Лабораториялық жұмыс №10

10 кірістірілген функцияны алып, бір бірімен үйлестіріп бағдарлама жазамыз. Барлық функциялар өзара байланысты болуы керек, жəне бағдарламаның логикалық басы мен соңы болуы қажет. Функциялар осыған дейін біз қолданбаған болуы қажет.

2.1.Пернетақтадан жолды оқитын жəне сол жолдағы барлық бірегей таңбаларды алфавиттік ретпен көрсететін бағдарламаны жазыңыз. Ол үшін кірістірілген функцияларды пайдалану керек. Напишите программу, которая считыва

strings = input("Введите строку: ")  # считываем строку с клавиатуры

# создаем множество, чтобы получить только уникальные символы

chars = set(strings)

 # сортируем символы по алфавиту

sorted\_chars = sorted(chars)

 # выводим результат на экран

print("Уникальные символы в строке: ", end="")

for char in sorted\_chars:

     print(char, end="")

2.2.Тізімде берілген шартты қанағаттандыратын кем дегенде бір элемент бар-жоғын тексеру үшін кірістірілген any() функциясын қолданатын Python бағдарламасын жазыңыз. Содан кейін тізімнің барлық элементтері берілген шартты қанағаттандыратынын тексеру үшін кірістірілген all() функциясын пайдаланыңыз

numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Проверяем, есть ли в списке хотя бы одно четное число

if any(num % 2 == 0 for num in numbers):

    print("В списке есть хотя бы одно четное число")

else:

    print("В списке нет четных чисел")

# Проверяем, все ли числа в списке меньше 11

if all(num < 11 for num in numbers):

    print("Все числа в списке меньше 11")

else:

    print("Не все числа в списке меньше 11")

2.3.Матрицаны (тізімдер тізімі) алатын жəне сағат тілімен 90 градусқа бұрылған матрицаны қайтаратын функцияны орындаңыз. zip(), list() жəне reversed() функцияларын пайдаланыңыз

def rotate\_matrix\_clockwise(matrix):

    """

    Функция, которая поворачивает матрицу на 90 градусов по часовой стрелке

    :param matrix: исходная матрица (список списков)

    :return: повернутая матрица (список списков)

    """

    # используем функцию zip() и reversed() для получения столбцов матрицы

    columns = [list(reversed(col)) for col in zip(\*matrix)]

    # возвращаем транспонированную матрицу, которая является результатом поворота на 90 градусов

    return [list(row) for row in zip(\*columns)]

matrix = [

    [1, 2, 3],

    [4, 5, 6],

    [7, 8, 9]

]

#Вызывваем функцию

rotated\_matrix = rotate\_matrix\_clockwise(matrix)

print(rotated\_matrix)

2.4.Динамикалық бағдарламалау арқылы сөмке мəселесін шешу функциясын орындаңыз. Функция екі тізімді (элементтердің салмағы мен мəндері) жəне рюкзактың максималды салмағын алады жəне рюкзакта тасымалдауға болатын максималды мəнді қайтарады. range(), enumerate() жəне max() функцияларын пайдаланыңыз

def knapsack(weights, values, max\_weight):

    n = len(weights) # количество предметов

    table = [[0 for x in range(max\_weight+1)] for x in range(n+1)]

    for i in range(n+1):

        for w in range(max\_weight+1):

            if i==0 or w==0:

                table[i][w] = 0

            elif weights[i-1] <= w:

                table[i][w] = max(values[i-1] + table[i-1][w-weights[i-1]], table[i-1][w])

            else:

               table[i][w] = table[i-1][w]

    return table[n][max\_weight]

max\_weight = 6

weights = [1, 2, 3, 4]

values = [1, 2, 3, 5]

result = knapsack(weights, values, max\_weight)

print(result)

2.5.Екі матрицаға (қосу, алу, көбейту) амалдар орындайтын функцияны іске асыру. Функция екі матрица мен операция белгісін алып, операцияның нəтижесін қайтарады. enumerate(), zip() жəне len() функцияларын пайдаланыңыз

def matrix\_operation(matrix1, matrix2, operation):

    # проверяем размерности матриц

    if len(matrix1) != len(matrix2) or len(matrix1[0]) != len(matrix2[0]):

        return None

    # создаем результирующую матрицу

    result\_matrix = [[0 for j in range(len(matrix1[0]))] for i in range(len(matrix1))]

    # выполняем операцию

    for i in range(len(matrix1)):

        for j in range(len(matrix1[0])):

            if operation == '+':

                result\_matrix[i][j] = matrix1[i][j] + matrix2[i][j]

            elif operation == '-':

                result\_matrix[i][j] = matrix1[i][j] - matrix2[i][j]

            elif operation == '\*':

                result\_matrix[i][j] = matrix1[i][j] \* matrix2[i][j]

    return result\_matrix

matrix1 = [[1, 2], [3, 4]]

matrix2 = [[5, 6], [7, 8]]

result\_matrix = matrix\_operation(matrix1, matrix2, '+')

print(result\_matrix)  # [[6, 8], [10, 12]]

result\_matrix = matrix\_operation(matrix1, matrix2, '-')

print(result\_matrix)  # [[-4, -4], [-4, -4]]

result\_matrix = matrix\_operation(matrix1, matrix2, '\*')

print(result\_matrix)  # [[5, 12], [21, 32]]