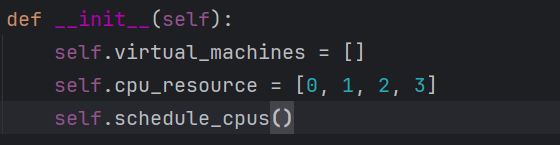
1. 虚拟机监控器（Hypervisor）实现: 设计一个轻量级的Hypervisor框架，支持加载和运行虚拟机实例。实现硬件资源的抽象化和虚拟化，包括CPU、内存和I/O设备。

操作指南原有代码框架已经实现

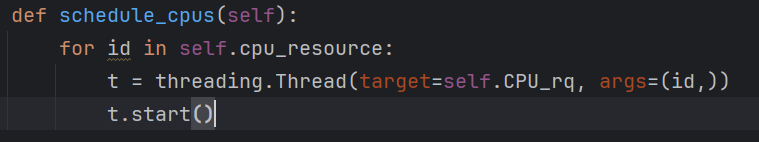
1. 虚拟CPU管理: 开发虚拟CPU调度器，模拟CPU时间片分配给不同的虚拟机。实现虚拟CPU状态管理，包括创建、运行、挂起和恢复等状态。

实现思路：hypervisor对象初始化时，设定可以使用的CPU资源，并执行CPU调度程序。

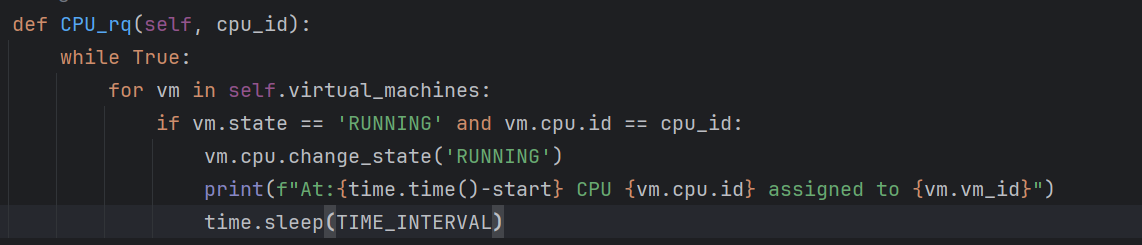


CPU调度程序的实现：

为每一个CPU创建一个轮询线程，每隔一个固定的时间将此CPU的占用切换给其他虚拟机。

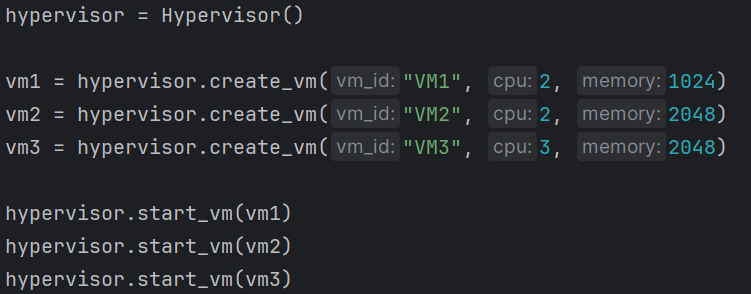


为每一个核创建一个轮训线程。



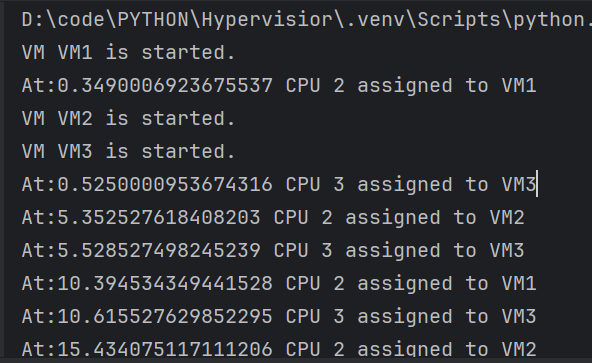
测试程序：

创建三个虚拟机并运行，其中VM1,2占用CPU2，VM3占用CPU3。在运行过程中，设置轮训时间片为5s。

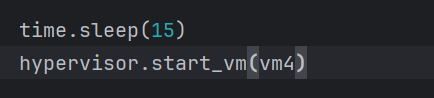


运行结果如下：

VM1,2每隔5s会完成CPU占用的切换，VM3独占CPU，不会完成CPU占用权的切换

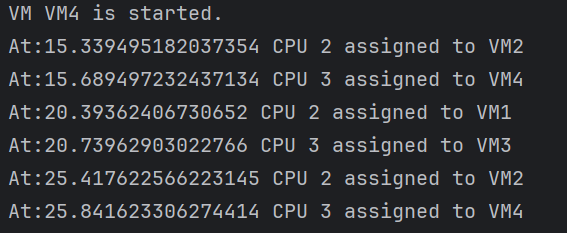


设定15s之后VM4开始运行，vm4同样占用CPU3。也就是说在15s之后CPU3由vm3,4共享。



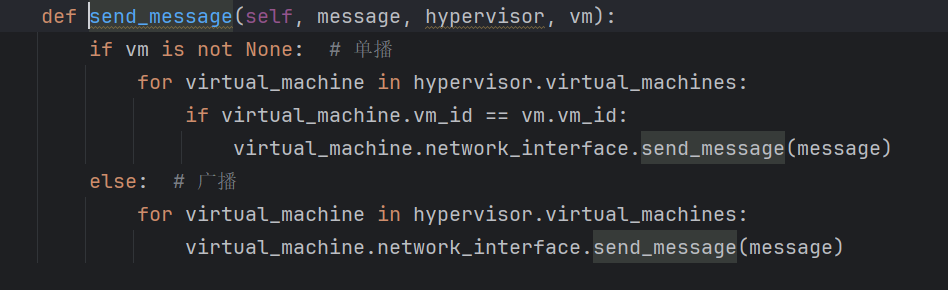
运行结果如下：

在VM4启动之后，CPU调度程序也能完成对CPU3的时间片分配，每隔5s在VM3,4间切换。

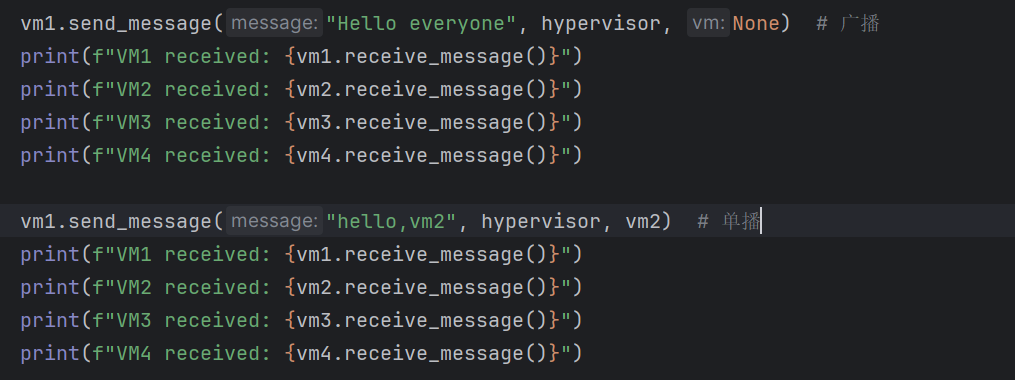


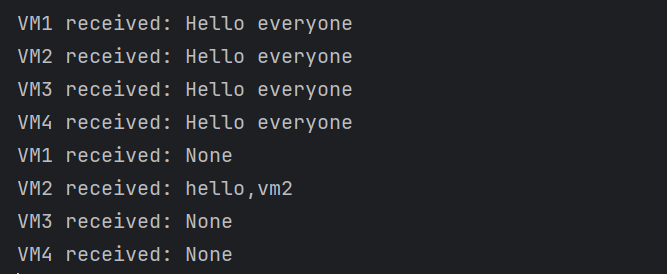
1. 虚拟机间通信与协作:设计并实现虚拟机间的通信机制，支持数据交换和协作。实现虚拟网络或共享内存机制，以支持虚拟机间的高效通信。

设计思路：每个虚拟机拥有独立的网卡和队列，虚拟机之间的通信可以使用单播和广播，单播的时候仅指定的虚拟机能收到消息，广播情况下所有的虚拟机均能收到消息。



测试程序如下：第一次vm1发送广播消息，所有虚拟机都收到消息。第二次仅对虚拟机2发送消息，仅2收到。

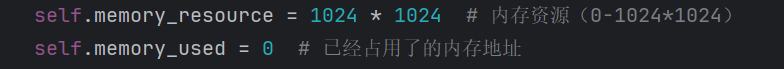




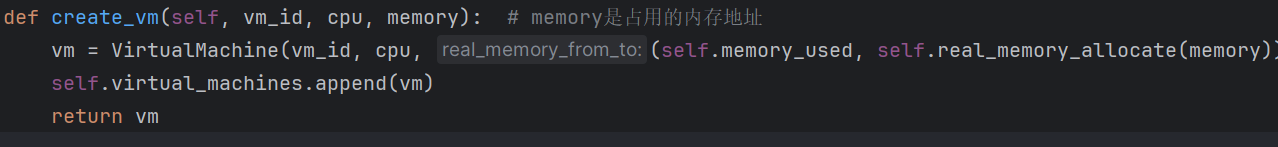
1. 虚拟内存管理:设计并实现虚拟内存管理器，为每个虚拟机提供独立的地址空间。实现地址转换机制，将虚拟地址映射到物理地址。

设计思路：

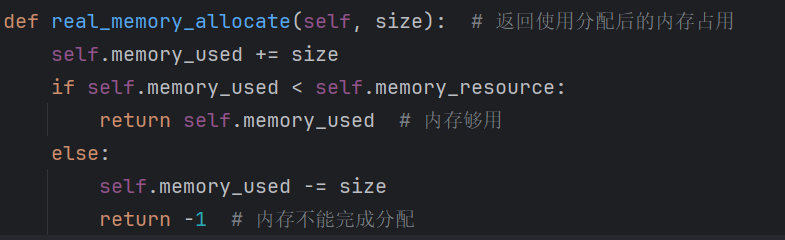
Hypervisor占用一块真实内存地址，在vm创建的时候可以为其分配地址。地址分配的信息储存在VirtualMemory类的memory\_map中，键为vm占用内存的size，值为真实内存的起止地址，由vm在创建时计算分配。



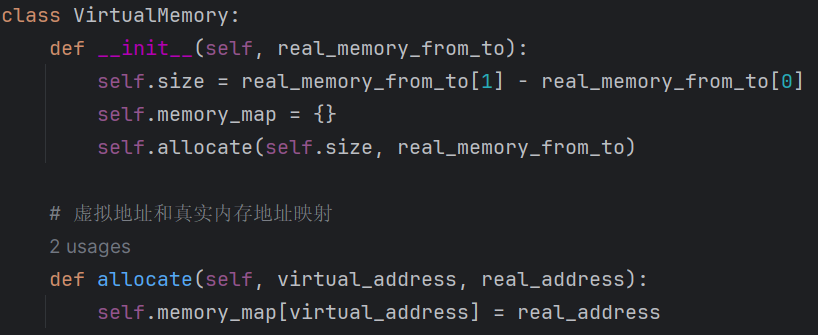
可使用的内存资源和已经使用的内存资源。初始化

在创建vm时根据传入的vm需求大小和当前hypervisor已经使用的内存大小，分配真实内存地址。

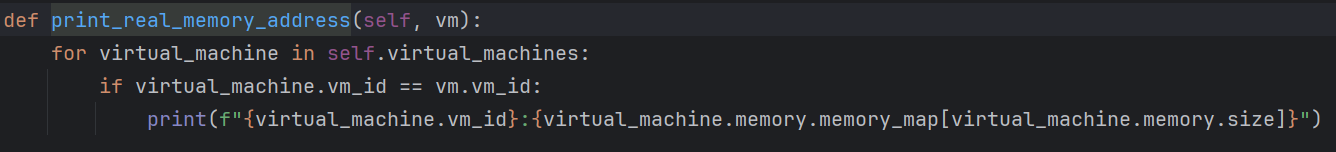
计算的方法如下：



虚拟机初始化时，创建内存。将真实起止地址传入Virtual\_Memory创建中，内存映射表的键为vm的大小（size），值对应真实地址的元组。



成功储存内存分配映射后，可以使用hypervisor内的print\_real\_memory\_address查看vm真实内存地址。



测试程序：

创建四个虚拟机，分别输出其真实内存地址。



运行结果如下：

