Printing to the screen : The simplest way to produce output is using the print statement where you can pass zero or more expressions separated by commas, This function converts the expressions we pass into a string and writes the result to standard output.

print("Python is a interesting language")

Reading keyboard Input: The input() functions read data from keyboard as string, irrespective of whether it is enclosed with quotes (“ or””) or not

The input function: The input[prompt] function is equivalent to raw\_input, except that it assumes that the input is a valid Python expression and returns the evaluated result to you.

The open function: Before you read or write a file, you have to open it using Python’s build-in open() function. This function creates a file object, which would be utilized to call other support methods associated with it.

Syntax: file object = open(file\_name[, acess\_mode] [, buffering])

Here the file\_name : The file\_name argument is a string value that contains the name of the file that you want to access.

Access\_mode: The access\_mode determines the mode in which the file has to be opened ie read, write, append etc. This is an optional parameter and the default file access mode is read(r)

Buffering: If the buffering value is set to 0, no buffering take place. If the buffering value is 1, line buffering is performed while accessing a file. If you specify the buffering value as a integer greater than 1, then buffering action is performed with the indicated buffer size. If negative, the buffer size is the system default(default behaviour).

The close() method: The close() method of a file object flushes any written information and closes the file object, after which no more writing can be done.

Syntax : fileobject.close();

Reading and Writing Files:

The file object provides a set of access methods to make our lives easier. We would see how to use read() and write() methods to read and write files.

The write() method writes any string to an open file. It is important to note that Python strings can have binary data and not just text.

Note: The write() method does not add a newline character(‘\n’) to the end of the string

Syntax : filobject.write(string);

Example : fo = open("foo.txt","w")

fo.write("Python is a great language.\n Yeah its great!!!\n")

fo.close()

The read() Method: The read() method reads a string from an open file. It is important to note that Python strings can have binary data apart from the text data.

Syntax : fileObject.read([count]);

Example: fo= open ("foo.txt", "rt")

str = fo.read(10)

print("Read String is :", str)

fo.close()

File Positions: The tell() method tells you the current position within the file; in other words, the next read or write will occur at that many bytes from the beginning of the file.

The seek(offset[, from]) method changes the current file position. The offset argument indicates the number of bytes to be moved. The from argument specifies the reference position from where the bytes are to be moved.

If form is set to 0, the beginning of the file is used as the reference position. If it is set to 1, the current position is used as the reference position. If it is set to 2 then the end of the file would be taken as the reference position.

Rename and Deleting Files:

The rename() Method: The rename() method takes two arguments, the current filename and the new filename.

Syntax: os.rename(current\_file\_name, new\_file\_name)

Example: Following is an example to rename an existing file test1.txt

import os

os.rename(“text1.txt”,”text2.txt”)

The remove() Method:

You can use the remove() method to delete files by supplying the name of the file to be deleted as the argument.

Syntax: os.remove(file\_name)

Example: Following is an example to delete an existing file test2.txt

import os

os .remove(“text2.txt”)

The mkdir() Method: You can see the mkdir() method of the os module to create directories in the current directory. You need to supply an argument to this method which contains the name of the directory to be created.

Syntax : os.mkdir(“newdir”)

Example: Following is an example to create a directory test in the current directory

import os

os.mkdir(“test”)

The chdir() Method :

You can use the chdir() method to change the current directory . The chdir() method takes an argument , which is the name of the directory that you want to make the current directory.

Synatax: os.chdir(“newdir”)

Example: import os

os.chdir(“/home/newdir”)

The getcwd() Method:

The getcwd() method displays the current working directory.

Syntax: os.getcwd()

Example: Following is a example to give the current directory.

Import os

os.getcwd()

The rdmir() Method: The rdmir() method deletes the directory, which as an argument in the method. Before removing a directory all the contents in it should be removed.

os.rmdir(‘dirname’)

Example: Following is an example to remove the “/tmp/test” directory. It is required to give fully qualified name of the directory otherwise it would search for that directory in the current directory

import os

os.rmdir(“/tp/test”)