Kalyani Government Engineering College

Department of Computer Application

Python Programming Lab – MCAN191
Year: 2023-2024
Semester: 1st Semester
Assignment 9
Name: Madhusudan Chand **Roll:** 13

9.1 Write a program that writes 10 random numbers to a file 'numbers.txt'. Each random number should be in the range of 1 through 100

```
Code :
import random as ra
f=open("random.txt","w")
for i in range(1,10):
    f.write(str(ra.randint(1,100)))
    f.write("\n")
print("Done!")
f.close()

Output:
Done!

random.txt
17
16
95
89
48
56
93
54
16
```

9.2 Write a program that reads and display all of the numbers stored in the file numbers.txt (created in question 1) and calculates their total.

```
Code :
```

```
f=open("random.txt","r")
sum=0
for i in f:
    print(i,end="")
    sum=sum+int(i)

print("Sum of the number stored in the file created in question 1 is",sum)
f.close()

Output:
17
16
95
89
48
56
93
54
16
Sum of the number stored in the file created in question 1 is 484
```

9.3 Write a function, digit_count() in Python that counts and displays the number of digits in the text file named 'sample.txt'

```
Code:
def digit_count(f):
    count=0
    while True:
        c=f.read(1)
        if not c:
            break
        if c.isnumeric():
            count+=1
    return count

f=open("sample.txt","r")
print("Number of digits",digit_count(f))
f.close()

Outout:
Number of digits 6
```

```
"sample.txt"
The team achieved a milestone in 2023. They completed a multi-million-dollar project
ahead of schedule.
Stakeholders were impressed with a 98% success rate.
```

9.4 Write a function lines_count() that reads lines from a text file named 'zen.txt' and displays the lines that begin with any vowel.

Code:

```
def lines_count(f):
    s="aeiouAEIOU"
    for i in f:
        if i[0] in s:
            print(i)

f = open("zen.txt","r")
print(lines_count(f))

f.close()
```

Output:

```
Explicit is better than implicit.
```

```
"zen.txt"
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
```

9.5 Assume that the file 'notes.txt' containing some text and exists on the computer's disk. Write a program that display only those words from 'notes.txt' file whose length is more than seven. Keep in mind that any punctuation marks at the beginning or end of a word should also be considered as part of the word's length.

Code :

```
f = open("notes.txt","r")
print("only those words from 'notes.txt' file whose length is more than seven")
for i in f:
    for j in range(len(i.split())):
        if len(i.split()[j])>7:
            print(i.split()[j],end=" ")

f.close()
```

```
Output:
only those words from 'notes.txt' file whose length is more than seven
struggle survival conquest existence, elimination subsistence. Computer6 imagination.
Outside, address. Beautiful Explicit implicit. complex. complicated.
```

```
"notes.txt"
If men wish to live, then they are forced to kill others.
The entire struggle for survival is a conquest of the means of existence,
which in turn results in the elimination of others from these same sources of subsistenc
The Computer6 hums softly as I sit with a Book3 in hand, diving into a world of
imagination. Outside, my friends gather at House9 and I quickly grab my Pen2 to jot down
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
```

9.6 Write a function last_digit_words() in Python to count the words ending with a digit in a text file "notes.txt"

```
Code:
def last_digit_words(f):
    count=0
    for i in f:
        m=i.split()
        for j in m:
            if j[-1:].isnumeric():
                count+=1
                print(j)
    print("Number of words ending with a digit are",count)

f=open("notes.txt","r")
last_digit_words(f)
```

```
Output:
Computer6
Book3
House9
Pen2
Number of words ending with a digit are 4
```

9.7 Assume that a file 'names.txt' containing a series of names (as strings) exists on the computer's disk. Write a function, first_five() that displays only the first five lines of the file's contents. If the file contains less than five lines, it should display the file's entire contents.

```
Code:
def first_five(f):
    j=0
    flag=0
    for i in f:
        if j>4:
            break
        j+=1
        print(i)
    if j<5:
        for i in f:
            print(i)
f=open("notes.txt","r")

first_five(f)

f.close()

Output:
If men wish to live, then they are forced to kill others.

The entire struggle for survival is a conquest of the means of existence,

which in turn results in the elimination of others from these same sources of subsistenc

The Computer6 hums softly as I sit with a Book3 in hand, diving into a world of

imagination. Outside, my friends gather at House9 and I quickly grab my Pen2 to jot down
```

9.8 Write a Python program that reads a text file and prints its contents in reverse order (from the last line to the first line).

```
Code:
f=open("zen.txt","r")
f1=list(f)
for i in f1[::-1]:
    j=(i.split()[::-1])
    print(" ".join(j))
f.close()

Output
complicated. than better is Complex
complex. than better is Simple
implicit. than better is Explicit
ugly. than better is Beautiful

"zen.txt"
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
```

9.9 Write the definition of a Python function named long_lines( ) which reads the contents of a text file named 'lines.txt' and displays those lines from the file which have at least 8 words in it

```
Code:
def long_lines(f):
    l=list(f)
    for i in l:
        m=i.split()
        if len(m)>7:
            print(i)

f=open("lines.txt","r")

long_lines(f)

f.close()
```

```
Output:
Special cases aren't special enough to break the rules.

"lines.txt"
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
```

9.10 Write a Python function named feedback_analysis() to calculate and display the following information:
Total feedbacks stored in the file.
Count of positive feedbacks.
Count of negative feedbacks.

```
Code:
def feedback_analysis(f):
    p_count=n_count=0
    for i in f:
        if i.split()[0]=="Positive:":
            p_count+=1
        elif i.split()[0]=="Negative:":
            n_count+=1
    print("Number of positive feedbacks",p_count)
    print("Number of positive feedbacks",n_count)


f=open("feedback.txt","r")

feedback_analysis(f)

f.close()

Output:
Number of positive feedbacks 2
Number of positive feedbacks 3

"feedback.txt"
Positive: Saksham improved grades, more confident now.
Negative: Arav needs better time management for coursework.
Negative: Samar should work on communication in group activities.
```

```
Negative: Soham could benefit from asking more questions in class.
Positive: Sakshi excels academically, a great team player.
```

9.11 Create a Python function make_copy() that reads a text file 'input.txt' and writes its contents to a new file 'output.txt', capitalizing the first letter of each word.

```
Code:
def make_copy(f):
    fw=open("ouput.txt","+x")
    for i in f:
        fw.write(i)
    print("Copied!")

f=open("input.txt","r")
make_copy(f)def make_copy(f):
    fw=open("ouput.txt","+x")
    for i in f:
        fw.write(i)
    print("Copied!")

f=open("input.txt","r")
make_copy(f)

Output:
Copied!

"input.txt"
In the world of programming, there are no limits to what you can achieve. Aim high!

"Output.txt"
In the world of programming, there are no limits to what you can achieve. Aim high!
```