CH10: Python I/O

Input /output are the basic or fundamental operation for any language. Python support a better way to do input / output. There are basically two kind of I/0 operations supported by python.

* I/O to the console.
* File I/O

**10.1 Inputting and outputting data to/from console.**

The simplest way to produce output to the screen is using print statement. “Print” converts an expression into string data and writes it to standard output. You can pass zero or more argument, separated by comma. Python insert spaces in between. For example:

>>> print 'Sum of ',10,20,'is: ',10+20

Sum of 10 20 is: 30

>>>

If there are two print statements, they got printed in separate line. Like:

>>> print 'hello';print 'world'

hello

world

>>>

If you want to print in same line, you need to give COMMA at the end of print statement like:

>>> print 'hello',;print 'World',;print'HelloWorld',

hello World HelloWorld

>>>

On the other side, python offer two build-in functions to retrieve line from standard input, which basically come from user.

Raw\_input([prompt]) reads one line from standard input, and return it as a string for example.

The below example show how I can read a line from user

>>> x = raw\_input('Enter a line:')

Enter a line: I love Djnago

>>> print x

I love Djnago

We can also read a list of integer from user. Still, they are treated as single string, we can convert it into a list of intergers.

>>> x = raw\_input('Enter a list of number:')

Enter a list of number:10 20 30 40 50

>>> print [ int(i) for i in x.split()]

[10, 20, 30, 40, 50]

The input([prompt] is similar as raw\_input(), but it treat the input as a valid python expression. Like:

>>> x = input('Enter a string:')

Enter a string:'dipankar'

>>> x

'dipankar'

>>> x = input('Enter a string without quote:')

Enter a string without quote:Dipankar # This is not a valid expression.

Traceback (most recent call last):

File "<pyshell#17>", line 1, in <module>

x = input('Enter a string without quote:')

File "<string>", line 1, in <module>

NameError: name 'dipankar' is not defined

>>> x = input('Enter a python list expression')

Enter a python list expression [x\*10 for x in range(2,20,2)]

>>> x

[20, 40, 60, 80, 100, 120, 140, 160, 180]

>>>

**10.2 File Programming Approach**

In Python programming approach is very simple. It’s very easy to create file, write something and read a file.

A file can be opened by standard open() method .

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

* file\_name: The file\_name argument is a string value that contains the name of the file that you want to access. The file should be in same directory as in python script.
* access\_mode: The access\_mode determines the mode in which the file has to be opened, i.e., read, write, append.
* buffering: If the buffering value is set to 0, no buffering will take place. If the buffering value is 1, line buffering will be performed while accessing a file. If you specify the buffering value as an integer greater than 1, then buffering action will be performed with the indicated buffer size.

Below List show the different file access mode:

|  |  |
| --- | --- |
| Modes | Description |
| r | Opens a file for reading only. The file pointer is placed at the beginning of the file. This is the default mode. |
| rb | Opens a file for reading only in binary format. |
| r+ | Opens a file for both reading and writing. |
| rb+ | Opens a file for both reading and writing in binary format. |
| w | Opens a file for writing only. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing. |
| wb | Opens a file for writing only in binary format. |
| w+ | Opens a file for both writing and reading. |
| wb+ | Opens a file for both writing and reading in binary format. |
| a | Opens a file for appending. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing. |
| ab | Opens a file for appending in binary format. |
| a+ | Opens a file for both appending and reading. |
| ab+ | Opens a file for both appending and reading in binary format. |

**10.2.1 Open a File and read attribute**

Let’s Open a file and read the file properties:

try:

f = open('hello.txt','wb')

print 'Name:',f.name

print 'closed or not:',f.closed

print 'Opening Mode: ',f.mode

f.close()

except Exception,e:

print 'Error:',e

Output :

Name: hello.txt

closed or not: False

Opening Mode: wb

In summary, We can say:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| **F = Open()** | **Open a file for read or write perpose** |
| f.closed | Returns true if file is closed, false otherwise. |
| f.mode | Returns access mode with which file was opened. |
| f.name | Returns name of the file. |
| f.softspace | Returns false if space explicitly required with print, true otherwise. |
| f.close() | The close() method of a *file* object flushes any unwritten information and closes the file object, after which no more writing can be done. |

**10.2.2 Write a File**

Let’s Start writing a file: f.write() method is used to write data into the file. For example,.

try:

f = open('hello.txt','wb')

f.write('Hello This is Django\n');

f.write('Ok. This is second line\n');

f.close()

print 'Write Successfully'

except Exception,e:

print 'Error:',e

This program writes only two lines in the file.

**10.2.1 Read record from a File**

Let’s we have a file having the record of student data. Each record contains <roll\_no, name, and department >, separated by space.

10 dipankar CSE

11 Subha CSE

12 Sougata CSE

15 Ram CSE

We like to read the data and find out the name of each department and total number of student in each branch.

try:

f = open('hello.txt','r')

data = f.read()

print 'All Data: ', data # this is just a raw data.

records = [ tuple(raw.split(' ')) for raw in data.split('\n')]

print 'All records: ', records # this is a list of tuple

f.close()

except Exception,e:

print 'Error:',e

Output :

All records: [('10', 'dipankar', 'CSE'), ('11', 'Subha', 'CSE'), ('12', 'Sougata', 'CSE'), ('15', 'Ram', 'CSE')]

Let write a program which contains a story. We like to find out number of lines, number of word and number of character in that file. The following program does that for you.

try:

f = open('hello.txt','r')

data = f.read()

print 'No of Line: ',len(data.split('\n'))

print 'No of Words: ',len(data.split('\S'))

print 'No of Chracter: ',len(data)

f.close()

except Exception,e:

print 'Error:',e

Output :

No of Line: 4

No of Words: 1

No of Chracter: 54

**10.2.1 Read record from a file using readline() and readlines()**

Instead using read the full data, we can use readline() or readlines() to read the file line by line. Following example print the data line by line.

try:

f = open('hello.txt','r')

for index,line in enumerate(f.readlines()):

print 'Line No#',index,':',line

f.close()

except Exception,e:

print 'Error:',e

**output:**

Line No# 0 : 10 dipankar CSE

Line No# 1 : 11 Subha CSE

Line No# 2 : 12 Sougata CSE

Line No# 3 : 15 Ram CSE