

Programming Assignment: Zeller's Algorithm

Problem Statement:

In this assignment, you will write a program that

- Asks user to enter the date as: Month, Day and Year. (*Note for this assignment, assume user enters integers for all the three inputs*)
- Validates that all the input values are within bounds as below. If not, prints "Invalid input" and exits. (*Hint: You can end a program by calling the `exit()` function.*)
 - o Year is between 1582 and 4902 (both included)
 - o Month is between 1 and 12 (both included)
 - o Day is valid for the given month and year accounting for leap year¹ adjustment.
- For valid input, uses Zeller's Algorithm given below to find out the day of the week for that date as a number between 0 and 6 where 0 indicates Sunday and 6 indicates Saturday.
- Prints the corresponding day of the week. E.g.

Day of the week: Wednesday.

- See sample runs below.

Save your program to a file with a name of the format `first_last_Zellers.py`. ***It is highly recommended that you follow the steps of the algorithm closely and in sequence as given below.***

Zeller's Algorithm:

Zeller's algorithm computes the day of the week on which a given date will fall (or fell).

Step 1: Adjust month and year: Convert the entered month such that March is 1, April is 2, ... , December is 10, January is 11 and February is 12. If month entered is Jan or Feb, decrement the year value. (This is because there was a period in history when March 1st, not January 1st, was the beginning of the year.) e.g. If entered date is 9/29/1989, the adjusted month will be 7 and adjusted year will be 1989. If entered date is 1/18/2015, the adjusted month will be 11 and adjusted year will be 2014.

¹ Definition of a leap year (https://en.wikipedia.org/wiki/Leap_year) : Every year that is exactly divisible by four is a leap year, except for years that are exactly divisible by 100, but these centurial years are leap years if they are exactly divisible by 400. For example, the years 1700, 1800, and 1900 are not leap years, but the years 1600 and 2000 are.

Step 2: Calculate intermediate values: Compute **a**, **b**, **c** and **d** have the following values:

a = adjusted value for the month of the year from step 1 (with March having the value 1, April the value 2, . . . , December the value 10, and January and February being counted as months 11 and 12)

b = the day of the month (1, 2, 3, . . . , 30, 31)

c = the year of the century for the adjusted year from Step 1 (e.g. **c** = 89 for the year 1989) (*Hint: Think modulo function %*)

d = the century of the adjusted year from Step 1 (e.g. **d** = 19 for the year 1989) (*Hint: you should be able to calculate this using adjusted year and c values*)

Print out intermediate values (**a**, **b**, **c**, **d**) e.g.

a b c d = 11 1 17 20

Step 3: Compute final values: Let **w**, **x**, **y**, **z**, **r** also denote integer variables. Compute their values in the following order using *integer arithmetic*:

w = $(13*a - 1) / 5$

x = $c / 4$

y = $d / 4$

z = $w + x + y + b + c - 2*d$

r = the remainder when **z** is divided by 7

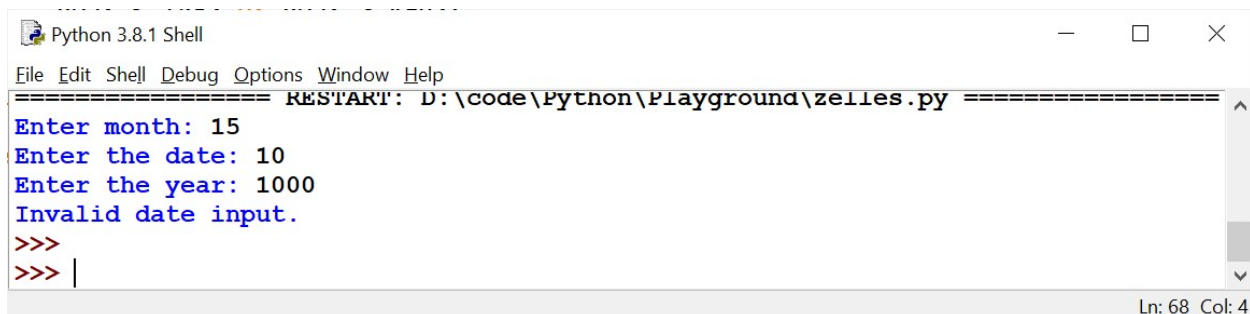
Print out final values (**w**, **x**, **y**, **z**, **r**) e.g.

w x y z r = 28 4 5 15 1

Step 4: Adjust r and print out day of the week: The value of **r** is the day of the week, where 0 represents Sunday, 1 is Monday, . . . , 6 is Saturday. If the computed value of **r** is a negative number, add 7 to get a non-negative number between 0 and 6. Print out the day of the week.

You can check to be sure your code is working by looking at <http://www.timeanddate.com/calendar/>.

Sample output 1: Month and year out of range



```
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
===== RESTART: D:\code\Python\Playground\zelles.py =====
Enter month: 15
Enter the date: 10
Enter the year: 1000
Invalid date input.
>>>
>>> |
```

Ln: 68 Col: 4

Sample output 2: Leap year dates

```
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
===== RESTART: D:\code\Python\Playground\zelles.py =====
Enter month: 2
Enter the date: 29
Enter the year: 2020
a b c d = 12 29 19 20
w x y z r = 31 4 5 48 6
Day of the week: Saturday
>>>
===== RESTART: D:\code\Python\Playground\zelles.py =====
Enter month: 2
Enter the date: 29
Enter the year: 2010
Invalid date input.
>>> |
```

Ln: 61 Col: 4

Sample output 3: Valid dates

```
Python 3.8.1 Shell
File Edit Shell Debug Options Window Help
===== RESTART: D:\code\Python\Playground\zelles.py =====
Enter month: 1
Enter the date: 1
Enter the year: 2018
a b c d = 11 1 17 20
w x y z r = 28 4 5 15 1
Day of the week: Monday
>>>
===== RESTART: D:\code\Python\Playground\zelles.py =====
Enter month: 8
Enter the date: 15
Enter the year: 1947
a b c d = 6 15 47 19
w x y z r = 15 11 4 54 5
Day of the week: Friday
>>>
===== RESTART: D:\code\Python\Playground\zelles.py =====
Enter month: 7
Enter the date: 4
Enter the year: 1776
a b c d = 5 4 76 17
w x y z r = 12 19 4 81 4
Day of the week: Thursday
>>> |
```

Ln: 27 Col: 4