Tuples

1] A complex number is a number of the form a + bi, whereby a and b are constants, and i is a special value that is defined as the square root of -1. Of course, you never try to actually calculate what the square root of -1 is, as that gives a runtime error; in complex numbers, you always let the i remain. For instance, the complex number 3 + 2i cannot be simplified any further. Addition of two complex numbers a + bi and c + di is defined as (a + c) + (b + d)i. Represent a complex number as a tuple of two numeric values, and create a function that calculates the addition of two complex numbers.

2] Multiplication of two complex numbers a + bi and c + di is defined as (a\*c - b\*d) + (a\*d + b\*c) i.

Write a function that calculates the multiplication of two complex numbers.

3] Distance between two points in N- dimensional space. The points should have the same

dimension , i.e., they are tuples of numeric values , and they should have the same length.

def distance ( p1 , p2 ):

4] Write a program that asks the user to enter some data, for instance the names of their friends. When the user wants to stop providing inputs, he just presses Enter. The program then displays an alphabetically sorted list of the data items entered. Do not just print the list, but print each item separately, on a different line.

5] Count how often each letter occurs in a string (case-insensitively). You can ignore every character that is not a letter. Store the counts in a list of 26 items that all start at zero. Print the resulting counts. As index you can use ord(letter) - ord("a"), where letter is a lower case letter.

LIST

1 ]A magic 8-ball, when asked a question, provides a random answer from a list. The code below contains a list of possible answers. Create a magic 8-ball program that asks a question, then gives a random answer.

Ex

import random

L=[5,6,7,8]

Print(random.choice(L))

Will return random no from list.

answers = [ "It is certain", "It is decidedly so", "Without a \ doubt", "Yes, definitely", "You may rely on it", "As I see it, \ yes", "Most likely", "Outlook good", "Yes", "Signs point to yes", "Reply hazy try again", "Ask again later", "Better not tell you \ now", "Cannot predict now", "Concentrate and ask again", "Don ' t \ count on it", "My reply is no", "My sources say no", "Outlook \ not so good", "Very doubtful" ]

2] A playing card consists of a suit ("Hearts", "Spades", "Clubs", "Diamonds") and a value (2, 3, 4, 5, 6, 7, 8, 9, 10, "Jack", "Queen", "King", "Ace"). Create a list of all possible playing cards, which is a deck. Then create a func-tion that shuffles the deck, producing a random order.

3] A first-in-first-out (FIFO) structure, also called a “queue,” is a list that gets new elements added at the end, while elements from the front are removed and processed. Write a program that processes a queue. In a loop, ask the user for input. If the user just presses the Enter key, the program ends. If the user enters anything else, except for a single question mark (?), the program considers what the user entered a new element and appends it to the queue. If the user enters a single question mark, the program pops the first element from the queue and displays it. You have to take into account that the user might type a question mark even if the queue is empty.

4] Count how often each letter occurs in a string (case-insensitively). You can ignore every character that is not a letter. Print the letters with their counts, in order from highest count to lowest count.

5] The sieve of Eratosthenes is a method to find all prime numbers between 1 and a given number using a list. This works as follows: Fill the list with the sequence of numbers from 1 to the highest number. Set the value of 1 to zero, as 1 is not prime. Now loop over the list. Find the next number on the list that is not zero, which, at the start, is the number 2. Now set all multiples of this number to zero. Then find the next number on the list that is not zero, which is 3. Set all multiples of this number to zero. Then the next number, which is 5 (because 4 has already been set to zero), and do the same thing again. Process all the numbers of the list in this way. When you have finished, the only numbers left on the list are primes. Use this method to determine all the primes between 1 and 100.

6] The “subset sum” problem asks the question whether a list of integers contains a subset of integers that, when summed, gives zero as answer. For instance, for the list [1, 4, -3, -5, 7] the answer is “yes,” as 1 + 4 5 = 0. However, for the list [1, 4, -3, 7] the answer is “no,” as there is no subset of integers that adds up to zero.

|  |  |
| --- | --- |
|  |  |
|  |  |

Write a program that solves the “subset sum” problem for a list of integers. If there is a solution, print it; if not, report that there is no solution.

Dictionary

1] Write a program that takes a text (for instance the one given below), splits it into words (where everything that is not a letter is considered a word boundary), and case-insensitively builds a dictionary that stores for every word how often it occurs in the text. Then print all the words with their quantities in alphabetical order.

text = """How much wood would a woodchuck chuck If a woodchuck could chuck wood?

He would chuck , he would , as much as he could ,

And chuck as much as a woodchuck would

If a woodchuck could chuck wood."""

2] The code block below shows a list of movies. For each movie it also shows a list of ratings. Convert this code in such a way that it stores all this data in one dictionary, then use the dictionary to print the average rating for each movie, rounded to one decimal.

movies = ["Monty Python and the Holy Grail", "Monty Python ' s Life of Brian", "Monty Python ' s Meaning of Life",

"And Now For Something Completely Different"]

grail\_ratings = [ 9, 10, 9.5 , 8.5 , 3, 7.5 ,8 ]

brian\_ratings = [ 10, 10, 0, 9, 1, 8, 7.5 , 8, 6, 9 ]

life\_ratings = [ 7, 6, 5 ]

different\_ratings = [ 6, 5, 6, 6 ]

3] A library contains books. Books have a writer, identified by last name and first name. Books also have a title. Books also have a location number that identifies where they can be found in the library. Librarians want to be able to locate a specific book if they know writer and title, and they want to be able to list all the books that they have of a specific writer. What data structure would you use to store the books?