Exercises

Write a code that prints every number from 100 to 1000 that is divisibile to 5 AND 6. (Use the module operator %)

The code must print the numbers in lines of 10 numbers, and with a white space between the numbers.

Write a code that asks the user two integers, then finds the greatest common divisor and print it.

The Ackermann function A(m,n) is defined by:

$$
A(m,n) = \begin{cases} n+1 & \text{if } m=0; \\ A(m-1,1) & \text{if } m>0 \text{ and } n=0 \\ A(m-1,A(m,n-1)) & \text{if } m,n>0 \end{cases}
$$

Write a code that compute the Ackermann function, and use it to compute A(3,4) =125. Try to compute A(4,3)…

A word is palindrome if it can be read in both directions. More analitically a palindrome word has the same character at the beginning and the end, and the remaining characters palindrome

Write a simple function *is_palindrome* that accept a string as input argument, and returns true if it’s palindrome, false otherwise.

The following code searches the value 2**5 in a given list of numbers, and if founds it returns the index.

```
L=[1,2,4,8,16,32,64]
X=5
found=False
i=0
while not found and i<len(L):
	if 2**X ==L[i]:
		found=True
	else:
		i=i+1
if found:
	print 'at index ', i
else:
	print X, ' not found'
```

Rewrite the code in a "python-like" fashion:

1. Use a *while/else* loop, so you can remove the control variable *found* e the final *if*

```
L=[1,2,4,8,16,32,64]
X=5
found=False
i=0
while not found and
i<len(L):
        if 2**X ==L[i]:
                found=True
        else:
                i=i+1
if found:
        print 'at index ', i
else:
        print X, ' not
found'
```

Rewrite the code in a "python-like" fashion:

2. Use a *for/else* loop so you can remove the explicit addressing of the list items (Suggestion: L.index(X) returns the index of the first X contained in the list L)

```
L=[1,2,4,8,16,32,64]
X=5
found=False
i=0
while not found and
i<len(L):
        if 2**X ==L[i]:
                found=True
        else:
                i=i+1
if found:
        print 'at index ', i
else:
        print X, ' not
found'
```

\

Rewrite the code in a "python-like" fashion:

3. Now remove any loop…

```
L=[1,2,4,8,16,32,64]
X=5
found=False
i=0
while not found and
i<len(L):
        if 2**X ==L[i]:
                found=True
        else:
                i=i+1
if found:
        print 'at index ', i
else:
        print X, ' not
found'
```

Write a Python script that is able to compute the natural logarithm of a number that is passed as input

For example if we run

```
calcola.py 1.0 xyz -0.9 2.1
```

the script must prints as results:

```
ln(1)=0
ln(xyz) non numeric input
ln(-0.9) not allowed
ln(2.1)=0.741937
```

- Note that you can loop over the input parameters in 4 diferent ways:
  1. `for r in sys.argv[1:]`
  2. `for` (with a integer index) on `sys.argv`
  3. `while` (with an integer index) on `sys.argv`
  4. `while 1:` that stops when `sys.argv[i]` becomes invalid

  - Write 4 scripts with the different loops on the input parameters
  - Always check that the scripts is used with at least one parameter.
  - Remember that the log() function is defined in the math module