from\_ip("127.0.0.01 "))

30.  
  
  
  
  
  
  
31.  
  
  
  
  
  
  
  
32.  
  
  
  
  
  
  
  
  
  
33.  
  
  
  
  
34.  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
35.  
  
  
  
  
  
  
  
  
36.  
  
  
  
  
  
  
  
  
  
37.  
  
  
  
  
  
  
  
  
38.  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
39.  
  
  
  
  
  
  
  
  
  
  
40.  
  
  
  
  
41.  
  
  
  
  
  
  
42.  
  
  
  
  
  
  
43.  
  
  
  
  
  
  
44.

def sum(numbers):

total = 0

for x in numbers:

total += x

return total

print(sum((8, 2, 3, 0, 7)))

for i in range(10):

print(str(i) \* i)

result\_str="";

for row in range(0,7):

for column in range(0,7):

if (((column == 1 or column == 5) and row != 0) or ((row == 0 or row == 3) and (column > 1 and column < 5))):

result\_str=result\_str+"\*"

else:

result\_str=result\_str+" "

result\_str=result\_str+"\n"

print(result\_str);

from array import \*

array\_num = array('i', [1, 3, 5, 3, 7, 1, 9, 3])

print("Original array: "+str(array\_num))

num\_list = array\_num.tolist()

print("Convert the said array to an ordinary list with the same items:")

print(num\_list)

from array import \*

array\_num = array('i', [1, 3, 5, 7, 9])

print("Original array: "+str(array\_num))

print("Insert new value 4 before 3:")

array\_num.insert(1, 4)

print("New array: "+str(array\_num))

def pascal\_triangle(n):

trow = [1]

y = [0]

for x in range(max(n,0)):

print(trow)

trow=[l+r for l,r in zip(trow+y, y+trow)]

return n>=1

pascal\_triangle(6)

import queue

q = queue.Queue()

#insert items at the end of the queue

for x in range(4):

q.put(str(x))

#remove items from the head of the queue

while not q.empty():

print(q.get(), end=" ")

print("\n")

import queue

q = queue.LifoQueue()

#insert items at the head of the queue

for x in range(4):

q.put(str(x))

#remove items from the head of the queue

while not q.empty():

print(q.get(), end=" ")

print()

def Sequential\_Search(dlist, item):

pos = 0

found = False

while pos < len(dlist) and not found:

if dlist[pos] == item:

found = True

else:

pos = pos + 1

return found, pos

print(Sequential\_Search([11,23,58,31,56,77,43,12,65,19],31))

class py\_solution:

def sub\_sets(self, sset):

return self.subsetsRecur([], sorted(sset))

def subsetsRecur(self, current, sset):

if sset:

return self.subsetsRecur(current, sset[1:]) + self.subsetsRecur(current + [sset[0]], sset[1:])

return [current]

print(py\_solution().sub\_sets([4,5,6]))

import itertools

x = itertools.cycle('ABCD')

print(type(x).\_\_name\_\_)

base\_1 = 5

base\_2 = 6

height = float(input("Height of trapezoid: "))

base\_1 = float(input('Base one value: '))

base\_2 = float(input('Base two value: '))

area = ((base\_1 + base\_2) / 2) \* height

print("Area is:", area)

def Previous\_Palindrome(num):

for x in range(num-1,0,-1):

if str(x) == str(x)[::-1]:

return x

print(Previous\_Palindrome(99));

print(Previous\_Palindrome(1221));

def file\_read\_from\_head(fname, nlines):

from itertools import islice

with open(fname) as f:

for line in islice(f, nlines):

print(line)

file\_read\_from\_head('test.txt',2)

def file\_size(fname):

import os

statinfo = os.stat(fname)

return statinfo.st\_size

print("File size in bytes of a plain file: ",file\_size("test.txt"))

45.

def remove\_newlines(fname):

flist = open(fname).readlines()

return [s.rstrip('\n') for s in flist]

print(remove\_newlines("test.txt"))

46.   
  
  
  
  
  
  
  
  
  
47.

import re

def text\_match(text):

patterns = '^\w+'

if re.search(patterns, text):

return 'Found a match!'

else:

return('Not matched!')

print(text\_match("The quick brown fox jumps over the lazy dog."))

print(text\_match(" The quick brown fox jumps over the lazy dog."))

import re

def text\_match(text):

patterns = 'ab+?'

if re.search(patterns, text):

return 'Found a match!'

else:

return('Not matched!')

print(text\_match("ab"))

print(text\_match("abc"))

48.

import re

def capital\_words\_spaces(str1):

return re.sub(r"(\w)([A-Z])", r"\1 \2", str1)

print(capital\_words\_spaces("Python"))

print(capital\_words\_spaces("PythonExercises"))

print(capital\_words\_spaces("PythonExercisesPracticeSolution"))