The whole argument parser(argparse) programming include two part:

1. build argparser
2. parse the argument

[Library]

Import argparse

[Build argparser]

1. declare a parser

>> parser = argparse.ArgumentParser()

1. define the how the parser parse the argument

>> parser.add\_argument(...)

There are two types of argument:

* positional argument

this kind of argument is mandatory, and is parsed according to its place

>> parser.add\_argument(<name>,

help=<description>,

type=<type>)

<name>: [str] the name of the argument

<description>: [str] the description of the argument

<type>: [type] the input type of the argument

Example:

argparse.add\_argument(‘num1’, type=int)

argparse.add\_argument(‘num2’, type=int)

>> argument: ’10 20’ => args.num1 is 10, args.num2 is 20

* optional argument

this kind of argument is optional, and is parsed according to its argument prefix name

>> parser.add\_argument( <name>,

help=<description>,

type=<type> |action=<action>)

<name>: [str] the name of the argument, must have prefix of - or --

(the predix of - means the name is a shortcut name, -- means the full name. Usually have both )

<description>: [str] the description

<type>: [type] the input type

<action>: [string] the action type (‘store\_true’: if this argument exist then it is true, else false)

Example:

parser.add\_argument(‘-o’, ‘--output’, action= ‘store\_true’)

>> argument: ‘20 -o 1 10’ => args.output is True

>> argument: ‘20 --output 1 10’ => args.output is True

parser.add\_argument(‘-o’, ‘--output’, type= ‘int’)

>> argument: ‘20 -o 1 10’ => args.output is 1

>> argument: ‘20 --output 1 10’ => args.output is 1

[Parse the argument]

>> arg = parser.parse\_args() # input the argument script to the argument parser and output a namespace

# arg.<name> will be the argument value of <argument\_name>