What is a command line argument?

A command line argument is nothing but an argument sent to a program being called. A program can take any number of command line arguments.

What is argv[0]?

Remember that sys.argv[0] is the name of the script.

Here – Script name is **sysargv.py**

```
import sys

print ("This is the name of the script: ", sys.argv[0])

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

print ("The arguments are: " , str(sys.argv))
```

Output:

```
This is the name of the script:  sysargv.py
Number of arguments:  1
The arguments are:  ['sysargv.py']
```

Why do we use them?

Command line arguments are parameters passed to a program/script at runtime. They provide additional information to the program so that it can execute.

It allows us to provide different inputs at the runtime without changing the code.

Here is a script named as argparse_ex.py:

```
import argparse
```

```
parser = argparse.ArgumentParser()

parser.add_argument("-n", "--name", required=True)

args = parser.parse_args()

print(f'Hi {args.name} , Welcome ')
```

Here we need to import argparse package

Then we need to instantiate the ArgumentParser object as parser. Then in the next line , we add the only argument, --name . We must specify either shorthand (-n) or longhand versions (--name) where either flag could be used in the command line as shown above . This is a required argument as mentioned by required=True

Output:

```
(venv) C:\Users\Nandank\PycharmProjects\DSA\venv>python a
rgparse_ex.py --name Nandan

Hi Nandan , Welcome

(venv) C:\Users\Nandank\PycharmProjects\DSA\venv>python a
rgparse_ex.py -n Nandan

Hi Nandan , Welcome
```

The example above must have the --name or –n option, or else it will fail.

```
(venv) C:\Users\Nandank\PycharmProjects\DSA\venv>python a
rgparse_ex.py --name

usage: argparse_ex.py [-h] --name NAME

argparse_ex.py: error: the following arguments are requir
ed: --name
```