```python
# -*- coding: utf-8 -*-
"""
Created on Sat Oct 10 22:35:15 2015
@author: hina
Reference: https://docs.python.org/3/tutorial/index.html
"""

# If you quit from the Python interpreter and enter it again,
# the definitions you have made (functions and variables) are lost.
# Therefore, if you want to write a somewhat longer program, you are better off
# using a text editor to prepare the input for the interpreter and running it
# with that file as input instead. This is known as creating a script.
#
# As your program gets longer, you may want to split it into several files for
# easier maintenance. You may also want to use a handy function that you’ve
# written in several programs without copying its definition into each program.
#
# To support this, Python has a way to put definitions in a file and use them
# in a script or in an interactive instance of the interpreter.
# Such a file is called a module.
#
# Definitions from a module can be imported into other modules or into the
# main module (the collection of variables that you have access to in a script
# executed at the top level and in calculator mode).
#
# A module is a file containing Python definitions and statements.
# The file name is the module name with the suffix .py appended.
# Within a module, the module’s name (as a string) is available as the value
# of the global variable __name__.

# Notes:
#  - When a module is imported, the interpreter will look for it per sys.path.
#  - Python comes with a library of standard modules, described in a separate
#    document, the Python Library Reference. Key among it is the sys module.

# For instance, we have created a module called fibo (in fibo.py ) to calculate
# the fobonacci series given n.

# Now you can import the module, and then call the function defined in the module.
import fibo
fibo.fib (10)

# Or you can import the specific function from module, and then call it directly.
from fibo import fib, fib2
fib (20)
lst = fib2 (30)
print (lst)

# Or you can import all functions defined in fibo module and call them.
# This is typically not recommended since you would want to maintain control
# over what is imported into your namespace.
from fibo import *
fib (40)
```