Jaydeep Mohapatra
2141013274
Sec-X

ASSIGNMENT-6

(Q1)→ WAP that takes a string as a parameter and returns a string with every successive character replaced with star (*).

```
a=input("Enter a string:")
modif = " "
    for i in a:
        if i in modif:
            modif= " *"
        else:
            modif +=i
        print(modif)
```

O/t:- Enter the string : MISSISSIPPI
MIS** * ** P **

(Q2)→ WAP that takes two string and return True if they are anagram or false otherwise

```
def anagram (str1, str2):
        str1= str1.replace (" ", " ").lower()
        str2= str2.replace (" ", " ").lower()
        return sorted(str1)== sorted (str2)
    str1= " listen"
    str2= " silent"
    res= anagram (str1, str2)
print(res)
```

(Q3)→ WAP that takes a sentence as input and display no. of words in a sentence.

```
def count(sent):
    word = sent.split()
    return len(word)
sent = "This is a simple sentence"
word = Count(sent)
print("The no. of words are", word)
```

(Q4) →

```
def caps(sent):
    words = sent.split()
    for i in range(len(words)):
        word = words[i]
        if word:
            words[i] = word[0].upper() + word[1:].lower()
    capital = ''.join(words)
    return capital
sent = "This is a simple sentence"
caps = caps(sent)
print("Capitalized sentence:", caps)
```

(Q5) WAF that takes a string as input and determine the count of the numbers of words using re.

```
import re
def wordcount(sent):
    word = re.compile(r'\b\w+\b')
    match = word.findall(sent)
    return (match)
send = "This is me."
word = wordcount(sent)
print("The no. of words are:", word)
```

⑥ What will be the output on executing each of the statements following the assignment statement?

address = 'B-6, Lodhi road, Delhi'

(a) len(address) → 20

(b) address[17:-1] → en

(c) address[-len(address):len(address)] → B-6, Lodhi road, Delhi

(d) address[:-12] + address[-12:] → B-6, Lodhi road, Delhi

(e) address.find('delhi') → -1

(f) address.swapcase() → b-6, LODHI ROAD, DELHI

(g) address.split(',') → ['B-6', 'Lodhi road', 'Delhi']

(h) address.isalpha() → false

⑦ Examine the following string:

greeting = 'Good Morning. Have a Good Day!!'

(a) greeting.count('hood')
↳ 2

(b) greeting.find('a')
↳ 15

(c) greeting.rfind('a')
↳ 27

(d) greeting.capitalize()
↳ Good morning. Have a good day!!

(e) greeting.lower()
↳ good morning. have a good day!!

(f) greeting.upper()
↳ GOOD MORNING. HAVE A GOOD DAY!!

(g) greeting.swapcase()
↳ good MORNING. HAVE A good DAY!!

(h) greeting.istitle()
↳ false

(i) greeting.replace('hood', 'sweet')
↳ Sweet Morning. Have a sweet Day!!

(j) greeting.strip()
↳ Good Morning, Have a Good Day!!
(k) greeting.split()
↳ ['Good', 'Morning,' 'Have', 'a', 'Good', 'Day!!']
(l) greeting.partition('.')
→ ('Good Morning', '.', 'Have a Good Day!!')
(m) greeting.startswith ('good')
→ False
(n) greeting.endswith ('!!')
→ True
(8) → i) determine the patterns extracted by the following regular expressions.

```
1. string1 = 'Python Programming Language'
1.match1 = re.search ('.....m?', string1)
print (match1.group())
→ Programm
2.match2 = re.search ('.....m{1,2}', string1)
print (match2.group())
→ Programm
3.match3 = re.search ('.*Language$', string1)
print (match3.group())
→ Python Programming Language
4.match4 = re.search ('\W*|s|w*', string1)
print (match4.group())
→ Python Programming
5. match5 = re.search ('.*', string1)
print (match5.group())
↳ Python Programming Language
```

(d)

2. string2 = 'Care Number DL5645'

1. match1 = re.search ('\w\w\s\d{3,4}', string2)
print(match1.group())
→ DL5645

2. match2 = re.search ('.*5', string2)
print(match2.group())
→ Care Number DL5645

3. match3 = re.search ('.*5?', string2)
print(match3.group())
→ Care Number DL5645

4. match4 = re.search ('\d{3}', string2)
print(match4.group())

(e)

→ 564

5. match5 = re.search ('^C.*5$', string2)
print(match5.group())
→ Care Number DL5645

3. string3 = 'cdcccdeddd8f3399aabb'

1. match1 = re.search ('(\d)*\d*(a|b)*', string3)
print(match1.group())
→ cdcccdeddd3f3399aabb

2. match2 = re.search ('(\d)*d', string3)
print(match2.group())

(f)

→ d

3. match3 = re.search ('(cc|d*d)* (3|4)* (aa|bb)', string3)
print(match3.group())
→ 3f3399aa

4. match4 = re.search ('(cc|cd|ddd)* (3|4)* (aa|bb)', string3)
print(match4.group())
→ cdcccdeddd3f3399aa

5. match5 = re.search ('(cc|cd|ddd)* (3|4)* (aa|bb)*', string3)
print(match5.group())
→ cdcccdeddd3f3399aabb

(a) → write a python program to perform the following tasks:

(a) → Reverse a string:→

```
n=input("Enter a string:")
length = len(n)
s1 = ''
for i in range (length-1, -1, -1):
        s1 = s1 + n[i]

print(s1)
```

(b) → Reverse a string, without reversing the words.

```
n = input("Enter the string")
s = n.split(' ')
k = ''
for i in range (len(s)-1, -1, -1):
        k = k + ' ' + s[i]

m = k.lstrip()
print(m)
```

(c) → check if a string is symmetric or assymetric

```
n = input("Enter the string")
s = n[0 : len(n)//2]
p = n[len(n)//2 : len(n)]
if (s == p):
    print("Symmetric")

else:
    print("Asymmetric")
```

(d) check if a string is palindrome.

```
n=input("Enter a string:")
m=n
length=len(n)
s1=''
for i in range(length-1,-1,-1):
        s1=s1+n[i]
if (s1==n):
        print("Yes palindrome")
else:
        print("Not palindrome")
```

(e) → Given a string s and index i, delete ith value from s.

```
s=input("Enter the string:")
n=len(s)
i=int(input("Enter the index:"))
l=s[0:i]
m=s[i+1:n+1]
k=l+m
print(k)
```

(f) → count the no. of vowels and consonants in a string.

```
s=input("Enter the string:")
n=len(s)
count=0
for i in s:
        if (i=='a' or i=='e' or i=='i' or i=='o' or i=='u'):
                count=count+1

vowel=count
consonant=n-count
print("No. of vowel is:",vowel)
print("No of consonant is:",consonant)
```

(g) → find length of a string without using inbuilt function.

```
s=input("enter a string:")
count=0
for i in s:
    count=count+1
print(count)
```

(h) → check if a string contains at least one digit and one alphabet.

```
s=input("enter the string:")
c=0
c1=0
for i in s:
    if(i.isdigit()):
        c=c+1
    elif(i.isalpha()):
        c1=c1+1
    else
        print("no")
if(c!=0 and c1!=0):
    print("true")
else:
    print("false")
```

(i) Remove duplicate from a string

```
s=input("enter the string:")
n=len(s)
j=0
for i in s:
    if(i not in j):
        j=j+1
print(j)
```

(j) count frequency of characters in a string.

```
s = input("Enter a string")
a = ""
for i in s:
    if (i not in a):
        a = a + i
        print(s.count(i) + "times" + i)
```

(k) Find the character having max frequency in a string.

```
s = input("Enter a string")
a = ""
l = 0
c = ""
for i in s:
    if (s.count(i) > l):
        l = s.count(i)
        c = i
print("The character having max frequency is:", c)
```

(l) check if a pair is anagram.

```
s = input("Enter a string")
a = input("Enter a string")
f = 1
for i in s:
    if (i not in a):
        f = 0
    elif (len(s) != len(a)):
        f = 0
if (f == 0):
    print("Not anagram")
else:
    print("Anagram")
```