SIKSHA 'O' ANUSANDHAN
DEEMED TO BE UNIVERSITY

Admission Batch: **Session:**

Laboratory Record

Programming in Python (CSE 3142)

Submitted by

Name: Saswat Mohanty

Registration No.: 1941012407

Branch: Computer Science and Engineering

Semester: 5th Section: 'D'

Department of Computer Science & Engineering

Faculty of Engineering & Technology (ITER)

Jagamohan Nagar, Jagamara, Bhubaneswar, Odisha - 751030

INDEX

Sl. No	Name of Program	Page No	Remarks
1.	Minor Assignment - 1	1-8	good 06/12/21
2.	Minor Assignment - 2	9-21	good 06/12/21
3.	Minor Assignment - 3	22-39	good 06/12/21
4.	Minor Assignment - 5	40-45	
5.	Minor Assignment - 6	46-57	
6.	Minor Assignment - 7	58-	

Minor Assignment – 6

Strings

Q1) Write a function that takes a string as a parameter and returns a string with every successive repetitive character replaced with a star (*). For example, 'balloon' is returned as 'bal*o*n'.

Program :-

```
def replaceWithStar(str):
    stores = ''
    for i in range(0, len(str)):
        if str[i] == str[i-1]:
            stores = stores[:i+1] + '*'
        else:
            stores += str[i]
    return stores
print(replaceWithStar('balloon'))
```

Output :-

```
bal*o*n
```

Q2) Write a function that takes two strings and returns True if they are anagrams and False otherwise. A pair of strings is anagrams if the letters in one word can be arranged to form the second one.

Program :-

```
def checkAnagrams(str, str1):
    resstr = ''.join(sorted(str))
    resstr1 = ''.join(sorted(str1))
```

```
    if (reostr == reostr1):
        return True
    else:
        return False
print (checkAnagrams ("kutta", "tutka"))
```

Output :-

True.

Q3) Write a function that takes a sentence as an input parameter & displays the number of words in the sentence.

Program :-

```
def countWord (str):
    return str.count (" ") + 1
print (countWord ("Hello World This is Saswat"))
```

Output :-

5

Q4) Write a function that takes a sentence as input parameter & replace the first letter of every word with the corresponding uppercase letter & rest of the letters in the word by corresponding letters in lowercase without using builtin function.

Program :-

```
def convert (str):
    ch = list (str)
    for i in range (len (str)):
        if (i == 0 and ch[i] != ' ' or ch[i] != '' and ch[i-1] = ' '):
            if (ch[i] >= 'a' and ch[i] <= 'z'):
                ch[i] = chr (ord (ch[i]) - ord ('a') + ord ('A'))
```

```
    elif[ ch[i] >='A' and ch[i] <='z']:
        ch[i] = chr(ord(ch[i])+ ord('a')- ord('A'))
    resstr =" ".join(ch)
    return resstr
print(convert("Hello WElcome 231"))
```

Output :-

Hello Welcome 231

Q5) Write a function that takes a string as an input and determines the count of the number of words without using regular expression.

Program :-

```
def countWords(s):
    count = 1
    for i in s:
        if i==" ":
            count +=1
    return count
print(countWords("Hello World This is Saswat"))
```

output :-

5

Q6) What will be the output on executing each of statements, following the assignment statement:

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

a) len(address)

b) address[17:-1]

c) address[-len(address):len(address)]

d) address[:-12]+address[-12:]

e) address.find ('delhi')
f) address.swapcase()
g) address.split (',')
h) address.isalpha()

Program :-

```
address = 'B-6, Lodhi road, delhi'.
print (len(address))
print (address[17:-1])
print (address[:-12] + address[-12:])
print (address.find ('delhi'))
print (address.swapcase())
print (address.split (','))
print (address.isalpha())
```

Output :-

```
B-6, Lodhi road, delhi
B-6, Lodhi road, delhi
-1
b-6, lODHI ROAD, dELHI
['B-6', ' Lodhi road', ' delhi']
False
```

Q7) Examine the following string:
greeting = 'Good Morning. Have a Good Day!!'
What will be the output for the following function calls:
a) greeting.count ('Good')
b) greeting.find ('a')

c) greeting.rfind ('a')

d) greeting.capitalize()

e) greeting.lower()

f) greeting.upper()

g) greeting.swapcase()

h) greeting.istitle()

i) greeting.replace ('good', 'sweet')

j) greeting.strip()

k) greeting.split()

l) greeting.partition ('.')

Program :-

```
greeting = 'good Morning. Have a good Day!!'
print (greeting.find ('a'))
print (greeting.count ('good'))
print (greeting.rfind ('a'))
print (greeting.capitalize())
print (greeting.lower())
print (greeting.upper())
print (greeting.swapcase())
print (greeting.istitle())
print (greeting.replace ('good', 'sweet'))
print (greeting.strip())
print (greeting.split())
print (greeting.partition ('.'))
```

```
print (greeting. startswith ('good'))
print (greeting. endswith ('!!'))
```

Output :-

```
good morning. have a good day!!
GOOD MORNING. HAVE A GOOD DAY!!
good MORNING. HAVE A good day!!
False
Sweet Morning. Have a sweet day!!
Good Morning. Have a good day!!
['good', 'Morning.', 'Have', 'a', 'good', 'day!!']
('good Morning', '.', ' Have a good day!!')
False
True.
```

Q8) Determine the patterns extracted by the following regular expressions.

```
1. string1 = 'Python Programming Language'

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

2. string2 = 'car Number DL5645'

1. match 1 = re.search ('\D\D?\d{1,4}', string2)
print (match1.group())

2. match 2 = re.search ('.*5', string2)
print (match2.group())

3. match 3 = re.search ('.*5?', string2)
print (match3.group())

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

5. match 5 = re.search ('^C.*5$', string2)
print (match 5.group())

3. string3 = 'cdcccdcddd 343344aabb'

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

2. match 2 = re.search ('c(cd)*d', string3)
print (match2.group())

3. match 3 = re.search ('c(cc|cd)*(3|4)*(aa|bb)', string3)
print (match 3.group())

4. match 4 = re.search ('c(cc|cd|dd)*(3|4)*(aa|bb)', string3)
print (match 4.group())

5. match 5 = re.search ('c(cc|cd|dd)*(3|4)*(aa|bb)*', string3)
print (match 5.group())

Program :-

```
import re
string1 = 'Python Programming Language'
match1 = re.search ('.....m?', string1)
print (match1.group())
```

```
match2 = re.search('.....m{1,2}', string1)
print(match2.group())
match3 = re.search('.*language$', string1)
print(match3.group())
match4 = re.search('\w*\s\w*', string1)
print(match4.group())
match5 = re.search('.*', string1)
print(match5.group)
string2 = 'car number DL5645'
match1 = re.search('\w\w?\d{1,4}', string2)
print(match1.group)
match2 = re.search('.*5', string2)
print(match2.group)
match3 = re.search('.*5?', string2)
print(match3.group())
match4 = re.search('\d{3}', string2)
print(match4.group())
match5 = re.search('^c.*5$', string2)
print(match5.group())
string3 = 'cdcccdcdcdd34 33344 aabb'
match1 = re.search('(c|d)*\d*(a|b)*', string3)
print(match1.group())
match2 = re.search('(cd)*d', string3)
print(match2.group())
match3 = re.search('(cc|dd)*(3|4)*(aa|bb)', string3)
print(match3.group())
match4 = re.search('(cc|cd|dd)*(3|4)*(aa|bb)', string3)
```

```
print (match4.group())
match5 = re.search('(cc|cd|dd)*(3|4)*(aa|bb)*', string3)
print (match5.group())
```

output :-

```
DL5645
Car Number DL5645
Car Number DL5645
564
Car Number DL5645
cdcccdcddd 343344aabb
d
343344aa
cdcccdcddd 343344aa
cdcccdcddd 343344aabb
```

Q9) Write a python program to perform the following tasks :

1. Reverse a string
2. Reverse a string without reversing the words. For Example:
   input data : 'welcome to iter'
   output : 'iter to welcome
3. check if a string is symmetric or asymmetric
4. check if a string is palindrome.
5. Given a string s and index i, delete i^{th} value from s.
6. count the number of vowels and consonants in a string.
7. Find length of a string without using inbuilt function.
8. check if a string contains at least one digit and one alphabet
9. Remove duplicates from a string
10. count frequency of characters in a string.

11. Find the character having maximum frequency in a string.
12. check if a pair of strings are anagram to each other.
Example: SILENT and LISTEN are anagrams.

Program :-

```
def a1(str):
    print(str[::-1])
def a2(str):
    list = str.split()
    print(" ".join(list[::-1]))
def a3(str):
    if str[:len(str)//2] == str[(len(str)//2):]:
        print("symmetric")
    else:
        print("Asymmetric")
def a4(str):
    if str == str[::-1]:
        print("Palindrome")
    else:
        print("Not a palindrome")
def a5(str,i):
    print(str[:i]+str[i+1:])
def a6(str,vowel):
    str = str.lower()
    list = [each for each in str if each in vowel]
    print(len(list)," ", len(str)-len(list))
```

```
def a7(str):
    c=0
    for i in str:
        c+=1
    print(c)
def a8(str):
    print(str.isalnum())
def a9(str):
    print("".join(set(str)))
def a10(str):
    temp={}
    for i in str:
        if i in temp:
            temp[i]+=1
        else:
            temp[i]=1
    return temp
def a11(str):
    dict=a10(str)
    max_freq=max(dict, key=dict.get)
    print(max_freq)
def a12(str,str1):
    print(sorted(str)==sorted(str1))
if __name__=='__main__':
    a1("welcome to iter")
    a2("welcome to iter")
```

```
a3("khokcho")
a4("amaama")
a5("hello", 2)
a6("amama", "aeiou")
a7("welcome to iter")
a8("hey123")
a9("amaama")
print(a10("amaama"))
a11("amaama")
a12("silent", "listen")
```

Output :-

```
15
True
ma
{'a': 4, 'm': 2}
a
True.
```