AI VIET NAM – COURSE 2024

Lập trình Python căn bản - Data structure

Ngày 17 tháng 6 năm 2024

Ngày thực hiện:	17/06/2024
Người thực hiện:	Đinh Thị Tâm
Nguồn:	AIO2024 - Week2
Nguồn dữ liệu (nếu có):	Link of Data Sources of week 2
Từ khóa:	Data structure
Người tóm tắt:	Đinh Thị Tâm

I. Câu hỏi tự luận

1. Câu 1:

   (a) Code

```
# bai 1: slicing window
def max_kernel(data, k):
    result = []
    length = len(data)-k
    for counter in range(length+1):
        result.append(max(data[counter:counter+k]))
    return result


# ham main
num_list = [3, 4, 5, 1, -44, 5, 10, 12, 33, 1]
k = 3
print(f'input: num_list = {num_list}, k={k}')
print(f'output: {max_kernel(num_list, k)}')


```

   (b) Kết quả thực thi **Input:**

```
            num_list = [3, 4, 5, 1, -44, 5, 10, 12, 33, 1], k=3
            output: [5, 5, 5, 5, 10, 12, 33, 33]

```

2. Câu 2

   (a) Code

```
# exercise 2: viet ham tra ve dictionary tra ve so luong chu xuat hien trong
    1 tu
# voi key la chu cai, value la so lan xuat hien
```

```
def count_chars(data):
    character_statistic = {}
    my_list = list(data)
    for x in my_list:
        character_statistic[x] = my_list.count(x)
    return character_statistic


# ham main
string = "Happiness"
print(count_chars(string))
string = "smiles"
print(count_chars(string))

```

(b) Kết quả thực thi

```
{'H': 1, 'a': 1, 'p': 2, 'i': 1, 'n': 1, 'e': 1, 's': 2}
{'s': 2, 'm': 1, 'i': 1, 'l': 1, 'e': 1}
```

3. Câu 3

(a) Code

```
# exercise 3: viet ham doc file text v   tra ve dictionary tra ve so luong tu
    xuat hien trong 1
# chuoi voi key la tu, value la so lan xuat hien
# input: duong dan den file
# output: dictionary dem so tu
# ham doc file text
import gdown


def count_word(filePath):
    counter = {}
    f = open(filePath, 'r')
    data = f.read()
    listData = data.split()
    tmp = set(listData)
    myList = list(tmp)
    myList.sort()
    for x in myList:
        counter[x] = listData.count(x)
    return counter


# ham main
url = 'https://drive.google.com/uc?id=1IBScGdW2xlNsc9v5zSAya548kNgiOrko'
file_path = 'content/P1_data01.txt'
gdown.download(url, file_path, quiet=False)

result = count_word(file_path)
print(f'result= {result}')


```

(b) Kết quả thực thi

```
{'A': 1, 'He': 1, 'Just': 1, 'One': 2, 'Success': 1, 'The': 1, 'Try': 1, 'We
    ': 1, 'You': 1, 'a': 6, 'again': 1, 'and': 1, 'are': 1, 'at': 1, 'be': 1,
    'become': 2, 'bricks': 1, 'busy': 1, 'but': 1, 'came': 1, 'can': 3, '
    cannot': 1, 'change': 1, 'comes': 2, 'conquers': 1, 'courage': 1, 'day':
    1, 'different': 1, 'employed': 1, 'enough': 1, 'everything': 1, 'firm': 1,
     'for': 3, 'foundation': 1, 'from': 1, 'get': 2, 'have': 1, 'help': 1, '
    him': 1, 'himself': 1, 'his': 2, 'if': 1, 'in': 4, 'is': 3, 'it': 2, 'just
    ': 1, 'kind': 1, 'lay': 1, 'life': 2, 'looking': 1, 'majority': 1, 'makes
    ': 1, 'man': 6, 'mightiest': 1, 'mistakes': 1, 'morning': 1, 'not': 1, 'of
    ': 4, 'one': 2, 'opportunity': 1, 'other': 1, 'others': 1, 'people': 1, '
    positive': 1, 'problems': 1, 'profit': 1, 'rather': 1, 'ready': 1, 'secret
    ': 1, 'small': 1, 'solve': 1, 'success': 2, 'successful': 2, 'the': 4, '
    them': 1, 'they': 1, 'thinking': 1, 'those': 1, 'thought': 1, 'thrown': 1,
     'to': 3, 'too': 1, 'try': 1, 'up': 1, 'usually': 1, 'value': 1, 'want':
    2, 'warrior': 1, 'way': 1, 'we': 2, 'what': 1, 'when': 2, 'who': 3, 'whole
    ': 1, 'will': 2, 'with': 4, 'you': 2, 'your': 1}
```

4. Câu 4

   (a) Code

```python
# khoi tao ma tran D
def levenshtein_distance(source, target):
    D = []
    M = len(source)+1  # M hang
    N = len(target)+1  # N cot
    D = [[0]*N for i in range(M)]
    # B2: cap nhat hang dau tien
    a = 0
    b = 0
    c = 0
    for k in range(N):
        D[0][k] = k
        if k < M:
            D[k][0] = k
    # B3 tinh toan cac gia tri bat dau tu D(1,1)
    for t1 in range(1, M):
        for t2 in range(1, N):
            if (source[t1-1] == target[t2-1]):
                D[t1][t2] = D[t1-1][t2-1]
            else:
                a = D[t1][t2-1]
                b = D[t1-1][t2]
                c = D[t1-1][t2-1]
                if (a <= b and a <= c):
                    D[t1][t2] = a+1
                elif (b <= a and b <= c):
                    D[t1][t2] = b+1
                else:
                    D[t1][t2] = c+1
    return D[M-1][N-1]


# ham main
print(f'Cac buoc chinh sua tu "yu" => "you" la: {
      levenshtein_distance("yu", "you")}')

```

   (b) Kết quả thực thi

```
Cac buoc chinh sua tu "yu" => "you" la: 1

```

II. Câu hỏi trắc nghiệm

1. Câu 1:

```python
        # bai 1: slicing window
def max_kernel(data, k):
    result = []
    length = len(data)-k
    for counter in range(length+1):
        result.append(max(data[counter:counter+k]))
    return result

# ham main
assert max_kernel([3, 4, 5, 1, -44], 3) == [5, 5, 5]
num_list = [3, 4, 5, 1, -44, 5, 10, 12, 33, 1]
k = 3
print(max_kernel(num_list, k))

```

2. Câu 2:

```python
        # exercise 2: viet ham tra ve dictionary tra ve so luong chu xuat hien trong 1 tu
# voi key la chu cai, value la so lan xuat hien
def character_count(data):
    character_statistic = {}
    my_list = list(data)
    for x in my_list:
        character_statistic[x] = my_list.count(x)
    return character_statistic


# ham main
assert character_count("Baby") == {'B': 1, 'a': 1, 'b': 1, 'y': 1}
print(character_count('smiles'))

```

3. Câu 3:

```python
        # exercise 3: viet ham doc file text v   tra ve dictionary tra ve so luong tu xuat hien trong 1
# chuoi voi key la tu, value la so lan xuat hien
# input: duong dan den file
# output: dictionary dem so tu
# ham doc file text
import gdown


def count_word(filePath):
    counter = {}
    f = open(filePath, 'r')
    data = f.read()
    listData = data.split()
    tmp = set(listData)
    myList = list(tmp)
    myList.sort()
    for x in myList:
        counter[x] = listData.count(x)
    return counter

```

```

# ham main
url = 'https://drive.google.com/uc?id=1IBScGdW2xlNsc9v5zSAya548kNgiOrko'
file_path = 'content/P1_data01.txt'
gdown.download(url, file_path, quiet=False)
# print(getCounterWord(file_path))
# file_path = 'content/P1_data.txt'
result = count_word(file_path)
assert result['who'] == 3
print(result['man'])

```

4. Câu 4:

```
        # khoi tao ma tran D
def levenshtein_distance(source, target):
    D = []
    M = len(source)+1  # M hang
    N = len(target)+1  # N cot
    D = [[0]*N for i in range(M)]
    # B2: cap nhat hang dau tien
    a = 0
    b = 0
    c = 0
    for k in range(N):
        D[0][k] = k
        if k < M:
            D[k][0] = k
    # B3 tinh toan cac gia tri bat dau tu D(1,1)
    for t1 in range(1, M):
        for t2 in range(1, N):
            if (source[t1-1] == target[t2-1]):
                D[t1][t2] = D[t1-1][t2-1]
            else:
                a = D[t1][t2-1]
                b = D[t1-1][t2]
                c = D[t1-1][t2-1]
                if (a <= b and a <= c):
                    D[t1][t2] = a+1
                elif (b <= a and b <= c):
                    D[t1][t2] = b+1
                else:
                    D[t1][t2] = c+1
    return D[M-1][N-1]


# ham main
assert levenshtein_distance("hi", "hello") == 4.0
print(levenshtein_distance("hola", "hello"))
```

5. Câu 5:

```
        def check_the_number (N):
    list_of_numbers = []
    result = ""
    for i in range (1, 5):
    # Your code here
    #Su dung append them i vao trong list_of_number
        list_of_numbers.append(i)
    if N in list_of_numbers:
        results = "True"
    if N not in list_of_numbers :
        results = "False"
    return results
N = 7
assert check_the_number (N) == "False"
N = 2
results = check_the_number (N)
print ( results )
```

6. Câu 6:

```
def my_function(data, max, min):
    result = []
    for i in data:
        # Your code here
        # Neu i < min thi them min vao result
        if i < min:
            result.append(min)
        elif i > max:
            result.append(max)
        else:
            result.append(i)
    return result


my_list = [5, 2, 5, 0, 1]
max = 1
min = 0
assert my_function(max=max, min=min, data=my_list) == [1, 1, 1, 0, 1]
my_list = [10, 2, 5, 0, 1]
max = 2
min = 1
print(my_function(max=max, min=min, data=my_list))
```

7. Câu 7:

```
     def my_function(x, y):
    x.extend(y)
    # Your code here
    # Su dung extend de noi y vao x
    return x


list_num1 = ['a', 2, 5]
list_num2 = [1, 1]
list_num3 = [0, 0]
assert my_function(list_num1, my_function(list_num2, list_num3)) == ['a', 2, 5,
    1, 1,
                                                                     0, 0]

list_num1 = [1, 2]
list_num2 = [3, 4]
list_num3 = [0, 0]

print(my_function(list_num1, my_function(list_num2, list_num3)))
```

8. Câu 8:

```
def my_function(n):
    return min(n)


my_list = [1, 22, 93, -100]
assert my_function(my_list) == -100

my_list = [1, 2, 3, -1]
print(my_function(my_list))
```

9. Câu 9:

```
def my_function(n):
    return max(n)


my_list = [1001, 9, 100, 0]
assert my_function(my_list) == 1001

my_list = [1, 9, 9, 0]
print(my_function(my_list))

```

10. Câu 10:

```
def My_function(integers, number=1):
    return any(item == number for item in integers)


my_list = [1, 3, 9, 4]
assert My_function(my_list, -1) == False

my_list = [1, 2, 3, 4]
print(My_function(my_list, 2))

```

11. Câu 11:

```
def my_function(list_nums=[0, 1, 2]):
    var = 0
    for i in list_nums:
        var += i
    return var/len(list_nums)


assert my_function([4, 6, 8]) == 6
print(my_function())
```

12. Câu 12:

```
def my_function ( data ):
    var = []
    for i in data :
        if i%3==0:
            var.append(i)
    return var

assert my_function ([3 , 9, 4, 5]) == [3, 9]
print ( my_function ([1 , 2, 3, 5, 6]))

```

13. Câu 13:

```python
def my_function(y):
    var = 1
    while (y > 1):
        var *= y
        y = y-1
    return var


assert my_function(8) == 40320
print(my_function(4))
```

14. Câu 14:

```python
def my_function(x):
 # your code here
    y = str()
    i = -1
    while x[i] != x[0]:
        y = y+x[i]
        i = i-1
    y = y+x[0]
    return y


x = 'I can do it'
assert my_function(x) == "ti od nac I"

x = 'apricot'
print(my_function(x))
```

15. Câu 15:

```python
def function_helper(x):
    if x > 0:
        return 'T'
    return 'N'


def my_function(data):
    res = [function_helper(x) for x in data]
    return res


data = [10, 0, -10, -1]
assert my_function(data) == ['T', 'N', 'N', 'N']

data = [2, 3, 5, -1]
print(my_function(data))
```

16. Câu 16:

```python
def function_helper(x, data):
    for i in data:
        if x == i:
            return 0
    return 1


def my_function(data):
    res = []
    for i in data:
        if function_helper(i, res):
            res.append(i)
    return res


lst = [10, 10, 9, 7, 7]
assert my_function(lst) == [10, 9, 7]

lst = [9, 9, 8, 1, 1]
print(my_function(lst))

```