test show python source code in pdf
here comes the code

```
import queue
import threading
import sys
import time
import urllib

#thread obj in thread pool
class MyThread(threading.Thread):
  def __init__(self, workQueue, resultQueue,timeout=2):
    threading.Thread.__init__(self)
    self.timeout = timeout#time that a thread wait for a
        queue
    self.setDaemon(True)#stop with main thread
    self.workQueue = workQueue
    self.resultQueue = resultQueue
    self.start()
  def run(self):
    #continuously run until workQueue empty
    while True:
      try:
        #get a job from workQueue, do it and add res to
            resultQueue
        callable, args=self.workQueue.get(timeout=self.
            timeout)
        print('{} running, job={}, parameters={}'.format(
            self.getName(),callable,args))
        res = callable(args)
        #self.resultQueue.put(res+" | "+self.getName())
      except queue.Empty:
        break
      except :
        print(sys.exc_info())

class ThreadPool:
  def __init__(self,num_of_threads=10):
    self.workQueue = queue.Queue()
    self.resultQueue = queue.Queue()
    self.threads = []
    self.__createThreadPool(num_of_threads)
  def __createThreadPool(self, num_of_threads ):
    for i in range( num_of_threads ):
      thread = MyThread( self.workQueue, self.resultQueue
          )
```

```
        self.threads.append(thread)
    def wait_for_complete(self):
      while len(self.threads):
        thread = self.threads.pop()
        if thread.isAlive():
          thread.join()
    def add_job(self, callable, *args):
      self.workQueue.put( (callable,args) )

def getNet1(rid):
  time.sleep(0.1)
  #print('getNet1 of {}'.format(rid[0]))
  return rid

def test():
  #test data:10 jobs,3 threads
  nJob=11
  nThread=3
  tp = ThreadPool(nThread)
  start=time.time()
  for i in range(nJob):
    tp.add_job(getNet1, i)
  stop=time.time()
  print('{} cost to add {} jobs'.format(stop-start,nJob))
  tp.wait_for_complete()

if __name__=='__main__':
  test()
```