这份作业是关于计算机体系结构的研究生课程，具体是关于缓存（Cache）的模拟。作业的目标是使用Python编程语言来创建两种类型的缓存：直接映射缓存（Direct-Mapped Cache）和集合关联缓存（Set-Associative Cache）。这个缓存模拟器主要处理内存地址，而不关心数据的实际值。作业的具体要求如下：

1. **环境准备**：
    - 需要在计算机上安装Python3。建议使用Ubuntu系统，并且可能需要通过pip3安装numpy库。
2. **直接映射缓存（Direct Mapped Cache）**：
    - **热身练习**：给定一系列的32位内存地址引用，要求根据直接映射缓存的规则（如块大小和缓存总大小）来判断每个引用是命中（hit）还是未命中（miss），并计算命中率。
    - **直接映射缓存模拟**：需要实现一个简单的直接映射缓存模拟器。具体需要实现以下几个Python函数：
        - `find_set`：根据地址返回集合（set）的值。
        - `find_tag`：根据地址返回标签（tag）的值。
        - `find`：根据地址判断是命中还是未命中。
        - `load`：在缓存未命中时，根据地址加载数据到缓存。
    - **验证**：验证通过Python缓存模拟器得到的命中率是否与热身练习中计算的一致。
3. **集合关联缓存（Set-Associative Cache）**：
    - **热身练习**：类似于直接映射缓存的热身练习，但是这次是集合关联缓存，并且需要考虑替换策略（如最近最少使用LRU）。
    - **集合关联缓存模拟**：需要实现一个简单的集合关联缓存模拟器。需要实现的函数与直接映射缓存类似，但是需要处理多个集合。
    - **验证**：验证通过Python缓存模拟器得到的命中率是否与热身练习中计算的一致。

作业的提交物包括：

- 两个Python文件：`cache_dm.py`（直接映射缓存）和`cache_sa.py`（集合关联缓存）。
- 一份报告，包含：
    - 代码的输出结果。
    - 对代码逻辑的解释和输出的澄清。
    - 热身练习和验证部分的工作。

这份作业的目的是让学生通过实践来更深入地理解课堂上学到的缓存概念。通过编写和测试缓存模拟器，学生可以更好地理解缓存的工作原理，如地址映射、命中和未命中的处理，以及不同的缓存替换策略。