```
[ etc. ]
```

You can also program your script in such a way that it should accept various options.

Accessing Command-Line Arguments

Python provides a **getopt** module that helps you parse command-line options and arguments.

```
$ python test.py arg1 arg2 arg3
```

The Python **sys** module provides access to any command-line arguments via the **sys.argv**. This serves two purposes:

- sys.argv is the list of command-line arguments.
- len(sys.argv) is the number of command-line arguments.

Here sys.argv[0] is the program i.e. script name.

Example

Consider the following script test.py:

```
#!/usr/bin/python


import sys


print 'Number of arguments:', len(sys.argv), 'arguments.'

print 'Argument List:', str(sys.argv)
```

Now run above script as follows:

```
$ python test.py arg1 arg2 arg3
```

This produces the following result:

```
Number of arguments: 4 arguments.

Argument List: ['test.py', 'arg1', 'arg2', 'arg3']
```