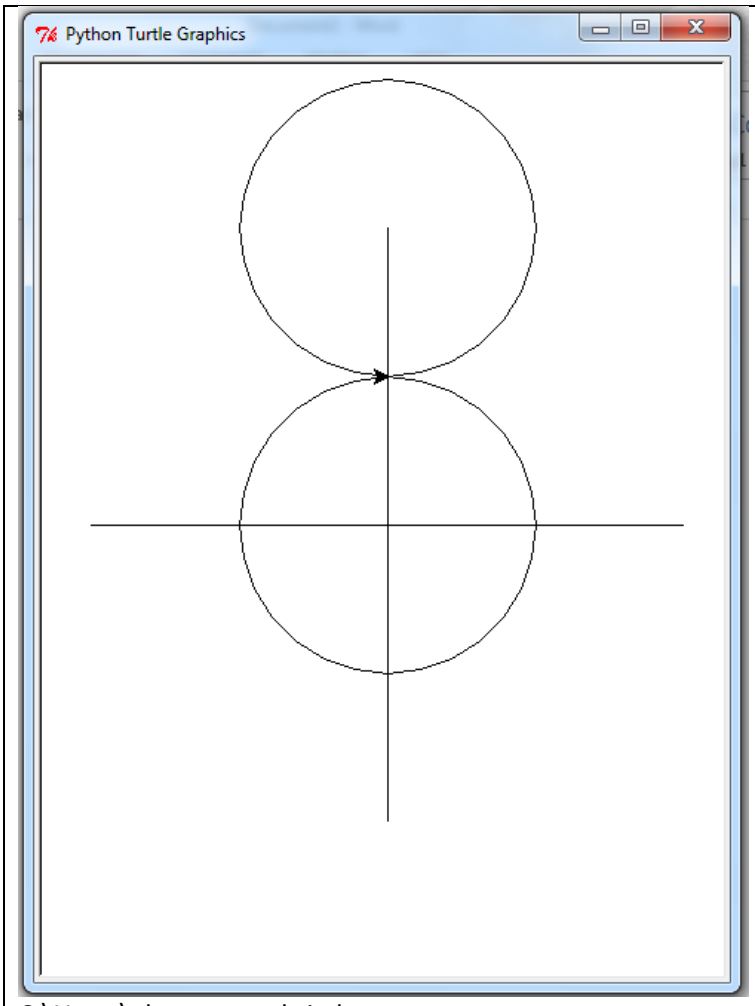
Using Python 2.7 from a command line:

```
c:\Python27>python
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import imp
>>> foo = imp.load_source('circle', 'c:\Users\abc\circle.py')
>>> foo.twoCircles()
0
```

Produces graphic output, below:

Second column is the source code for the circles



C:\Users\abc>notepad circle.py

```
C:\Users\abc>type circle.py
import turtle

def twoCircles():
    t = turtle.Turtle()

    # Draw the X and Y axes
    t.up() # don't draw yet
    t.goto(0, 200)
    t.down()
    t.right(90)
    t.forward(400)
    t.up()
    t.goto (-200, 0)
    t.down()
    t.left(90)
    t.forward(400)

    # Draw two circles, first at the origin

    t.up()
    t.goto(0, -100)
    t.down()
    t.circle(100)

    # Draw the second of the two cirles

    t.up()
    t.goto(0, 100)
    t.down()
    t.circle(100)
    return 0
```

C:\Users\abc>notepad circle.py – used to edit program

Notes:

1. The program is stored in a file called circle.py – in this example
2. The name of the module is twoCircles()
3. The section highlighted in yellow, above is used within the Python 2.7 command line interpreter
4. Importing a module using a full path is taken from stackoverflow; displayed on next page

The sections below demonstrate how to import a module given the full path, for various versions of Python – taken from the website: *stackoverflow* as well as the Python documentation:

https://stackoverflow.com/questions/67631/how-to-import-a-module-given-the-full-path

https://docs.python.org/

For Python 3.5+ use:

```
import importlib.util

spec = importlib.util.spec_from_file_location("module.name", "/path/to/file.py")

foo = importlib.util.module_from_spec(spec)

spec.loader.exec_module(foo)

foo.MyClass()
```

For Python 3.3 and 3.4 use:

```
from importlib.machinery import SourceFileLoader



foo = SourceFileLoader("module.name", "/path/to/file.py").load_module()

foo.MyClass()
```

(Although this has been deprecated in Python 3.4.)

For Python 2 use:

```
import imp



foo = imp.load_source('module.name', '/path/to/file.py')

foo.MyClass()
```