**Assignment 5**

1. Write a Python script to display the various Date Time formats -  
   a) Current date and time  
   b) Current year  
   c) Month of year  
   d) Week number of the year  
   e) Weekday of the week  
   f) Day of year  
   g) Day of the month  
   h) Day of week
2. Write a Python program to determine whether a given year is a leap year.

**3.** Write a Python program to convert a string to datetime.  
Sample String : Jul 1 2014 2:43PM  
Expected Output : 2014-07-01 14:43:00

**4.** Write a Python program to get the current time in Python.  
Sample Format :  13:19:49.078205

**5.** Write a Python program to subtract five days from the current date.  
Sample Date :  
Current Date : 2015-06-22  
5 days before Current Date : 2015-06-17

Write a Python program to convert a Unix timestamp string to a readable date.  
Sample Unix timestamp string : 1284105682  
Expected Output : 2010-09-10 13:31:22

**7.** Write a Python program to print yesterday, today, tomorrow.

**8.** Write a Python program to convert the date to datetime (midnight of the date) in Python.  
Sample Output : 2015-06-22 00:00:00

**9.** Write a Python program to print the next 5 days starting today.

**10.** Write a Python program to add 5 seconds to the current time.  
Sample Data :  
13:28:32.953088  
13:28:37.953088

**11.** Write a Python program to convert Year/Month/Day to Day of Year in Python.

**12.** Write a Python program to get the current time in milliseconds in Python.

**13.** Write a Python program to get the week number.  
Sample Date : 2015, 6, 16  
Expected Output : 25

**14.** Write a Python program to find the date of the first Monday of a given week.  
Sample Year and week : 2015, 50  
Expected Output : Mon Dec 14 00:00:00 2015

**15.** Write a Python program to select all the Sundays in a specified year.

**16.** Write a Python program to add year(s) to a given date and display the updated date.  
  
Sample Data : (addYears is the user defined function name)  
print(addYears(datetime.date(2015,1,1), -1))  
print(addYears(datetime.date(2015,1,1), 0))  
print(addYears(datetime.date(2015,1,1), 2))  
print(addYears(datetime.date(2000,2,29),1))  
  
Expected Output :  
2014-01-01  
2015-01-01  
2017-01-01  
2001-03-01

**17.** Write a Python program to drop microseconds from datetime.

**18.** Write a Python program to get days between two dates.  
Sample Dates : 2000,2,28, 2001,2,28  
Expected Output : 366 days, 0:00:00

**19.** Write a Python program to get the date of the last Tuesday.

**20.** Write a [Python](https://www.w3resource.com/python-exercises/date-time-exercise/index.php) program to test the third Tuesday of a month.

**21.** Write a Python program to get the last day of a specified year and month.

**22.** Write a Python program to get the number of days in a given month and year.

**23.** Write a Python program to add a month to a specified date.

**24.** Write a Python program to count the number of Mondays on the 1st day of the month from 2015 to 2016.

**25.** Write a Python program to print a string five times, with a delay of three seconds..

**26.** Write a Python program that calculates the date six months from the current date using the datetime module.

**27.** Write a Python program to create 12 fixed dates from a specified date over a given period. The difference between two dates is 20.

**28.** Write a Python program to get the dates 30 days before and after today.

**29.** Write a Python program to get GMT and the local time.

**30.** Write a Python program to convert a date to a timestamp.

**31.** Write a Python program to convert a string date to a timestamp.

**32.** Write a Python program to calculate the number of days between two dates.

**33.** Write a Python program to calculate the number of days between two date times.

**34.** Write a Python program to display the date and time in a human-friendly string.

**35.** Write a Python program to convert a date to a Unix timestamp.

**36.** Write a Python program to calculate the difference between two dates in seconds.

**37.** Write a Python program to convert difference of two dates into days, hours, minutes, and seconds.

**38.** Write a Python program to get the last modified information of a file.

**39.** Write a Python program to calculate an age in years.

**40.** Write a Python program to get the current date and time information.

**41.** Write a Python program to generate a date and time as a string.

**42.** Write a Python program to display formatted text output of a month and start the week on Sunday.

**43.** Write a Python program to print a 3-column [calendar](https://www.w3resource.com/python-exercises/date-time-exercise/index.php) for an entire year.

**44.** Write a Python program to display a calendar for a locale.

**45.** Write a Python program to get the current week.

**46.** Write a Python program to create a HTML calendar with data for a specific year and month.

**47.** Write a Python program display a list of the dates for the 2nd Saturday of every month for a given year.

**48.** Write a Python program to display a simple, formatted calendar of a given year and month.

**49.** Write a Python program to convert a string into datetime

**50.** Write a Python program to get a list of dates between two dates.

**51.** Write a Python program to generate RFC 3339 timestamp.

**52.** Write a Python program to get the first and last second.

**53.** Write a Python program to validate a Gregorian date. The month is between 1 and 12 inclusive, the day is within the allowed number of days for the given month. Leap year's are taken into consideration. The year is between 1 and 32767 inclusive.

**54.** Write a [Python](https://www.w3resource.com/python-exercises/date-time-exercise/index.php) program to set the default timezone used by all date/time functions.

**55.** The epoch is the point where time starts, and is platform dependent. For Unix, the epoch is January 1, 1970, 00:00:00 (UTC). Write a Python program to find out what the epoch is on a given platform. Convert a given time in seconds since the epoch.  
Sample Output:  
Epoch on a given platform:  
time.struct\_time(tm\_year=1970, tm\_mon=1, tm\_mday=1, tm\_hour=0, tm\_min=0, tm\_sec=0, tm\_wday=3, tm\_yday=1, tm\_isdst=0)  
Time in seconds since the epoch:  
time.struct\_time(tm\_year=1970, tm\_mon=1, tm\_mday=1, tm\_hour=10, tm\_min=0, tm\_sec=0, tm\_wday=3, tm\_yday=1, tm\_isdst=0)

**56.** Write a Python program to get time values with components using local time and gmtime.  
Sample Output:  
localtime:  
tm\_year : 2021  
tm\_mon : 4  
tm\_mday : 13  
tm\_hour : 11  
tm\_min : 20  
tm\_sec : 37  
tm\_wday : 1  
tm\_yday : 103  
tm\_isdst: 0  
gmtime:  
tm\_year : 2021  
tm\_mon : 4  
tm\_mday : 13  
tm\_hour : 11  
tm\_min : 20  
tm\_sec : 37  
tm\_wday : 1  
tm\_yday : 103  
tm\_isdst: 0

**57.** Write a Python program to get different time values with components timezone, timezone abbreviations, the offset of the local (non-DST) timezone, DST timezone and time of different timezones.  
Sample Output:  
Default Zone:  
TZ : (not set)  
Timezone abbreviations: ('UTC', 'UTC')  
Timezone : 0 (0.0)  
DST timezone 0  
Time : 11:30:05 04/13/21 UTC  
Pacific/Auckland :  
TZ : Pacific/Auckland  
Timezone abbreviations: ('NZST', 'NZDT')  
Timezone : -43200 (-12.0)  
DST timezone 1  
Time : 23:30:05 04/13/21 NZST  
Europe/Berlin :  
TZ : Europe/Berlin  
Timezone abbreviations: ('CET', 'CEST')  
Timezone : -3600 (-1.0)  
DST timezone 1  
Time : 13:30:05 04/13/21 CEST  
America/Detroit :  
TZ : America/Detroit  
Timezone abbreviations: ('EST', 'EDT')  
Timezone : 18000 (5.0)  
DST timezone 1  
Time : 07:30:05 04/13/21 EDT  
Singapore :  
TZ : Singapore  
Timezone abbreviations: ('+08', '+08')  
Timezone : -28800 (-8.0)  
DST timezone 0  
Time : 19:30:05 04/13/21 +08

**58.** Write a Python program that can suspend execution of a given script for a given number of seconds.  
Sample Output:  
Sorry, Slept for 3 seconds...  
Sorry, Slept for 3 seconds...  
Sorry, Slept for 3 seconds...  
Sorry, Slept for 3 seconds...

**59.** Write a Python program to convert a given time in seconds since the epoch to a string representing local time.  
Sample Output:  
Tue Apr 13 11:51:51 2021  
Thu Jun 30 18:36:29 1977

**60.** Write a Python program that prints the time, names, representation format, and the preferred date time format in a simple format.  
Sample Output:  
Simple format of time:  
Tue, 13 Apr 2021 12:02:01 + 1010  
Full names and the representation:  
Tuesday, 04/13/21 April 2021 12:02:01 + 0000  
Preferred date time format:  
Tue Apr 13 12:02:01 2021  
Example 11: 04/13/21, 12:02:01, 21, 2021

**61.** Write a Python program that takes a given number of seconds and passes since the epoch as an argument. Print structure time in local time.  
Sample Output:  
Result: time.struct\_time(tm\_year=1983, tm\_mon=2, tm\_mday=19, tm\_hour=21, tm\_min=38, tm\_sec=18, tm\_wday=5, tm\_yday=50, tm\_isdst=0)  
Year: 1983

**62.** Write a Python program that takes a tuple containing 9 elements corresponding to structure time as an argument and returns a string representing it.  
Sample Output:  
Result: Sun Jan 22 02:34:06 2020  
Result: Tue Nov 12 02:54:08 1982

**63.** Write a [Python](https://www.w3resource.com/python-exercises/date-time-exercise/index.php) program to [parse](https://www.w3resource.com/python-exercises/date-time-exercise/index.php) a string representing time and return the time structure.  
Sample Output:  
String representing time: 22 January, 2020  
time.struct\_time(tm\_year=2020, tm\_mon=1, tm\_mday=22, tm\_hour=0, tm\_min=0, tm\_sec=0, tm\_wday=2, tm\_yday=22, tm\_isdst=-1)  
String representing time: 30 Nov 00  
time.struct\_time(tm\_year=2000, tm\_mon=11, tm\_mday=30, tm\_hour=0, tm\_min=0, tm\_sec=0, tm\_wday=3, tm\_yday=335, tm\_isdst=-1)  
String representing time: 04/11/15 11:55:23  
time.struct\_time(tm\_year=2015, tm\_mon=4, tm\_mday=11, tm\_hour=11, tm\_min=55, tm\_sec=23, tm\_wday=5, tm\_yday=101, tm\_isdst=-1)  
String representing time: 12-11-2019  
time.struct\_time(tm\_year=2019, tm\_mon=12, tm\_mday=11, tm\_hour=0, tm\_min=0, tm\_sec=0, tm\_wday=2, tm\_yday=345, tm\_isdst=-1)  
String representing time: 13::55::26  
time.struct\_time(tm\_year=1900, tm\_mon=1, tm\_mday=1, tm\_hour=13, tm\_min=55, tm\_sec=26, tm\_wday=0, tm\_yday=1, tm\_isdst=-1)