Document 30 - python notes.

Python Programming (Gujarat Technological University)

Qs 3

A) Write a program to find the number of times an element occurs in the list.

Ans-

```python
def count_occurrences(lst, element):
    count = 0
    for item in lst:
        if item == element:
            count += 1
    return count
my_list = [1, 2, 3, 4, 2, 2, 5, 2]
target_element = 2
occurrence_count = count_occurrences(my_list, target_element)
print(f"The element {target_element} occurs {occurrence_count} times in the list.")
```

Output-

```
The element 2 occurs 4 times in the list.
```

B) Differentiate between append() and extend() methods of list.

Ans-

	Append()	Extend()
Define	Adds a single element to the end of the list.	Adds the elements of another list to the end of the list.
Arguments	Takes a single element as an argument.	Takes a list as an argument.
Effects	The element is added to the end of the list. The length of the list increases by 1.	The elements of the other list are added to the end of the list.
Time Complexity	O(1)	O(k)
Example	list1 = [] list1.append(1) list1.append(2) list1.append(3) print(list1)	list2 = [4, 5, 6] list1.extend(list2) print(list1)

C) Write an automated censor script that reads the text from a file and creates a new file where all of

the four-letter words have been replaced by “****” python.

Ans-

```python
def censor_four_letter_words(input_file, output_file):
    with open(input_file, 'r') as file:
        text = file.read()
    censored_text = ' '.join('****' if len(word) == 4 else word for word in text.split())
    with open(output_file, 'w') as file:
        file.write(censored_text)
input_file = 'input.txt'
output_file = 'output.txt'
censor_four_letter_words(input_file, output_file)
print("Censoring complete.")
```

Output-

```
loads/pyhton/1.py
Censoring complete.
```

OR

A) Write syntax of if-else and nested if-else

Ans.

If-Else Syntax-

```
if condition:
    # code block to execute
if condition is True
else:
    # code block to execute
if condition is False
```

Nested If Else Syntax-

```
if condition1:
    # code block to execute
if condition1 is True
    if condition2:
        # code block to
execute if condition2 is also
True
    else:
        # code block to
execute if condition2 is
False
else:
    # code block to execute
if condition1 is False
```

B) Explain basic tuple operations with example.

Ans-

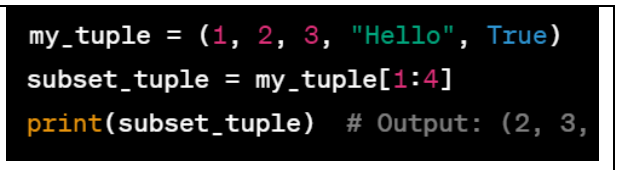
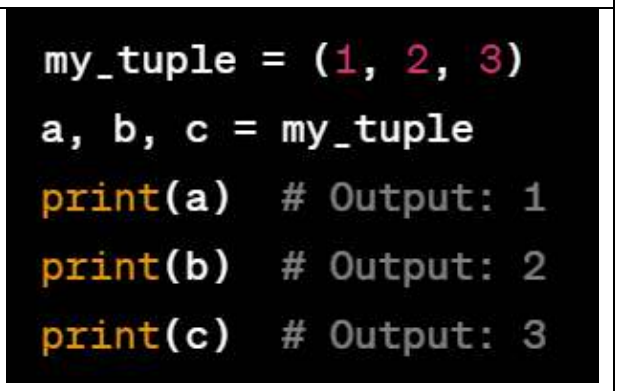
No	Operations	Example
1	Creating a Tuple: You can create a tuple by enclosing a sequence of elements in parentheses "()".	`my_tuple = (1, 2, 3, "Hello", True)`

2	Accessing Elements: You can access individual elements of a tuple using indexing, similar to lists. The index starts from 0.	`my_tuple = (1, 2, 3, "Hello", True)`
		`print(my_tuple[0])  # Output: 1`
		`print(my_tuple[3])  # Output: Hello`

3	Tuple Concatenation: You can concatenate two or more tuples together using the "+" operator. It creates a new tuple containing all the elements from the original tuples.	`tuple1 = (1, 2, 3)`
		`tuple2 = ("Hello", "World")`
		`concatenated_tuple = tuple1 + tuple2`
		`print(concatenated_tuple)  # Output:`

4	Tuple Repetition: You can repeat a tuple multiple times using the "*" operator. It creates a new tuple with the repeated elements.	`my_tuple = ("Hello", "World")`
		`repeated_tuple = my_tuple * 3`
		`print(repeated_tuple)  # Output:`

5	Tuple Length: You can find the number of elements in a tuple using the len() function.	`my_tuple = (1, 2, 3, "Hello", True)` `print(len(my_tuple))  # Output: 5`

6	Tuple Slicing: You can extract a subset of elements from a tuple using slicing. It returns a new tuple containing the specified elements.	
7	Unpacking a Tuple: You can assign individual elements of a tuple to separate variables using tuple unpacking. The number of variables must match the number of elements in the tuple.	

C) Write a program to randomly fill in 0s and 1s into a 4x4 2-dimension list, print the list and find the rows and columns with the most number of 1s.

Ans-

```python
import random
matrix = [[0] * 4 for _ in range(4)]
for i in range(4):
    for j in range(4):
        matrix[i][j] = random.randint(0, 1)
for row in matrix:
    print(row)
max_ones_row = 0
max_ones_row_count = 0
for i, row in enumerate(matrix):
    count = row.count(1)
    if count > max_ones_row_count:
        max_ones_row_count = count
        max_ones_row = i
max_ones_column = 0
max_ones_column_count = 0
for j in range(4):
    count = sum(matrix[i][j] for i in range(4))
    if count > max_ones_column_count:
        max_ones_column_count = count
        max_ones_column = j
print("Row with the most number of 1s:", max_ones_row)
print("Column with the most number of 1s:", max_ones_column)
```

Output-

```
[1, 1, 1, 1]
[1, 0, 0, 1]
[0, 1, 1, 1]
[0, 1, 0, 0]
Row with the most number of 1s: 0
Column with the most number of 1s: 1
```