1. Take **a** date as a string in "YYYY/MM/DD" format and a number of day before or after

(+/- n) the given date as the command line arguments

1. def valid\_date(today): if len(today) != 10: sys.exit('Error: wrong date entered')

The length of the date is not exactly 10 it will generate an error message: ‘Error: wrong day entered’ that is in this format “yyyy/mm/dd”. = 10

1. def usage():""

This function will input no argument and will output with a string describing the usage.

1. str\_year, str\_month, str\_day = today.split('/')

Here, the string is converted to a date string format that is (yyyy/mm/dd)(separate by “/”).

1. def valid\_date(today):

This function need a date in the formation of "YYYY/MM/DD", and will output True, if the input date is a valid one, else, it will return False with an appropriate message output

1. if month > 12 or month <1: sys.exit('Error: wrong month entered')

Here, it says that, if the month is less than 0 and greater than 12, the script will generate an error status: ‘Error: wrong month entered’

1. mon\_max = days\_in\_mon(year) to\_day = mon\_max[month] if 1 > day or day > to\_day:

sys.exit('Error: wrong day entered')

Here, it says, the days cannot be less than or it cannot be more that maximum days in a year. Otherwise, it will return an error message: 'Error: wrong day entered’

1. def leap\_year(year):

This function will ask for a year like "YYYY" structure, which will return True if the input year is a leap year, else return with a False statement

1. lyear = year % 4 if lyear == 0:

Here it says, year divisible by 4 is a leap year and it starts from 0.

1. def days\_in\_mon(year):

This function will input a year like "YYYY", and return back with a dictionary object which contains the total number of days in a month for the instructed year.

1. leapyearornot = leap\_year(year)

    if leapyearornot is False:

       feb\_max = 28

    else:

       feb\_max = 29

Here, it checking that the year is leap year or not? If February has a maximum days it is not a leap year else February has maximum days 29, it is a leap year.

1. mon\_max = { 1:31, 2:feb\_max, 3:31, 4:30, 5:31, 6:30, 7:31, 8:31, 9:30, 10:31, 11:30, 12:31}

Create a dictionary object for the number of days in each month

1. def dbda(today,days,step):

This function has to be the main function of your script. This function will intake a date in "YYYY/MM/DD" condition, it may be a positive or a negative integer, and return with a date, which can be before or after the given date, according to the number provided in the same date format.

1. next\_date = str(year)+"/"+str(month).zfill(2)+"/"+str(to\_day).zfill(2)

zfill: takes a single character ”width”. Its function is to return with a copy of string with ‘0’ filled to the left.

1. if \_\_name\_\_ == '\_\_main\_\_':

Python interpreter does 2 thing when it reads a source file

* Sets a few special variables like \_name\_, and then executes all codes found in the file.