Assignment No: 1

Title :- Write python code for Word count and find occurance of word from file.

Problem statement:-
To python code for word count and find occurance of word from file.

Pre-lab:- a basic understanding of computer programming languages will help in understanding the python programming any datascience concept.

Theory:-
What is python?
- python is powerful modern computer programming language it bears some similarities to fortran, one of the earliest programming languages but it is much more powerful than fortan
python allows you to use variables without declaring them (i.e. it determines types implicity.)
- python was developed by Guido von Rossum, and it is a free software.
- Python is a good choice for mathematical calculations, since we can write code quickly test it easily and it's syntax is similar to the way mathematical ideas are expressed in the mathematical literature.

Python commands:-
- comments in a python command anything after a # symbol is a comment.

For eg-

print "Hellow World" # this is siry
- comments are not part of the command, but rather intended as documentation for anyone reading the code
- multiple comment are also possible and are enclosed by triple double-quote symbol

Comments:-
- In python command, anything after a '#' symbol is a command for comment.
For example,
print " Hellow World" # this is silly
- multiple comments are also possible, and are enclosed by multiple triple double-quote symbol:
"This is an example of a long comment that goes on and on and on" "" ""

Numbers and other data type:-
- python recognize several different types of data For instance, 29 and -75 are integers, while 5.0 and -23.0g are floats or floating point numbers.
- The type float is (roughly) the same as a real number in mathematics.
- The type float is (roughly) the same as a real number in mathematics
- The number 12345678901 is a long integer

The type function:-
- To see the type of same data use pythons built in type function
>>> type (-75)
<type 'int'>

```
>>> type (5.0)
<type 'float'>
>>> type (1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1)
    <type 'long'>
```

- Another useful date type is complex Used for complex numbers.

String:-

- Other useful data types are strings (short for "character strings")

For example, "Hellow World"

- strings are sequence of characters enclosed in single or double quotes.

```
>>> "This is a string"
    This is a string
>>> type ("This a string")
      <type 'str'>
```

- String are an example of sequence type

* List & tuples:-

- Other important sequence types used in python include lists and tuples
- A sequence type is formed by putting together some other types in a sequence.
- here is how we from list & tuples:

example of list:

```
>>> [1, 3, 4, 1, 6]
    [1, 3, 4, 1, 6]
  type ([1, 3, 4, 1, 6])
    <type 'list'>
```

example of tuples:

```
>>> (1, 3, 2)
 (1, 3, 2)
>>> type ((1, 3, 2))
 <type 'tuple'>
```

- Notice that list are enclosed in square bracket while tuples are enclosed in parenthesis
- list and tuples do not need to be homogen-ous that is the component can be of diff types

ex.

```
>>> [2, 2, "Hello", (1, 2)]
 [1, 2, "Hello", (1, 2)]
```

- Here we created a list containing for components can be diff types.

eg.

```
>>>[1, 2, "Hello" (1, 2)]
  [1, 2, "Hello", [1, 2)]
```

- Here we created a list containing for component two integers, a string, and a tuples, component of list may be other lists and so on

- sequence types such as a list, tuples and a strings are always ordered, as opposed to a set in mathematics which is always an ordered repeatation is allowed.

In a sequence, but not in sequence

* The range function:-

The range function is often used to create list

of integers. It has three forms range(n) produce a list of all number 0, 1, 2 ··· n-1 starting with a and ending with n-1 for instance

>>> ranger (7)

It has three forms,

[0, 1, 2, 3 ··· 17]

you can also specify an optional, starting point and an increment, which may be negative.

eg.

>>> range (1,10)

[1, 2, 3, 4, 5, 6, 7, 8, 9]

>>> range (1,10,2)

[1, 3, 5, 7, 9]

>>> range (10,0,2)

[10, 8, 6, 4, 2]

- the use of a negative increment in the last ex

* Boolean values :-

This is a value which is either true or false

eg >>> True

True

>>> Type (True)

<Type 'bool'>

>>> False

false

>>> type (false) <type 'bool'>

* Expressions :-

- Python expressions are not commands but rather

Form part of command.

- An expression is anything which produces a value e.g.

```
2+2
2**100
f(x-1)|x+1))
```

- Expressions are formed from variables, constants Function evaluation and operators parenthesis are used to indicate order of operations and grouping as used.

- The common binary operator for a arithmatic are
  + for addition
  - For substraction
  * for multiplication
  / for division As already mentioned, python uses
  ** for expontiation

ex

```
>>> 2**3
8
>>> 512
2
```

Float type

```
>>> 25.0**3
8.333333333333
>>> 512.0
25
```

- if just one of the operands is of type float, then the result will be of type float here is another ex of this pitfall.

>>> 2**(1/2)

Where we wanted to compute the square root of 2 as the exponent produce a list of result of 0 becomes of integer division is:
1 power of 2, but the division in the A corred way to this computation.

>>> 2**0.5
1.4142135623730951

- Another useful operator is %. which is read as "mod"
- it gives the reminder of an integer division

>>> 5 % 2
1
>>> 25 % 3
= 1

- which shows that 5 mod 2 = 1 & 25 mod 3 = 1 this operator is useful in number theory and cryphotography.

* Variable & Assignment:-
- An assignment statement in python has the form variable = expression
- First the expression on the right hand side is equivalent then the result is assigned to the variable. after the assignment the variable become a name for the result.
- executing the assignment produce no output it's purpose it to make the association betn the variable & its value.

```
>>> x = 2 + 2
>>> print x
  4
```

- in the example, the assignment statement sets x to 4, producing no output. If we want to see the value of x, we must print it if we execute another assignment to x then the precious value is lost.

eg

```
>>> x = 380.5
  Print x
  380.5
  Print x
  380.5
>>> y = 2 * x
  Printy y
  761.0
```

a single is used for assignment and the double == is used to test for equality

- in computer science the statement x = x+1 is useful it's purpose to add 1 to x

e.g

```
>>> x = 10
>>> x = x + 1
>>> print x
  11
```

- Variable names may be any contiguos sequence of letters, numbers, and the (_ underscore) character the first character must not be a number and you may not used a reserved keyword as a

Variable name.

eg.

a, v1, v1-1, abc, Bucket, monthly_total etc

Decisions :-

The if- else is used to make choice in python code this is compound statement

Syntax :-

```
if condition:
    action_1
else
    action_2
```

- The indentation is required else and its actions are optional.
- The actions action_1 and action_2 is executed In either case execution continues with the statement after the if- else.

ex.

```
x=1
if x>0:
    Print " Friday is Wonderful"
else:
    Print: "Monday sucks"
Print " Have a good Weekend"
```

results in the output -

Friday is Wonderful

Have a Good Weekend

- If We change the first time to say x = 0 then the output would be

Monday sucks

Have a good Weekend.

```
if x>=0 and x<10:
    digits=1
elif x>=10 and x<100:
    digit 2
elif x>=100 and x<1000:
    digit=3
elif x>=1000 and x<1000
elif x>=1000 digit 4
else:
    digit=0 # more than 4.
```

* Loops :-

- Python provides two looping commands
  1) For
  2) While
- these are compound command

① for loop -

Syntax :-

```
For item in list:
    action
```

- the action consists of one or more statement all at some indentation level.
- the statements are known as the body of the loop
- The item is a variable name & list is a list.

ex

```
For i in [2,4,6,0]
print
    output - 2460
```

2) While loop:-

Syntax:-

While condition:
    action

- The action may consists of one or more statement all of the same indentation level.
- The statement in the actions are called the body of the loop.

Execution of loop:-

- First the condition is evaluated if true then the body doesn't change the subsequent evaluations of the conditions an infinite loop may occur

for example,

```
While true:
    print "Hello"
```

- It will print Hellos endlessly
- to interrupt the execution of an infinite loop use ctrl + c

* else in loops:-

- A loop may have an optional else which is executed When the loop finishes

eg.

```
for n[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
    print n
else:
    print "blast off!!"
```

O/p:

10 9 8 7 6 5 4 3 2 1 Blast off

* Break Continue and pass :-

- The break statement like in c breaks out of the smallest enclosing for or while loop.
- The continue statement also borrowed from c continues with the next iteration of the loop.
- here is an example of the use of break statement and an else clause in a loop.

```
For n in range (2, 10):
    For x in range (2, n):
    If n % x == 0
    print n 'equals' x, 'x', n/x
    break
else:
    # loop fell through without finding a factor.
    Print n is a prime number.
```

* Lists :-

- A list is a finite sequence of items and one could use the range function to create list of integer in python, lists are not required to be homogenous ie. the items could be of diff type.

ex.

```
a=[2, "Jack", 45, "23 Went North Ave"]
```

- it's a perfectly valid list consisting of two integers & two strings.

- one can refer to the entire list using the identifier a or the i# item in list using a[i]

eg

```
>>> a = [2, "Jack", 45, "23 West Worth Ave"]
>>> a
[2, "Jack", 45, "23 West Worth Ave"]
>>> a[0]
2
>>> a[1]
"Jack"
>>> a[2]
45
>>> a[3]
"23 West Worth Ave"
```

- numbering of list items in Python always begins of 0 so the 4 items in the above list are indexed by the numbers 0, 1, 2, 3.

* List item may be assigned a new value :-

e.g

```
>>> a
[2, "Jack", 45, "23 West Worth Ave"]
```

* Length of a list :-
  - every list has a length, the number of items in the list.
  - it's obtain using len function.

e.g

```
>>> x = [9 4, 900, -45]
>>> len(x)
4
```

- the empty list of length of 0
>>> x = [ ]
>>> len(x)

* Sublists (slicing)
- Sublist are obtained by slicing which work analogously to the range function discussed before
- If x is an existing list then x [start : end] is the sublist consisting of items in the original list at index positions i such that
start ≤ i < end
- We must remember that indexing items always starts 0 in python
For eg

x = range (0, 20, 2)
>>> x [2:5]
[4: 6: 8]
>>> x[0:5]
[0, 2, 4, 6, 8]

* Joining two lists: -
Two existing list maybe concentrated together to make a longer list using the '+' operator.

>>> [2, 3, 5, 10] + [4, 0, 0, 5, 0]
[2, 3, 5, 10, 4, 0, 0, 5, 0]

List method : append:-
If x is the name of an existing list, we can append an item to the end of the list using
x . append (item)

To delete the item at index position I use

```
x.pop[1] >>> x.pop(1)
>>> x
['c', 'Junk', '13', 'd', '7']
```

- by default x.pop() pops the last item

```
>>> x.pop()
'7'
>>> x
['c', 'Junk', 3', 'd']
```

* String :-

A string in python is a sequence of characters. strings are similar to lists, python strings are immutable. meaning that we are not allowed to change individual part of them as we could for a list.

Conclusion: -

Thus we studied the basic syntax, data type, variables, operators, vectors, lists, taking suitable data from keyboard and count no. of word from file.