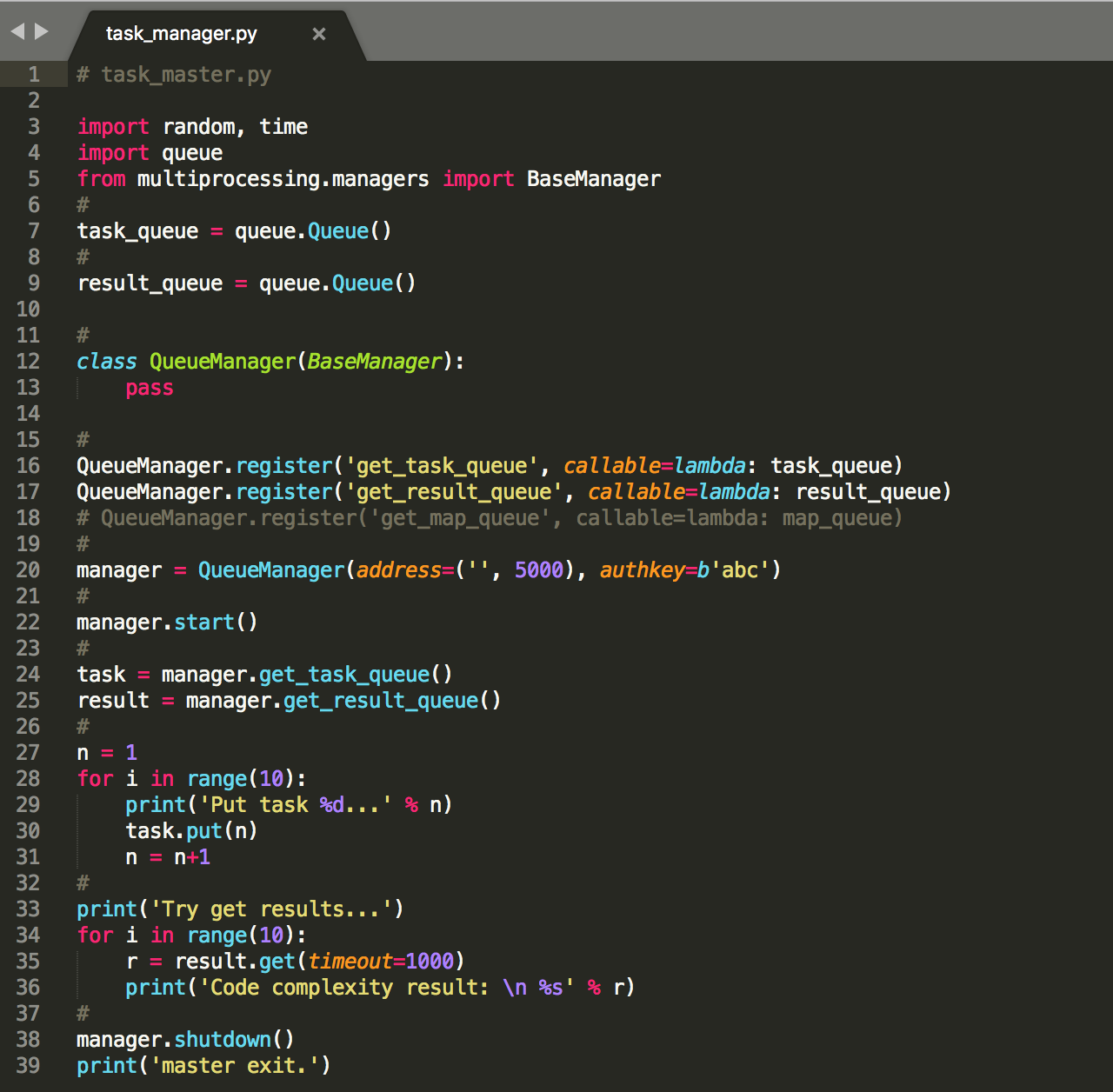
Python REST Service Report

1.REST Service

The purpose of this project is to provide a baseline demonstration of the use of python distribute system with python radon library in the context of computing code complexity with a distribute system. The radon library provides is a brilliant tool that support measure the cyclomatic complexity of codes. A simple message passing abstraction forms the basis of all communication.

This project provides a command line way to start the service. The distribute system using the work-stealing pattern described in http://www.well-typed.com/blog/71/. This system use multiprocessing package in python library.

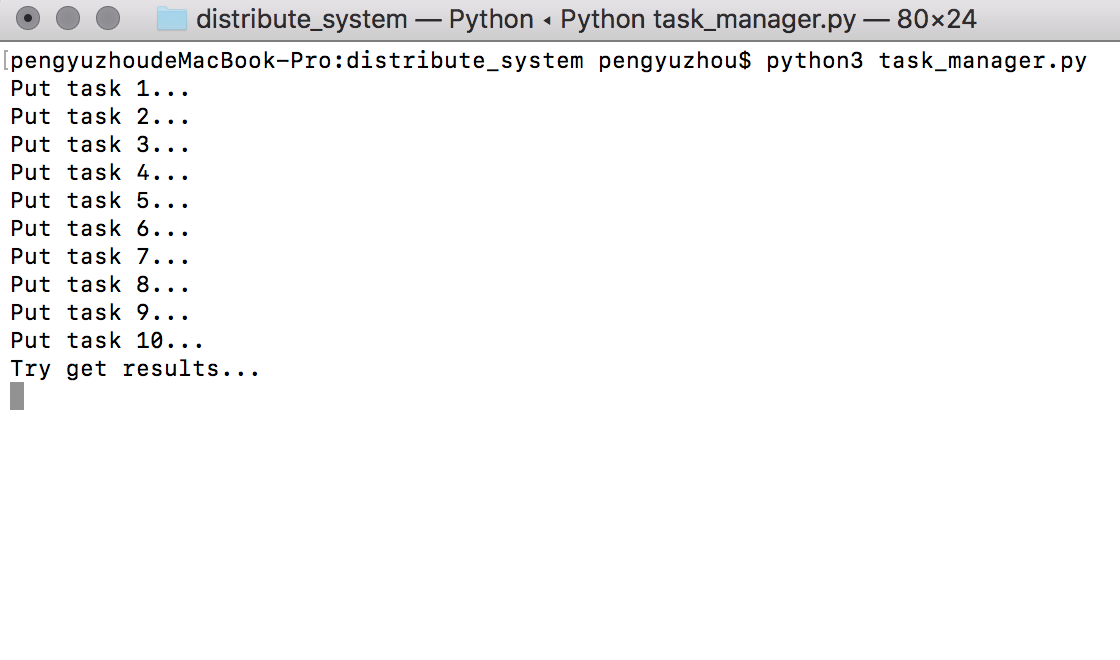


*snippet of task\_manager.py*

2. The start of python REST Service

In order to start and build the python REST service. We need to start task\_manager.py first.

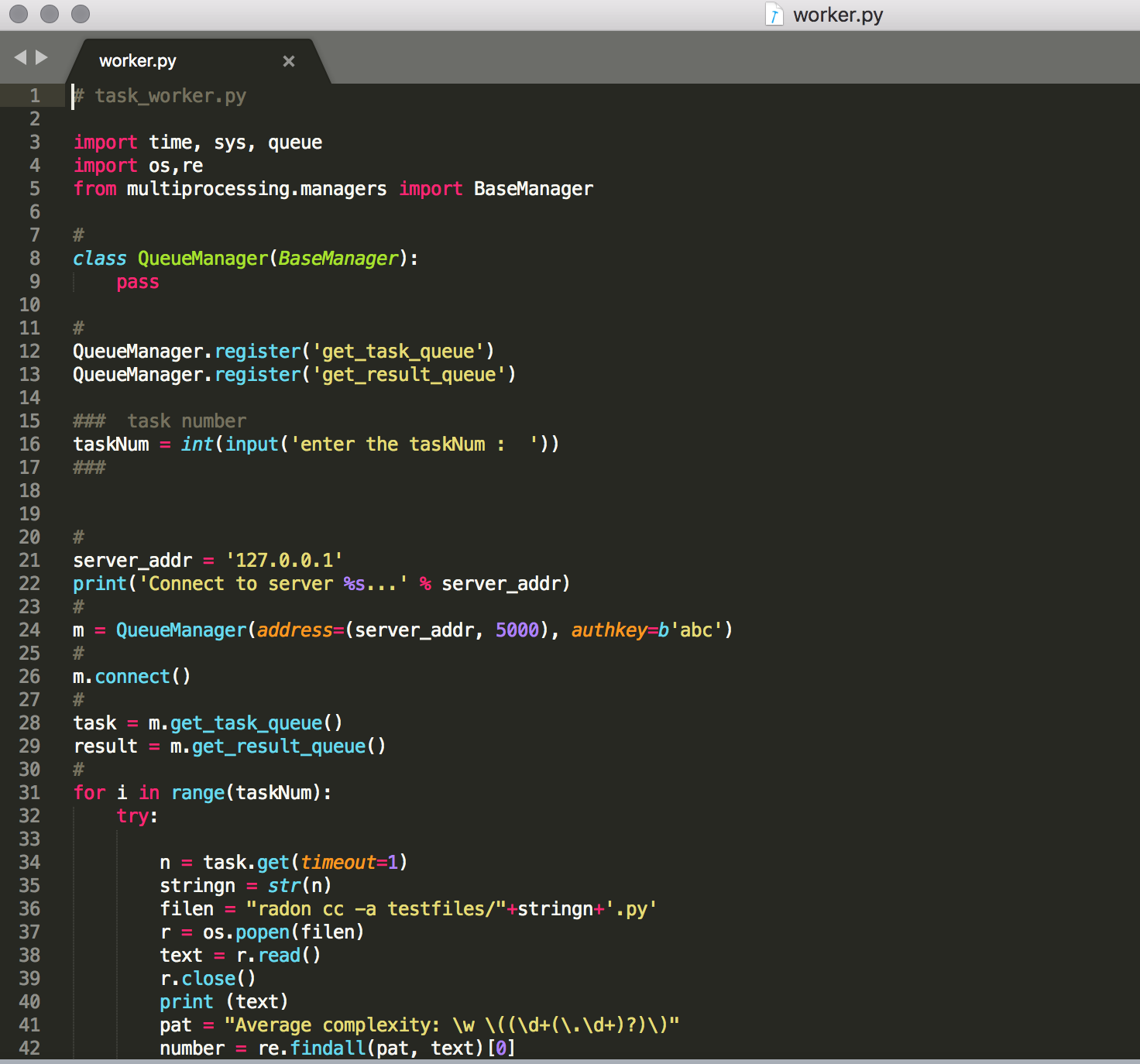
Now, we need to start our REST service, use python3 task\_manager.py to start the REST service. The service will put tasks in a list and let workers to acquire tasks.



*snippet of starting task\_manager.py*

Then, we will start worker nodes.

We assume multiple workers from different distribute systems. And we have worker.py as a worker, we could create as many workers as we want.



*snippet of starting task\_manager.py*

We can start our server and multiple worker nodes, then we could allocating tasks to worker nodes, worker nodes will read python files in testfiles folder and calculate the code complexity of these files and return the results of them.

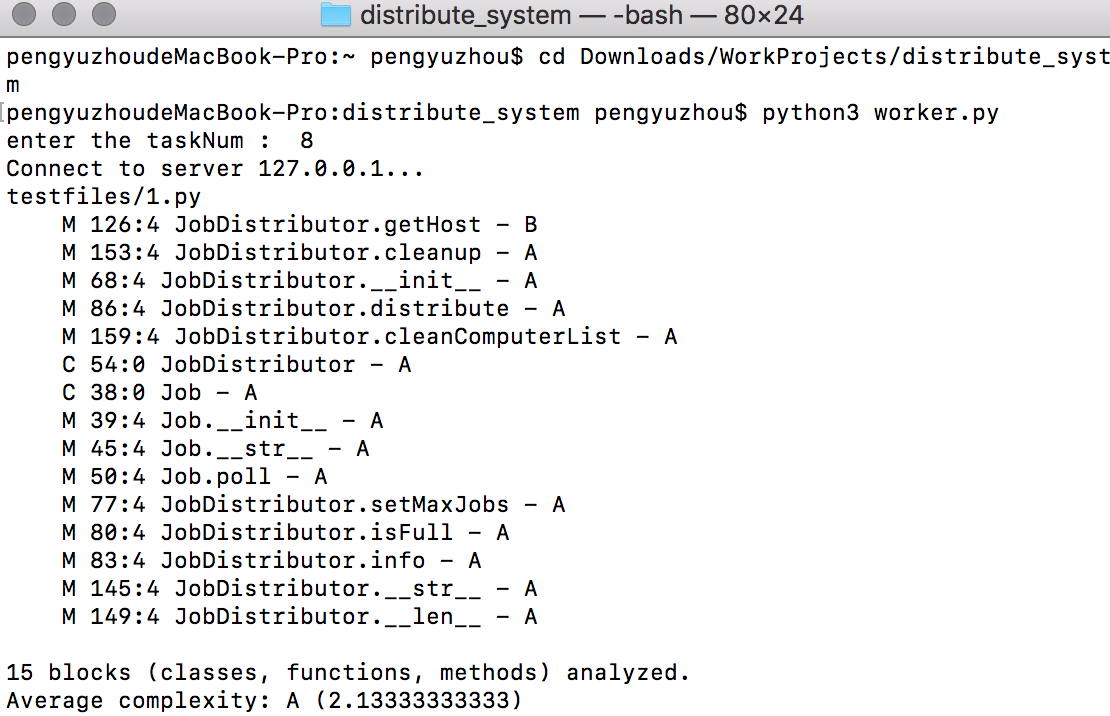
We can use following codes to start worker nodes in different terminals:

python worker.py

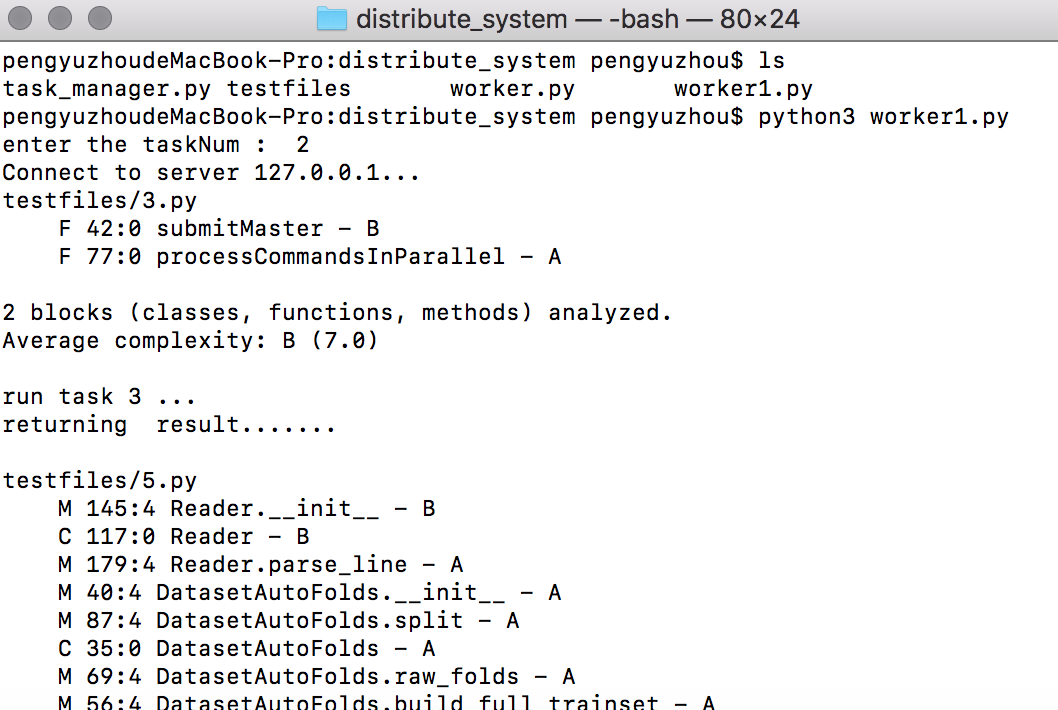
python worker1.py

…….

Then we can start multiple worker nodes.

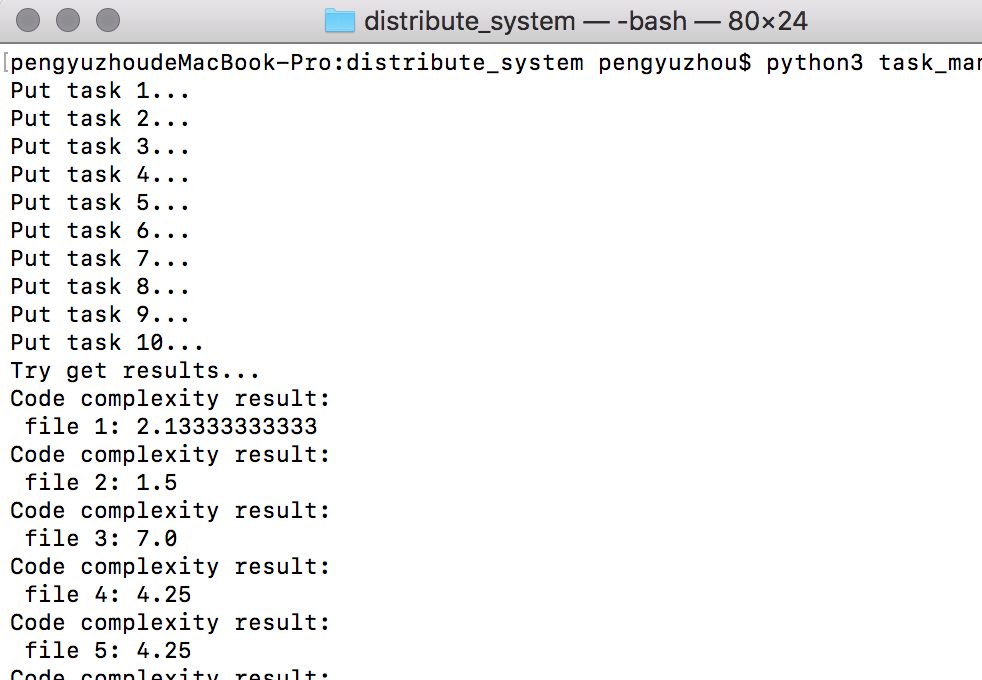


*start worker node*

**

*start worke1 node*

And then the manager will receive the code complexity computing result from workers.



*output of manager node*

Finally, worker nodes and manager node exit.

We could adjust the task number of each worker node.

3.Evaluate the result.

result:

1. With the increase of nodes, the processing time of REST service will decrease first and then increase.
2. With the increase of node number, the memory occupy will slowly increase.
3. In low memory environment, for example, I’m testing REST service when my computer is running other apps like PyCharm and Chrome. the running time will be influenced by number of nodes, the running time will increase quickly because of the increasing of node number.