- day05知识体系
  - 多任务
    - 目的
    - 并行
    - 并发
  - 线程
    - 概念
      - 是一种多任务方式
    - 创建
      - threading.Thread类实例对象
      - threading.Thread子类中实现run放的的实例对象
      - 主线程 子线程
    - 调度序列是无序的
      - windows mac linux不同的操作系统调度策略不同 执行顺序不同
    - 主线程等待子线程执行完成
      - 默认情况下
      - thd.join()主动等待
    - 同一个进程内部的多个线程共享全局数据
      - 数据竞争
    - 多个线程按照一定顺序<秩序>运行
  - 多线程同步
    - 互斥锁
      - 一种多线程同步方式
      - 保证在任一时刻，只能有一个线程访问互斥锁
      - 创建 mutex = threading.Lock()
      - 申请 mutex.acquire()
      - 释放 mutex.release()
      - 缺点–多任务效率低 容易死锁
    - 死锁问题
      - 主要原因
        - 不正确的申请方式
          - 多个任务申请多个资源 造成相互等待的现象
          - 解决方式–银行家算法
        - 不正确的释放方式
          - 缓兵之计–使用超时等待锁 防止被无限阻塞
          - 根本上 规范释放锁
  - 案例–多任务聊天程序