II. COMMAND LINE ARGUMENTS

- It is possible to pass some values from the command line to python programs when they are executed. These values are called command line arguments and it can be used to control program from outside instead of hard coding.
- The command line arguments are handled using sys module. We can access command-line arguments via the sys.argv This serves two purposes –

Example 1

Consider the following script command.py

```
import sys
program_name = sys.argv[0]
arguments = sys.argv[1:]
count = len(arguments)
print(program_name)
print(arguments)
print(count)C
```

```
>>>C:\Python30>python.exe     command.py
                              Hello World
                              11
```

- 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 name.

III. ERRORS AND EXCEPTIONS

ERRORS

Errors or mistakes in a program are often referred to as bugs. The process of finding and eliminating the errors is called debugging. Errors can be classified into three major groups:

- Syntax Error
- Runtime Error
- Logical Error

i) Syntax Error

- Python will find these kinds of errors when it tries to parse your program, and exit with an error message without running anything.

Syntax errors are mistakes in the use of the Python Language, and are analogous to spelling or grammar mistakes in a language like English

Common Python Syntax Errors include:

- Leaving out a Keyword
- Putting a keyword in a wrong place
- Misspelling a Keyword
- Incorrect Indent
- Unnecessary Spaces
- Empty Spaces

ii) Runtime Error

If a program is syntactically correct that is free from syntax errors, however the program exits unexpectedly it is due to runtime error. When a program comes to halt because of runtime error, then it has crashed.

Common Python Runtime Errors include:

- Division by Zero
- Performing an operation on different data type
- Using an identifier which has not been defined
- Accessing a list element which doesn't exist
- Trying to access a file which doesn't exist

Example
a=10
b=0
c=a/b

iii) Logical Error

- Logical Errors are the most difficult one to fix. They occur when the program runs without crashing but it produces an incorrect result. The error is occurred by a mistake in program logic.

Common Python Logical Errors include:

- Using the wrong variable name
- Indenting a block to the wrong level
- Using Integer division instead of float division
- Getting Operator Precedence wrong (Priority)