Рівень А

№ 1

Завдання:

Опишіть рекурсивну функцію GCD, яка повертає найбільший спільний дільник чисел x та y. Обчислення НСД відбувається рекурсивним чином відповідно до формули

Код:

def gcd(x,y):   
  
 if (x == y):   
 return y  
   
 if (x < y):   
 return gcd(x, y-x)  
 if (x > y):   
 return gcd(x-y, y)   
   
x = 98  
y = 56  
if(gcd(x, y)):   
 print('НСД', x, 'і', y, 'є', gcd(x, y))   
else:   
 print('Не знайдено')   
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

Вивід:

C:\Users\Рома\Documents\ShareX\Screenshots\2019-06\tOtyHf3AK3.png

№ 2

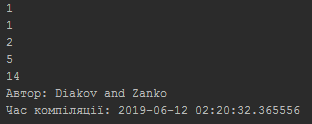
Завдання:

Вивести n-те число Каталана за допомогою рекурсивного алгоритму

Код:

def catalan(n):  
 if n >= 2:  
 c = ((2 \* ((2\*n) - 1)) / (n+1)) \* catalan(n-1)  
 return int(c)  
 return 1  
   
   
print(catalan(0))  
print(catalan(1))  
print(catalan(2))  
print(catalan(3))  
print(catalan(4))  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

Вивід:



№ 3

Завдання:

Опишіть рекурсивну функцію для піднесення числа до цілої додатної степені

Код:

def expt(b, n):  
 if n==0:  
 return 1  
 return b\*expt(b, n-1)  
  
  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

№ 4

Завдання:

Більш ефективне рекурсивне визначення , що охоплює випадок

Код:

def MinusPower(x, n):  
 if n == 0:  
 return 1  
 elif n < 0:  
 return x\*\*n  
 elif n % 2 == 0:  
 return MinusPower(x, n/2)\*\*2  
 else:  
 return x \* MinusPower(x, n-1)  
  
print(MinusPower(2,-1))  
  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

C:\Users\Рома\Documents\ShareX\Screenshots\2019-06\pycharm64_OBNtn9cWvq.pngВивід:

№ 6

Завдання:

Градова послідовність (градові числа, послідовність Колатца) називається так тому, що її значення зазвичай зростають та зменшуються подібно до форми градових хмар.

Створіть програму, яка *рекурсивно* генеруватиме градову послідовність чисел, починаючи з деякого введеного цілого числа n. Правила формування послідовності:

Якщо n = 1, послідовність закінчується.

Якщо n – парне, наступне число послідовності буде n/2

Якщо n – непарне, наступне число послідовності = (3 \* n) + 1

Використайте програму, щоб показати, що для числа 27 послідовність складатиметься з 112 елементів, починаючись з 27, 82, 41, 124 та закінчуючись числами 8, 4, 2, 1

Код:

def hailstone(n):  
 print(n)  
 if n == 1:  
 return 1  
 elif n % 2 == 0:  
 return 1 + hailstone(n // 2)  
 else:  
 return 1 + hailstone(3 \* n + 1)  
print(hailstone(27))  
  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

Вивід:

№ 7

Завдання:

Напишіть програму, яка повністю зчитує файл та дописує в нього ПІБ її автора

Код:

pib = input("Введите ваше ФИО: ")  
f = open(r'A7.txt', 'r')  
infile = f.read()  
print("Изначальный текст")  
print(infile)  
  
f = open(r'A7.txt', 'a')  
filewrite = f.write(pib)  
print(filewrite)  
print("Текст с ФИО")  
f = open(r'A7.txt', 'r')  
prf = f.read()  
print(prf)

Вивід:

№ 8

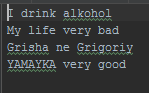
Завдання:

Напишіть програму, яка видалить з файлу всі переходи на новий рядок

Код:

rf = open("teht.txt")  
wf = open("newteht.txt","w")  
for string in rf:  
 new\_string = string.replace("\n","")  
 wf.write(new\_string)  
 wf.write('. ')  
rf.close()  
wf.close()  
  
  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

C:\Users\Рома\Documents\ShareX\Screenshots\2019-06\pycharm64_AT26QbmfY0.pngВивід:



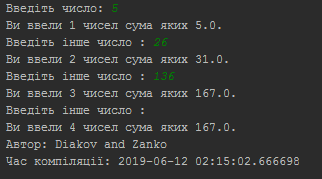
№13

Створіть програму, яка підраховує суму всіх введених користувачем чисел, пропускаючи некоректно введені (нечислові) дані. Програма має виводити поточну суму після кожного введеного числа. При вводі нечислових даних виводиться повідомлення про помилку, проте робота програми продовжується. Вихід з програми здійснюється після вводу порожнього рядка. Переконайтесь, що програма працює коректно як для цілих, так і дробових значень.

Підказка: вправа вимагає використання виключень без використання файлів.

Код:

prompt = "Введіть число: "  
sum = 0  
count = 1  
while True:  
 s = input(prompt)  
 prompt = "Введіть інше число : "  
 if s.lower() == '':  
 break  
 try:  
 sum += float(s)  
 print ("Ви ввели %s чисел сума яких %s." % (count, sum))  
 count += 1  
 except ValueError:  
 print( "Неправильний ввід")  
print ("Ви ввели %s чисел сума яких %s." % (count, sum))  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)



№ 14

Завдання:

Генерування паролю шляхом вибору випадкових символів дає відносно безпечний, проте загалом складний для запам’ятовування пароль. Альтернативним способом є конструювання паролю з двох агнлійських слів з подальшою конкатенацією. Такий пароль простіше запам’ятати, хоча і зламати також.

Напишіть програму, яка зчитує файл зі списком слів, обирає два випадковим чином та з’єднує їх в одне. При створенні паролю переконайтесь, що загальна його довжина знаходиться в діапазоні від 8 до 10 символів, а кожне використане слово складається принаймні з трьох букв. Зробіть так, щоб у паролі кожне слово починалось з великої літери та виведіть пароль для користувача.

Код:

import random  
lines = open('A14.txt').read().splitlines()  
word1 = random.choice(lines)  
word2 = random.choice(lines)  
print("Your Password: " + (word1+word2))  
  
import datetime  
def printTimeStamp(name):  
 print('Автор програми: ' + name)  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
printTimeStamp('Diakov and Zanko')

Вивід:

Рівень B

№ 1

Завдання:

Трикутник Паскаля – це трикутник чисел, який містить біноміальні коефіцієнти, що знаходяться за формулою

Напишіть функцію, яка будуватиме трикутник Паскаля заданої висоти (аргумент функції). Наприклад,

1

1 1

1 2 1

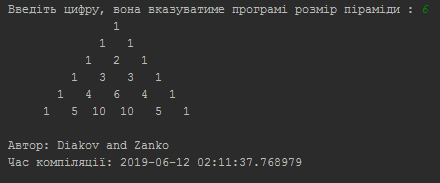
1 3 3 1

1 4 6 4 1

Зверніть увагу на форматування виводу трикутника.

Код:

import math  
  
coef = 1  
  
rows = int(input('Введiть цифру, вона вказуватиме програмi розмiр пiрамiди : '))  
  
stringVal = ""  
  
for i in range(rows):  
  
 for space in range(1,rows - i + 1):  
  
 stringVal = stringVal + " "  
 for j in range(0,i + 1):  
 if(i == 0 or j == 0):  
 coef = 1  
 else:  
 coef = coef \* (i - j + 1) / j  
  
 temp = coef  
 TotalSpace=0  
 while(temp != 0):  
 TotalSpace = TotalSpace + 1  
 temp = int(math.floor(temp / 10))  
 p=0  
 while((p+TotalSpace)!=4):  
 stringVal = stringVal + " "  
 p=p+1  
 stringVal = stringVal + str(int(math.floor(coef)))  
 stringVal = stringVal + "\n"  
print(stringVal)  
  
  
  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

Вивід:

№ 7

Завдання:

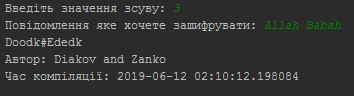
Один з перших відомих прикладів шифрування використовував Юлій Цезар. Йому потрібно було надсилати письмові інструкції своїм генералам, проте він не хотів, щоб вороги дізнались про його плани, якщо повідомлення потрапить їм у руки. Тому Цезар запропонував просту ідею: кожна літера первинного повідомлення зсувалась на 3 позиції в алфавіті. Замість A записували D, замість B – E, C ставало F, D – G і т. д. Останні 3 букви в алфавіті мали замінники з його початку (X = A, Y = B, Z = C). Небуквенні символи не зашифровувались.

Напишіть програму, яка реалізує шифр Цезаря. Дозвольте користувачеві ввести повідомлення та розмір зсуву, а потім виведіть «зсунуте» повідомлення. Переконайтесь, що Ваша програма коду як великі, так і маленькі букви. Додаток також має підтримувати від’ємний зсув, щоб мати можливість як зашифрувати, так і розшифрувати повідомлення.

Код:

ALPHA = "".join(map(chr, range(ord(" "), ord("¤") + 1)))  
   
def encode(text, step):  
 return text.translate(  
 str.maketrans(ALPHA, ALPHA[step:] + ALPHA[:step]))  
   
  
b = input("Введіть значення зсуву: ")  
  
  
a = input("Повідомлення яке хочете зашифрувати: ")  
  
print(encode(a, int(b)))  
  
  
  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

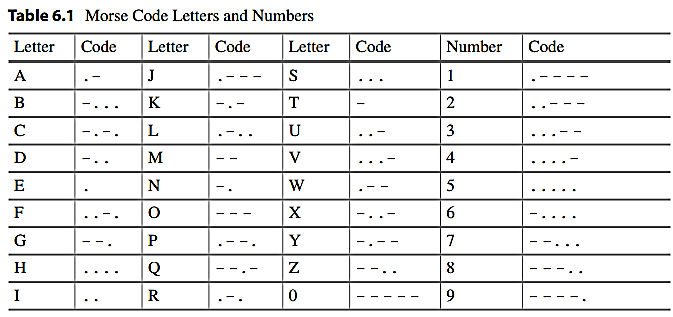
Вивід:



№ 8

Завдання:

Код Морзе – схема кодування, яка використовує точки та тире для представлення цифр і букв. Напишіть програму, яка буде використовувати словник для зберігання відображення букв і цифр на код Морзе (таблиця).

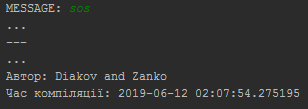


Ваша програма має читати повідомлення користувача та перетворювати його в код Морзе з наступним виводом закодованого тексту. Пробіли між словами слід залишати, а решту символів, що не є цифрою або буквою – ігнорувати.

Код:

CODE = {'A': '.-', 'B': '-...', 'C': '-.-.',   
 'D': '-..', 'E': '.', 'F': '..-.',  
 'G': '--.', 'H': '....', 'I': '..',  
 'J': '.---', 'K': '-.-', 'L': '.-..',  
 'M': '--', 'N': '-.', 'O': '---',  
 'P': '.--.', 'Q': '--.-', 'R': '.-.',  
 'S': '...', 'T': '-', 'U': '..-',  
 'V': '...-', 'W': '.--', 'X': '-..-',  
 'Y': '-.--', 'Z': '--..',  
   
 '0': '-----', '1': '.----', '2': '..---',  
 '3': '...--', '4': '....-', '5': '.....',  
 '6': '-....', '7': '--...', '8': '---..',  
 '9': '----.'   
 }  
  
  
def main():  
   
 msg = str(input('MESSAGE: '))  
   
 for char in msg:  
 print (CODE[char.upper()],)  
   
if \_\_name\_\_ == "\_\_main\_\_":  
 main()  
  
  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

Вивід:



№12

Реалізуйте гру «Хрестики-нолики». Крім головної функції, виділіть окремі підпрограми для

* + - ініціалізації матриці 3х3 «порожніми» символами,
    - обробки ходу гравця,
    - обробки ходу комп’ютера,
    - перевірки виграшу,
    - виводу результатів.

Нехай комп’ютер під час свого ходу заповнює першу «порожню» клітинку, а гравець – задає координати (X,Y), які треба перевірити на валідність.

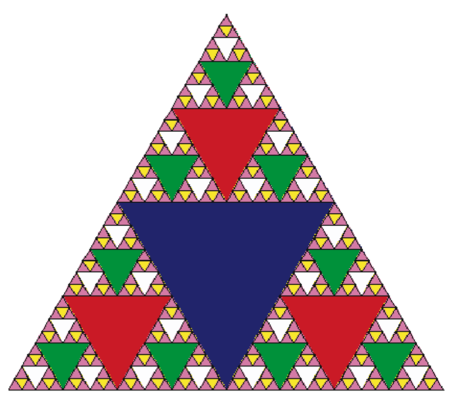
Код:

import random  
  
def drawBoard(board):  
 print(' | |')  
 print(' ' + board[7] + ' | ' + board[8] + ' | ' + board[9])  
 print(' | |')  
 print('-----------')  
 print(' | |')  
 print(' ' + board[4] + ' | ' + board[5] + ' | ' + board[6])  
 print(' | |')  
 print('-----------')  
 print(' | |')  
 print(' ' + board[1] + ' | ' + board[2] + ' | ' + board[3])  
 print(' | |')  
  
def inputPlayerLetter():  
 letter = ''  
 while not (letter == 'X' or letter == 'O'):  
 print('Do you want to be X or O?')  
 letter = input().upper()  
 if letter == 'X':  
 return ['X', 'O']  
 else:  
 return ['O', 'X']  
def whoGoesFirst():  
 if random.randint(0, 1) == 0:  
 return 'computer'  
 else:  
 return 'player'  
def playAgain():  
 print('Do you want to play again? (yes or no)')  
 return input().lower().startswith('y')  
def makeMove(board, letter, move):  
 board[move] = letter  
def isWinner(bo, le):  
 return ((bo[7] == le and bo[8] == le and bo[9] == le) or   
 (bo[4] == le and bo[5] == le and bo[6] == le) or   
 (bo[1] == le and bo[2] == le and bo[3] == le) or   
 (bo[7] == le and bo[4] == le and bo[1] == le) or   
 (bo[8] == le and bo[5] == le and bo[2] == le) or  
 (bo[9] == le and bo[6] == le and bo[3] == le) or   
 (bo[7] == le and bo[5] == le and bo[3] == le) or   
 (bo[9] == le and bo[5] == le and bo[1] == le))  
def getBoardCopy(board):  
 dupeBoard = []  
 for i in board:  
 dupeBoard.append(i)  
 return dupeBoard  
def isSpaceFree(board, move):  
 return board[move] == ' '  
def getPlayerMove(board):  
 move = ' '  
 while move not in '1 2 3 4 5 6 7 8 9'.split() or not isSpaceFree(board, int(move)):  
 print('What is your next move? (1-9)')  
 move = input()  
 return int(move)  
def chooseRandomMoveFromList(board, movesList):  
 possibleMoves = []  
 for i in movesList:  
 if isSpaceFree(board, i):  
 possibleMoves.append(i)  
 if len(possibleMoves) != 0:  
 return random.choice(possibleMoves)  
 else:  
 return None  
def getComputerMove(board, computerLetter):  
 if computerLetter == 'X':  
 playerLetter = 'O'  
 else:  
 playerLetter = 'X'  
 for i in range(1, 10):  
 copy = getBoardCopy(board)  
 if isSpaceFree(copy, i):  
 makeMove(copy, computerLetter, i)  
 if isWinner(copy, computerLetter):  
 return i  
 for i in range(1, 10):  
 copy = getBoardCopy(board)  
 if isSpaceFree(copy, i):  
 makeMove(copy, playerLetter, i)  
 if isWinner(copy, playerLetter):  
 return i  
 move = chooseRandomMoveFromList(board, [1, 3, 7, 9])  
 if move != None:  
 return move  
 if isSpaceFree(board, 5):  
 return 5  
 return chooseRandomMoveFromList(board, [2, 4, 6, 8])  
def isBoardFull(board):  
 for i in range(1, 10):  
 if isSpaceFree(board, i):  
 return False  
 return True  
print('Welcome to Tic Tac Toe!')  
while True:  
 theBoard = [' '] \* 10  
 playerLetter, computerLetter = inputPlayerLetter()  
 turn = whoGoesFirst()  
 print('The ' + turn + ' will go first.')  
 gameIsPlaying = True  
 while gameIsPlaying:  
 if turn == 'player':  
 drawBoard(theBoard)  
 move = getPlayerMove(theBoard)  
 makeMove(theBoard, playerLetter, move)  
 if isWinner(theBoard, playerLetter):  
 drawBoard(theBoard)  
 print('Hooray! You have won the game!')  
 gameIsPlaying = False  
 else:  
 if isBoardFull(theBoard):  
 drawBoard(theBoard)  
 print('The game is a tie!')  
 break  
 else:  
 turn = 'computer'  
 else:  
 move = getComputerMove(theBoard, computerLetter)  
 makeMove(theBoard, computerLetter, move)  
 if isWinner(theBoard, computerLetter):  
 drawBoard(theBoard)  
 print('The computer has beaten you! You lose.')  
 gameIsPlaying = False  
 else:  
 if isBoardFull(theBoard):  
 drawBoard(theBoard)  
 print('The game is a tie!')  
 break  
 else:  
 turn = 'player'  
 if not playAgain():  
 break  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

Рівень С

№3

Фрактали є самоподібними структурами, тому природною є рекурсивна структура алгоритму їх побудови. Напишіть програму, яка буде виводити на екран килим Серпінського.



Ідея побудови килиму наступна. Починаємо з великого залитого одним кольором трикутника. Ділимо його на 4 частини, з’єднавши середини кожної зі сторін трикутника (темносиній трикутник на рисунку). Продовжуємо робити так і для отриманих трикутників (червоні трикутники, потім зелені, білі та жовті). Таким чином, Вам будуть потрібні функції, що

* знаходитимуть середини сторін заданого трикутника (визначається покоординатно)
* відрисовуватимуть сам трикутник, наприклад, за допомогою turtle

формуватимуть нові трикутники за вище наведеними правилами

Код:

import turtle  
  
def drawTriangle(points,color,myTurtle):  
 myTurtle.fillcolor(color)  
 myTurtle.up()  
 myTurtle.goto(points[0][0],points[0][1])  
 myTurtle.down()  
 myTurtle.begin\_fill()  
 myTurtle.goto(points[1][0],points[1][1])  
 myTurtle.goto(points[2][0],points[2][1])  
 myTurtle.goto(points[0][0],points[0][1])  
 myTurtle.end\_fill()  
  
def getMid(p1,p2):  
 return ( (p1[0]+p2[0]) / 2, (p1[1] + p2[1]) / 2)  
  
def sierpinski(points,degree,myTurtle):  
 colormap = ['red','purple','orange','dark blue','gold',  
 'violet','lawn green']  
 drawTriangle(points,colormap[degree],myTurtle)  
 if degree > 0:  
 sierpinski([points[0],  
 getMid(points[0], points[1]),  
 getMid(points[0], points[2])],  
 degree-1, myTurtle)  
 sierpinski([points[1],  
 getMid(points[0], points[1]),  
 getMid(points[1], points[2])],  
 degree-1, myTurtle)  
 sierpinski([points[2],  
 getMid(points[2], points[1]),  
 getMid(points[0], points[2])],  
 degree-1, myTurtle)  
  
def main():  
 myTurtle = turtle.Turtle()  
 myWin = turtle.Screen()  
 myPoints = [[-200,-100],[0,200],[200,-100]]  
 sierpinski(myPoints,5,myTurtle)  
 myWin.exitonclick()  
  
main()  
  
  
import datetime  
  
def printTimeStamp(name):  
  
 print('Автор: ' + name)  
  
 print('Час компіляції: ' + str(datetime.datetime.now()))  
name="Diakov and Zanko"  
printTimeStamp(name)

Вивід:

