ASSIGNMENT 3

Name:- Harsh Balaprasad Baheti

Div :- A

Roll No.:-SECOA112

Assignment 3

Objective :

To understand the different string manipulation and operations.

Outcome :

Students will be able to perform different string operations.

Theory:

What is String:

The string can be defined as the sequence of characters represented in the quotation marks.
In python, we can use single, double, or triple quotes to define a string.
In the case of string handling, the operator + is used to concatenate two strings as the operation "hello"+" python" returns "hello python".
The operator * is known as repetition operator as the operation "Python " *2 returns "Python Python "

String in Python with example:

```
str1 = 'hello rahul' #string str1
str2 = ' how are you' #string str2
print (str1[0:2]) #printing first two character using slice operator
print (str1[4]) #printing 4th character of the string
print (str1*2) #printing the string twice
print (str1 + str2) #printing the concatenation of str1 and str2
```

Harsh Baheti

Output:

he

o

hello rahul hello rahul

hello rahul how are you

Problem Statement :

Write a Python program to compute following operations on String:

a) To display word with the longest length

b) To determines the frequency of occurrence of particular character in the string

c) To check whether given string is palindrome or not

d) To display index of first appearance of the substring

e) To count the occurrences of each word in a given string

Algorithm :

1. longtest_word Function:-

Step 1) start

Step 2) Intilise the longest as empty string.

Step 3) for every i element in spilted list of input_string jump to step4 else step6.

Step 4) if length of i is greater than longest jump to step5 else step3.

Step 5) set i to longest jump to step 3.

Harsh Baheti

Step 6) Return longest

Step 7) exit()

2. frequency_chr Function:-

Step 1) start

Step 2) Intilise the dictionary as an empty dictionary.

Step 3) for every i character in input_string jump to step4 else step5.

Step 4) if key i is in dictionary then increment its value by 1 else set to 1 jump to step3.

Step 5) Return dictionary

Step 6) exit()

3. palindrome Function:-

Step 1) start

Step 2) for every i range from 0 to half the length of string +1 jump to 3 else jump to 5

Harsh Baheti

Step 3) if chr at 'i' and 'n-i-1' are same then jump to 2 else jump to 4.

Step 4) return "not a palindrome" jump to 6.

Step 5) return "a palindrome" jump to 6.

Step 6) exit()

4. first_indexFunction:-

Step 1) start

Step 2) if substr in input_string jump to 3 else jump to 4.

Step 3) return input_string.find(substr) jump to 5.

Step 4) Return "substring not present in string" jump to 5.

Step 5) exit()

5.Counter Function:-

Step 1) start

Harsh Baheti

Step 2) Intilise the dictionary as an empty dictionary.

Step 3) for every i in spilted string jump to step4 else step5.

Step 4) if key i is in dictionary then increment its value by 1 else set to 1 jump to step3.

Step 5) Return dictionary

Step 6) exit()

Program/Code:

```
# a) To display word with the longest length
def longtest_string(input_string):
    longest = ""
    for i in input_string.split():
        if len(longest) < len(i):
            longest = i
    return longest


# b) To determines the frequency of occurrence of particular character in the
string
def frequency_chr(input_string):
    dictionary = {}

    for i in input_string:
        dictionary[i] = dictionary.get(i, 0) + 1
    return dictionary
```

Yogesh Baheti

```python
# c) To check whether given string is palindrome or not
def palindrome(input_string):
    for i in range(0, len(input_string) // 2 + 1):
        if input_string[i] == input_string[len(input_string) - i - 1]:
            continue
        else:
            return "not a palindrome"
    return ' a palindrome'


# d) To display index of first appearance of the substring
def first_index(input_string, substr):
    if substr in input_string:
        return input_string.find(substr)


# e) To count the occurrences of each word in a given string
def counter(input_string):
    dictionary = {}
    for i in input_string.split():
        dictionary[i] = dictionary.get(i, 0) + 1
    return dictionary


string = input("Enter String : ")
substring = input("Enter Substring : ")

print("Word with the longest length", longtest_string(string))
print("The frequencies of occurrence of particular character in the string",
frequency_chr(string))
print("The string '{}' is {}".format(string, palindrome(string)))
print("The index of first appearance of the substring ", first_index(string,
substring))
print("Count the occurrences of each word in a given string", counter(string))
```

Output :

```
Enter String : hello world it's harsh
Enter Substring : harsh
Word with the longest length hello
The frequencies of occurrence of particular character in the string {'h': 3,
'e': 1, 'l': 3, 'o': 2, ' ': 3, 'w': 1, 'r': 2, 'd': 1, 'i': 1, 't': 1, "'": 1,
's': 2, 'a': 1}
The string 'hello world it's harsh ' is not a palindrome
The index of first appearance of the substring  17
Count the occurrences of each word in a given string {'hello': 1, 'world': 1,
"it's": 1, 'harsh': 1}
```

Time Complexity :

sr no.	Function	Time Complexity
1	**longtest_word**	**O(n)**
2	**frequency_chr**	**O(n)**
3	**palindrome**	**O(n)**
4	**first_index**	**O(n)**
5	**counter**	**O(n)**
6	**print**	**O(1)**

Total Time Complexity:= O(n)

Conclusion :-

Understood the different string manipulation and operations.

Harsh Baheti