ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

ИАиИТ институты

Программная инженерия кафедрасы



ЗЕРТХАНАЛЫҚ ЖҰМЫС № 10

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Орындалған жұмыстың сапасы** | **Баға диапазоны** | **Алған %** |
| 1 | Орындалмаған  Себепсіз сабақтан қалуы | 0% |  |
| 2 | Орындалуы және білім алушының белсенділігі | 0-50% |  |
| 3 | Жұмыстың рәсімделуі | 0-20% |  |
| 4 | Анықтамаларды, техникалық әдебиеттерді, дәріс конспектілерін, пәннің оқу-әдістемелік кешенін пайдалана білуі | 0-5% |  |
| 5 | Техникалық құралдарды пайдалана білуі | 0-5% |  |
| 6 | Жұмысты қорғауы | 0-20% |  |
|  | Қорытынды | 0-100% |  |

Орындаған:Оразалиев Н.

Тексерген:Шаяхметов Д.

Группа: Среда 7.50

Алматы 2023 ж

Лабораториялық жұмыс №10

GIDHUB-пен жұмыс жасауды жалғастырамыз. Бір репозиториде бірнеше студент жұмыс жасау міндетті. Топтың əрбір студенті кодтың бір бөлігін бір файлға қосуы қажет. Бірақ бөлек қорғау керек.

1. 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.

# 1,2 (str), (print)

jasy = 15

print("Nurmukhammed: " + str(jasy))

#3,4 (min-max)

my\_list = [1, 2, 3, 4, 5]

print(max(my\_list))

print(min(my\_list))

#5 list

my\_list = [1, 2, 3, 4, 5]

print(len(my\_list))

#6

for i in range(5):

print(i)

#7

name = input("What is your name? ")

print("Hello, " + name + "!")

#8,9,10

my\_variable = "Hello, World!"

my\_string = "3.14"

my\_number = float(my\_string)

print(type(my\_number))

print(type(my\_variable))

1. Берілген есептерді шығарыңыз:
   1. Пернетақтадан жолды оқитын жəне сол жолдағы барлық бірегей таңбаларды алфавиттік ретпен көрсететін бағдарламаны жазыңыз. Ол үшін кірістірілген функцияларды пайдалану керек.

*Напишите программу, которая считывает строку с клавиатуры и выводит на экран все уникальные символы в этой строке в алфавитном порядке. Для этого необходимо использовать встроенные функциию.*

#esep 1

strings = input("Введите строку: ")

s2=strings.lower()

chars = set(s2)

sorted\_chars = sorted(chars)

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

for char in sorted\_chars:

print(char, end="")

* 1. 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

*Напишите программу на Python, которая использует встроенную функцию any() для проверки, есть ли в списке хотя бы один элемент, удовлетворяющий заданному условию. Затем используйте встроенную функцию all() для проверки, удовлетворяют ли все элементы списка заданному условию*

#esep 2

list=[1,2,-3,4,5]

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

print("Список содержит четное число")

else:

print("Спиосок не содержит четное число")

if all(num>0 for num in list):

print("Все элементы в списке положительные")

else:

print("В списке присутсвует отрицательное число")

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

*Реализуйте функцию, которая принимает матрицу (список списков) и возвращает матрицу, повернутую на 90 градусов по часовой стрелке. Используйте функции zip(), list() и reversed()*

#esep 3

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]

]

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

print(rotated\_matrix)

* 1. 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

*Реализуйте функцию для решения задачи о рюкзаке с помощью динамического программирования. Функция принимает два списка (веса и стоимости предметов) и максимальный вес рюкзака, а возвращает максимальную стоимость, которую можно унести в рюкзаке. Используйте функции range(), enumerate() и max()*

#esep 4

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

n = len(weights)

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)

* 1. 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

*Реализуйте функцию, которая выполняет операции над двумя матрицами (сложение, вычитание, умножение). Функция принимает две матрицы и символ операции, а возвращает результат операции. Используйте функции enumerate(), zip() и len()*

#esep 5

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

operations = {

'+': lambda a, b: a + b,

'-': lambda a, b: a - b,

'\*': lambda a, b: a \* b

}

result = [[0 for \_ in range(len(matrix1[0]))] for \_ in range(len(matrix1))]

for i in range(len(matrix1)):

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

result[i][j] = operations[operation](matrix1[i][j], matrix2[i][j])

return result

matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

matrix2 = [[9, 8, 7], [6, 5, 4], [3, 2, 1]]

operation = '+'

print(matrix\_operation(matrix1, matrix2, operation))

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

result = []

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

return None

for i, row1 in enumerate(matrix1):

row\_result = []

row2 = matrix2[i]

for j, val1 in enumerate(row1):

val2 = row2[j]

if operation == '+':

row\_result.append(val1 + val2)

elif operation == '-':

row\_result.append(val1 - val2)

elif operation == '\*':

row\_result.append(val1 \* val2)

else:

return None

result.append(row\_result)

return result

matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

matrix2 = [[9, 8, 7], [6, 5, 4], [3, 2, 1]]

operation = '+'

print(matrix\_operation(matrix1, matrix2, operation))