"""

Project for Week 4 of "Python Programming Essentials".

Collection of functions to process dates.

Be sure to read the project description page for further information

about the expected behavior of the program.

"""

import datetime

def days\_in\_month(year, month):

"""

Inputs:

year - an integer between datetime.MINYEAR and datetime.MAXYEAR

representing the year

month - an integer between 1 and 12 representing the month

"""

if year < datetime.MINYEAR or year > datetime.MAXYEAR:

return 0

if month < 12:

# Compute the two variables that gives the date of the given input

date1 = datetime.date(year, month, 1)

# To compute the next date we add a month to the one given

month = month + 1

date2 = datetime.date(year, month, 1)

# to determine the number of days in a month is to subtract the first

#of the given month from the first of the next month

difference = (date1 - date2) \* -1

# gives back just the days

return difference.days

if month == 12 :

# Compute the two variables that gives the date of the given input

date1 = datetime.date(year, month, 1)

# To compute the next date we add a month to the one given

year = year + 1

month = month - 11

date2 = datetime.date(year, month, 1)

# to determine the number of days in a month is to subtract the first

# of the given month from the first of the next month

difference = (date1 - date2) \* -1

# gives back just the days

return difference.days

else:

return "Error: Invalid date"

print(days\_in\_month(1996,6))

print(days\_in\_month(2020,2))

print(days\_in\_month(2021,2))

print(days\_in\_month(2016,2))

print(days\_in\_month(2013,12))

print(days\_in\_month(2013,13))

print(days\_in\_month(0,13))

print()

print("parte 2")

print("---------------")

def is\_valid\_date(year, month, day):

"""

Inputs:

year - an integer representing the year

month - an integer representing the month

day - an integer representing the day

Returns:

True if year-month-day is a valid date and

False otherwise

"""

max\_day = days\_in\_month(year, month)

if year < datetime.MINYEAR or year > datetime.MAXYEAR:

return False

if (month in range(1, 13)) and (day in range(1, max\_day + 1)):

return True

else:

return False

# and (year in range(datetime.MINYEAR, datetime.MAXYEAR))

print(is\_valid\_date(1996,6, 25))

print(is\_valid\_date(2020,2, 29))

print(is\_valid\_date(2021,2, 29))

print(is\_valid\_date(2016,2, 29))

print(is\_valid\_date(2013,12, 31))

print(is\_valid\_date(2013,13, 30))

print(is\_valid\_date(0,13, 30))

print()

print("parte 3")

print("---------------")

def days\_between(year1, month1, day1, year2, month2, day2):

"""

Inputs:

year1 - an integer representing the year of the first date

month1 - an integer representing the month of the first date

day1 - an integer representing the day of the first date

year2 - an integer representing the year of the second date

month2 - an integer representing the month of the second date

day2 - an integer representing the day of the second date

Returns:

The number of days from the first date to the second date.

Returns 0 if either date is invalid or the second date is

before the first date.

"""

#primero asignamos una variable para hacer check a si las fechas son validas

valid\_date1 = is\_valid\_date(year1, month1, day1)

valid\_date2 = is\_valid\_date(year2, month2, day2)

# asignamos variables para cada una de las fechas dadas los inputs

date1 = datetime.date(year1, month1, day1)

date2 = datetime.date(year2, month2, day2)

# hacemos el condicionante que las dos fechas sean validas y si es asi

#que arroje la diferencia entre los dias

if year1 < datetime.MINYEAR or year1 > datetime.MAXYEAR:

return 0

if year2 < datetime.MINYEAR or year2 > datetime.MAXYEAR:

return 0

if date2 < date1:

return 0

# chequeamos que las fechas sean validas

if valid\_date1 and valid\_date2:

diff = date2 - date1

return diff.days

# si no son validas devuelve 0

else:

return 0

print(days\_between(2015, 7, 8, 2018, 12, 15))

print(days\_between(2021, 9, 13, 2022, 8, 22))

print(days\_between(2011, 7, 8, 2018, 12, 15))

print(days\_between(2007, 12, 10, 2033, 12, 31))

print(days\_between(2007, 12, 10, 2000, 12, 31))

print(days\_between(2007, 12, 10, 2033, 12, 31))

print()

print("parte 4")

print("---------------")

def age\_in\_days(year, month, day):

"""

Inputs:

year - an integer representing the birthday year

month - an integer representing the birthday month

day - an integer representing the birthday day

Returns:

The age of a person with the input birthday as of today.

Returns 0 if the input date is invalid or if the input

date is in the future.

"""

#creamos las variables que que traen la fecha de hoy y las asginadas para el input

date1 = datetime.date(year, month, day)

todays\_date = datetime.date.today()

#sacamos la diferencia para que nos arroje los dias de diferencia y aplicamos

# el diff.day para qeu solo traiga los dias

diff = todays\_date - date1

agedays = diff.days

# hacemos las validaciones para que den los resultados 0 en caso de ser una fecha

# en el futuro o si no es valida la fecha input

valid\_date1 = is\_valid\_date(year, month, day)

if year < datetime.MINYEAR or year > datetime.MAXYEAR:

return 0

if todays\_date < date1:

return 0

if valid\_date1:

return (agedays)

else:

return 0

print(age\_in\_days(1994, 5, 18))