Programming and Problem Solving through Python Language
O Level / A Level

Chapter - 7: File Processing

Command Line Arguments

getopt Module

- This module implements the parsing of options and arguments from the list structure of command line arguments named **sys.argv**.
- To use this module, it is required to import the **getopt Module. e.g. import getopt.**
- This module provides functions and exception to parse the command line argument.
    - .**getopt**( ) - This method returns (option, value) pair and option list left unused.
    - .**GetoptError** Exception – This raise an error when unrecognized option found or argument for the required option not provided.

Syntax

getopt.getopt(args, options , [long_options])

args	- List of arguments**.**
options	- String of option letters to be recognized , and **colon**(:) follows the letter which requires an argument. This option letter has prefix hyphen (-). e.g. –i , -o
long_options	- It is Optional. It is a list of strings which represent the long names equivalent for the option character. **Equal Sign**(=) follows the long name which requires an argument. This long option name has prefix(--). e.g. –inFile , --outFile

Example

If we have defined for parsing the argument

```
opts,args=getopt.getopt( argv, "hi:o:", ["help","inFile=","outFile="])
```

It means

-i or --inFile	can be used to specify input files. It requires an argument.
-o or --outFile	can be used to specify output files. It requires an argument.
-h or –help	can be used for help. It does not require an argument.

```
showarg.py  -i  abc -o xyz
showarg.py  --inFile  abc -o xyz
showarg.py  --inFile  abc --outFile xyz
showarg.py  --inFile  abc --outFile        #It shows error, as output argument not specified.
showarg.py  --inFile  abc –z xyz           #It shows error, -z is an unknown option.
```

Program

```
# show_arg.py script file to handle the command line argument
import sys
import getopt

print("List of arguments received ")
print(sys.argv)

len_argv=len(sys.argv)
print("Count of Argument - ", len_argv)

print("List of argument with Loop ")
for arg in sys.argv:
   print( arg)

print("Output from getopt")
argv=sys.argv[1:]

try:
    opts , args=getopt.getopt(argv,"hi:o:",["help","inFile=","outFile="])
    print(opts)
    print(args)
except getopt.GetoptError :
   print("some error")
```

Output

First Set of Output

```
C:\Python38> python show_arg.py  -i abc  -o xyz

List of arguments received
[ 'show_arg.py',  '-i',  'abc', '-o', 'xyz' ]

Count of Argument -  5

List of argument with Loop
show_arg.py
-i
abc
-o
xyz

Output from getopt
[('-i', 'abc'), ('-o', 'xyz') ]
[ ]
```

Second Set of Output

```
C:\Python38> python show_arg.py  --inFile abc  --outFile xyz

List of arguments received
[ 'show_arg.py',  '--inFile',  'abc', '--outFile', 'xyz' ]
```

```
Count of Argument -  5

List of argument with Loop
show_arg.py
--inFile
abc
--outFile
xyz

Output from getopt
[ ( '--inFile', 'abc' ), ( '--outFile', 'xyz') ]
[ ]
```

Third Set of Output

```
C:\Python38> python show_arg.py  --inFile abc  -z xyz

List of arguments received
[ 'show_arg.py',  '--inFile',  'abc', '--outFile', 'xyz' ]

Count of Argument -  5

List of argument with Loop
show_arg.py
--inFile
abc
--outFile
xyz

Output from getopt
Some error
```

Fourth Set of Output

```
C:\Python38> python show_arg.py  -i abc  pqr -o xyz

List of arguments received
[ 'show_arg.py',  '-i',  'abc',  'pqr', '-o', 'xyz' ]

Count of Argument -  6

List of argument with Loop
show_arg.py
-i
abc
pqr
-o
xyz

Output from getopt
[('-i', 'abc') ]
[ ('pqr'), ('-o', 'xyz')]              #unutilized parameters
```

Program To show the message depending on the input option provided.

```
import sys
import getopt

#to remove the script name from arguments list
argv=sys.argv[1:]

try :
    opts,args=getopt.getopt( argv, "hi:o:", ["help","inFile=","outFile="] )
except getopt.GetoptError :
    print("show_arg.py -i filename -o filename")
    sys.exit( )

for opt, arg in opts:
     if opt == '-h':
        print ("show_arg.py -i filename -o filename")
        sys.exit()
     elif opt in ("-i", "--inFile"):
        inputfile = arg
     elif opt in ("-o", "--outFile"):
        outputfile = arg

print ('Input file is ', inputfile)
print ('Output file is ', outputfile)
```

Output

```
D:\Python38>python show_arg.py -i  abc  --outFile xyz
Input file is  abc
Output file is  xyz


D:\Python38>python show_arg.py -i  abc  --ouftile xyz
show_arg.py -i filename -o filename


D:\Python38>python show_arg.py -h
show_arg.py -i filename -o filename


D:\Python38>python show_arg.py -help
show_arg.py -i filename -o filename
```