

Gedächtnisprotokoll - BM14: Python WS2025/2026

1. Given a list of strings, find the three most common words and their frequencies.

Lösung:

```
def top_three_words(words):
    word_counts = {}
    for word in words:
        if word not in word_counts:
            word_counts[word] = 1
        else:
            word_counts[word] += 1
    sorted_words = sorted(word_counts.items(),
                           key=lambda x: x[1],
                           reverse=True)

    return sorted_words[:3]

word_list = ["apple", "banana", "apple",
             "orange", "banana", "apple",
             "grape", "grape", "banana"]
print(top_three_words(word_list))
```

2. Given a dictionary, remove all the items with a key length greater than 5.

Lösung:

```
example_dict = {'apple': 3, 'banana': 5, 'cherry': 8, 'watermelon': 12}

filtered_dict = dict()
for key in example_dict.keys():
    if len(key) <= 5:
        filtered_dict[key] = example_dict[key]
print(filtered_dict)
```

Or different Variant:

```
example_dict = {'apple': 3, 'banana': 5, 'cherry': 8, 'watermelon': 12}

filtered_dict = {key: value for key, value in example_dict.items() if len(key)
<= 5}
print(filtered_dict)
```

3. ## 3) String Handling

Write a Python program that takes a sentence as input and returns the sentence with each word reversed (e.g., "dog bites man" -> "god setib nam"). Please implement a function called `reverse\_words` that reverses each word. Please note that only the characters of each word need to be reversed but the order of words in the sentence should stay like the original sentence.

Lösung:

```
def reverse_words(sentence):
```

```

words = sentence.split()
reversed_words = [word[::-1] for word in words]
reversed_words = ' '.join(reversed_words)
return reversed_words

# Example usage
input_sentence = "dog bites man"
result = reverse_words(input_sentence)
print("Reversed words:", result)

```

4. For-Loops: Matrix handling:

```
mat = [[1, 2, 3, 4, 5], [1, 2, 3, 4, 5], [1, 2, 3, 4, 5], [1, 2, 3, 4, 5], [1, 2, 3, 4, 5]]
```

Bilden der Summe über das 2. und 3. Element der Reihen der Matrix.

Also $2+2+2+2+2+3+3+3+3+3=25$

Mögliche Lösung:

```

index_sum = 0

for row in mat:
    row_sum = sum(row[1:3])
    index_sum += row_sum

print(index_sum)

```

5. While Loops and User Input:

Look at the code

a) are there any mistakes if yes correct the mistakes

b) what does the output look like? (Annahme input is "Mueller" und es wird nie "q" eingegeben)

Was a code with a while-loop and some user input

Code sah in etwa so aus:

```

def get_formatted_name (first_name = "Jack", last_name):
    full_name = first_name + " " + last_name
    return full_name.title()

while True:
    print("/n get formatted name")
    print("enter 'q' to quit")
    f_name = input("Enter name: ")
    if f_name == 'q':
        break
    formatted_name = get_formatted_name(f_name, l_name)
    print("Hello:" + full_name)

```

Fehler waren zum Beispiel:

- Parameter mit default value müssen immer als letztes stehen
- Unten (Line 8+9) statt f_name müsste l_name stehen, da man ja nur einen Input hat und f_name ein default value hat ...
- In der letzten Zeile wurde statt der zuvor definierten Variable formatted_name etwas falsches gecallt...

6. Classes:

Create a class called Post with the attributes author, content, and likes. Use a constructor... Add a method increase_like() that increments the like count by 1. Finally, instantiate the class with sample data...

Mögliche Lösung:

```
class Post:
    def __init__(self, author, content, likes=0):
        self.author = author
        self.content = content
        self.likes = likes

    def increase_like(self):
        self.likes += 1

user = "Alice"
example_content = "This - is - a - sample - post"

#Initialize Class
post = Post(user, example_content)
```