1. Leap or common year

Write a function that returns true or false depending on whether its input integer is a leap year or not.

A leap year is defined as one that is divisible by 4, but is not otherwise divisible by 100 unless it is also divisible by 400.

For example, 2001 is a typical common year and 1996 is a typical leap year, whereas 1900 is an atypical common year and 2000 is an atypical leap year.

1900%4 ==0 but
1900%100==0 and 1900%400 !=0 so not leap year
2000%4==0 and
2000%100 ==0 and 2000%400==0 so leap year

2. Combined number

Write a function accepting a list of non negative integers, and returning their largest possible combined number as a string. For example

given [50, 2, 1, 9] it returns "95021" (9 + 50 + 2 + 1)
given [5, 50, 56] it returns "56550" (56 + 5 + 50)
given [420, 42, 423] it returns "42423420" (42 + 423 + 420)

3. ISBN code

ISBN - International Standard Book Number

There are two ISBN standards: ISBN-10 and ISBN-13. Support for ISBN-13 is essential, whereas support for ISBN-10 is optional.
Here are some valid examples of each:

ISBN-10:
0471958697
0 471 60695 2
0-470-84525-2
0-321-14653-0

ISBN-13:
9780470059029
978 0 471 48648 0
978-0596809485

978-0-13-149505-0
978-0-262-13472-9

ISBN-10 is made up of 9 digits plus a check digit (which may be 'X') and ISBN-13 is made up of 12 digits plus a check digit. Spaces and hyphens may be included in a code, but are not significant. This means that 9780471486480 is equivalent to 978-0-471-48648-0 and 978 0 471 48648 0.

The check digit for ISBN-10 is calculated by multiplying each digit by its position (i.e., 1 x 1st digit, 2 x 2nd digit, etc.), summing these products together and taking modulo 11 of the result (with 'X' being used if the result is 10).

The check digit for ISBN-13 is calculated by multiplying each digit alternately by 1 or 3 (i.e., 1 x 1st digit, 3 x 2nd digit, 1 x 3rd digit, 3 x 4th digit, etc.), summing these products together, taking modulo 10 of the result and subtracting this value from 10, and then taking the modulo 10 of the result again to produce a single digit.

Basic task:
Create a function that takes a string and returns true if that is a valid ISBN-13 and false otherwise.

4. Friday 13th

Write a program to show that the 13th day of the month falls more often on a Friday than any other day of the week. The 1st of January 1973 was a Monday. You should aim at producing the clearest possible program, not the fastest.

[Zeller's Rule

`F=k+ [(13*m-1)/5] +D+ [D/4] +[C/4]-2*C where`

`k` is the day of the month.

`m` is the month number.

`D` is the last two digits of the year.

`C` is the first two digits of the year.

Note:

According to Zeller's rule the month is counted as follows:

March is 1, April is 2….. January is 11 and February is 12.

So the year starts from March and ends with February. So if the given date has month as January or February subtract 1 from the year. For example:

For 1st January 1998 subtract 1 from 1998 i.e. 1998-1=1997 and use 1997 for calculating D.

Discard all the decimal values and then find the final value of F.

After getting the value of F, divide it by 7.The value of F can be either positive or negative. If it is negative, let us suppose F = -15. When we divide by 7 we have to find the greatest multiple of 7 *less* than -15, so the remainder will be positive (or zero). -21 is the greatest multiple of 7 less than -15, so the remainder is 6 since -21 + 6 = -15.

Alternatively, we can say that -7 goes into -15 twice, making -14 and leaving a remainder of -1.If we add 7 since the remainder is negative i.e. -1 + 7 we again get 6 as remainder. After getting the remainder we can find the day of the week for the given date. Following are the values for the corresponding remainders:

Sun	Mon	Tue	Wed	Thurs	Fri	Sat
0	1	2	3	4	5	6

Examples for day calculation using Zeller's Rule:

Let us calculate the day for the following dates:

1st April 1983 and 27th February 2023.

A) 1st April 1983:

Here
k = 1
m=2
D=83
C=19.
Putting the values in the formula, we get,

```
F= 1+ [(13*2-1)/5] +83+83/4+19/4-2*19
= 1+ [(26-1)/5]+83+20.75+4.75-38
= 1+25+83+20+4-38                    (discarding the decimal values)
= 133-38
= 75
```

After calculating F divide it by 7 and get the remainder.
78/7=11 Quotient
5-Remainder
Therefore, the day on **1st April 1983** was **Friday** since the remainder is 5.

B) 2nd March 2004:

Here,
k = 2
m= 1
D= 04
C= 20.

Putting the values in the formula, we get,

```
F= 2+ [(13*1-1)/5] +04+04/4+20/4-2*20
= 2+ [(13-1)/5] +04+01+05-40
= 2+ [12/5] +10-40
= 2+2+10-40
= 14-40
= -26
```

(discarding the decimal values)

Here F is negative. So when we divide by 7 we have to find the greatest multiple of 7 *less* than -26, so the remainder will be positive (or zero). -28 is the greatest multiple of 7 less than -26, so the remainder is 2 since -28 + 2 = -26.

So, the remainder is 2.

Therefore, the day on **2nd March 2004** was **Tuesday** since the remainder is 5.

C) 27th February 2023:

Here,
k = 27
m = 12
D = 22 (Since month count starts from March)
C = 20

Putting the values in the formula, we get,

```
F = 27+ [(13*12-1)/5] +22+22/4+20/4-2*20
```

`= 27+ [(159-1)/5] +22+5.5+5-40`

`= 27+ [158/5] +22+5.5+5-40`

`= 27+ [31.6] + 22 + 5.5 + 5 – 40`

`= 27+ 31+22+5+5-40` *(discarding the decimal values)*

`= 90-40`

`= 50`

After dividing F by 7, we get remainder as 50/7=1.

Therefore, the day on **27th February 2023** is **Monday** since the remainder is 1.

5. Loops and if

Create a random number list of N (argument to the function) entries between 1 and 100 and find the closest number pair. Use two methods without using sort and using sort.

Hint: Without using sort**,** Use two cascading for loops. For every outer loop number, find the difference (using abs() function) with inner loop number. Inner loop can go through a sliced list starting with one after the outer loop number.

If the list is sorted, just one loop is enough to find the closest pair.

6. greatest common divisor (GCD)

Find the greatest common divisor of two numbers.
Hint: Use two methods Euclid's algorithm and Euclidean algorithm as in

Euclid's algorithm[edit]

Main article: Euclidean algorithm

The method introduced by Euclid for computing greatest common divisors is based on the fact that, given two positive integers a and b such that $a > b$, the common divisors of a and b are the same as the common divisors of $a - b$ and b.

So, Euclid's method for computing the greatest common divisor of two positive integers consists of replacing the larger number by the difference of the numbers, and repeating this until the two numbers are equal: that is their greatest common divisor.

For example, to compute $\gcd(48,18)$, one proceeds as follows:

$$\begin{array}{lllll} (48,18) & \to & (48-18,18)=(30,18) & \to & (30-18,18)=(12,18) \\ & \to & (12,18-12)=(12,6) & \to & (12-6,6)=(6,6). \end{array}$$

So $\gcd(48, 18) = 6$.

This method can be very slow if one number is much larger than the other. So, the variant that follows is generally preferred.

Euclidean algorithm[edit]

Main article: Euclidean algorithm

The biggest number that goes into 62 and 36 is 2
36
26
10
10
6
4

Animation showing an application of the Euclidean algorithm to find the greatest common divisor of 62 and 36, which is 2.

A more efficient method is the *Euclidean algorithm*, a variant in which the difference of the two numbers a and b is replaced by the *remainder* of the Euclidean division (also called *division with remainder*) of a by b.

Denoting this remainder as $a \bmod b$, the algorithm replaces (a, b) by $(b, a \bmod b)$ repeatedly until the pair is $(d, 0)$, where d is the greatest common divisor.

For example, to compute gcd(48,18), the computation is as follows:

$$\begin{aligned}(48,18) &\to (18, 48 \bmod 18) = (18,12)\\ &\to (12, 18 \bmod 12) = (12,6)\\ &\to (6, 12 \bmod 6) = (6,0).\end{aligned}$$

This again gives $\gcd(48, 18) = 6$.

7. roman numerals

Write a python function to convert roman numerals into decimal value.

I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Hint:
If Current character is bigger than the previous character, use subtraction otherwise addition.
E.g. IX = 9, IV = 4, XC = 90
LX = 60

Decimal number to roman numerals conversion

For decimal number x:

1. From the following table, find the highest decimal value v that is less than or equal to the decimal number x
   and its corresponding roman numeral n:

2.

Decimal value (v)	Roman numeral (n)
1	I
4	IV
5	V
9	IX
10	X

40	XL
50	L
90	XC
100	C
400	CD
500	D
900	CM
1000	M

1. Write the roman numeral n that you found and subtract its value v from x:
   x = x - v
2. Repeat stages 1 and 2 until you get zero result of x.

Example #1

x = 36

Iteration #	Decimal number (x)	Highest decimal value (v)	Highest roman numeral (n)	Temporary result
1	36	10	X	X
2	26	10	X	XX
3	16	10	X	XXX
4	6	5	V	XXXV
5	1	1	I	XXXVI

Example #2

$$x = 2012$$

Iteration #	Decimal number (x)	Highest decimal value (v)	Highest roman numeral (n)	Temporary result
1	2012	1000	M	M
2	1012	1000	M	MM
3	12	10	X	MMX
4	2	1	I	MMXI
5	1	1	I	MMXII

Example #3

$x = 1996$

Iteration #	Decimal number (x)	Highest decimal value (v)	Highest roman numeral (n)	Temporary result
1	1996	1000	M	M
2	996	900	CM	MCM
3	96	90	XC	MCMXC
4	6	5	V	MCMXCV
5	1	1	I	MCMXCVI