一、 Linux 操作系统采用的是 32 位线性地址模式，将内存物理空间映射到虚

拟地址空间。在 32 位线性地址的 4GB 虚拟空间中，1G 系统空间由所有进程共用，

3G 用户空间由每一个进程独占。请结合本课程所学习的存储管理理论知识，对

Linux 操作系统的地址映射、虚拟内存实现机制、请求分页实现过程、内存分配

与回收、换页策略等进行详细阐述。

二、 如果 CPU 在 I/O 完成之前多次忙于等待的循环，轮询 I/O 的完成可能会浪

费大量的 CPU 周期。但如果 I/O 设备已经准备好，轮询的效率 要比中断的效率

更高。᧿述一种混合策略，结合轮询、休眠和中断为 I/O 设备服务。对于这三种

策略中的每一种（纯轮询、纯中断、 混合），᧿述一种计算环境，在这种环境中，

该策略比其他任何一种都更有效率

**三、 根据本课程所学习的进程调度的知识，为一个有 N 个进程并发执行系统**

**设计进程调度方案，并给出详细的设计细节。要求如下：**

**a、问题᧿述**

1）进程调度算法：系统采用**多级反馈队列**方式进行调度，一共有三级就绪队

列，一个进程被创建后，首先进入第一级就绪队列进行排队（就绪状态 Wait），

系统按照先来先服务的方式调度进程运行，从此一级就绪队列获得 CPU 的进程，

运行一个时间片单位（运行状态 Run），如果在此时间片内，进程运行完毕，进程

到达完成 F（Finish）状态；如果没有运行完成，该进程进入第二级就绪队列（就

绪状态 Wait），也按照先来先服务的方式排队，在此一级就绪队列调度进程运行

（运行状态），时间片的长度是第一级就绪队列中时间片长度的两倍（即连续运

行两个时间片单位）。如果在此一级时间片内完成，进程达到完成 F（Finish），

如果在此一级时间片内没有完成，则该进程进入第三级就绪队列中排队（就绪状

态 Wait），仍然采用先来先服务方式排队，在第三级就绪队列中获得 CPU 的进程，

其运行时间片的长度为第二级就绪队列时间片长度的两倍（即连续运行四个一级

就绪队列时间片），如果在此一级时间片内完成，进程达到完成 F（Finish）状态

退出，如果没有运行完毕，则重新回到第三级就绪队列排队等待调度。

**三级就绪队列调度规则**，第一级就绪队列的优先级最高，如果此级队列非空，3

则优先调度处于此一级队列中的进程。当第一级队列为空时，从第二级就绪队列

中选择队首进程进行调度。当第一级和第二级就绪队列同时为空时，从第三级就

绪队列中选择进程调度。

2）第一级就绪队列的时间片长度自己设定，第二级就绪队列时间片长度是第

一级的二倍，第三级就绪队列时间片长度是第一级的四倍。

3）进程的状态：就绪状态 W（wait），运行状态 R（run），完成状态 F（Finish）。

4）进程控制块 PCB 所包含的信息包括：进程名、创建时间、需要 CPU 运行

时间、已用 CPU 时间、进程状态等内容。

**b、题目要求**：根据以上᧿述，设计一个实现此进程调度系统的方案。方案的要

素：

1）用 C 语言或类 C 语言方式给出数据结构的设计，

2）给出此调度系统的逻辑设计（有多少个函数以及他们的调用关系）、流程

图，并给出功能函数的设计细节（可以采用伪代码算法的方式，也可以给出真实

的代码）。

3）此题目基本要求是给出设计方案，如果你能用 C 语言实现，将获得更高的

成绩。

如果你只完成了设计方案（各种流程图和伪代码算法），你需要ᨀ供一个实

例，在逻辑上把整个调度过程讲解一遍。

如果你用 C 语言实现了该系统，你需要把输入和输出的截图粘贴到作业中。

**四、 请根据本课程所学的文件系统管理知识设计一个简单文件系统。**

**该系统要求具备基本的文件处理功能，包括文件的建立、打开、删除、关闭、**

**读、写、查询等基本功能。**要求如下：

**a、问题᧿述**

1）在内存中建立一个虚拟磁盘空间作为文件存储器（模拟磁盘），设计一个

多用户二级目录文件系统。

2）为了简化文件系统的实现过程，文件的具体读写过程可简化为用输出语句4

表述，不进行实际的读写操作。

3）采用二级目录结构，即设置主目录（MFD）和用户文件目录（UFD）。主目

录和用户文件目录均采用结构体数据结构。主目录（MFD）包含主目录用户名、

用户密码、用户文件目录数组等内容，用户文件目录（UFD）包括文件名、文件

读写属性（

r、w 和 e）、文件创建日期、文件长度等内容。

主目录（MFD）结构体：

struct mdf

{

char uname[10]; //用户名

char password[20]; //用户登录密码

UF Udir; //用户文件目录

}UFD[10]; //主目录

用户文件目录结构体：

typedef struct ufd

{

char fname[10]; //用户文件名称、

int flag; //文件存在标志

int fprotected[3]; //文件保护标志数组,数组元素依次为 r\w\e

//1 表示允许，0 表示不允许

char fdate[8]; //文件创建日期

int flength; //文件长度

}ufd,UF[10];

4）要求所设计文件文件系统ᨀ供以下文件操作：文件的创建 create、文件

的删除 delete、文件的打开 open、文件的关闭 close、读文件操作 read、写文

件操作 write。这些文件操作功能均以函数的形式进行设计和实现。对文件进行

打开、关闭、读、写等操作时，要先判断文件是否已经打开。

用户打开文件结构体:

Struct afd

{5

char opname[10]; //打开文件名

int flag; //文件存在标志

char opfprotect[3] //文件保护标志数组,数组元素依次为 r\w\e

int rwpoint; //文件读写指针

}AFD[5]； //已打开文件表，文件系统最多允许同时打开 5 个文件

**b、题目要求：**

1）用 C 语言或类 C 语言方式（伪代码）给出该简单文件系统的设计过程。

2）给出此文件系统的逻辑设计、流程图，并给出功能函数的设计细节（可以

采用伪代码方式，也可给出真实代码）。

3）此题目基本要求是给出设计方案，如果你能用 C 语言实现，将获得更高的

成绩。

如果你只完成了设计方案（各种流程图和伪代码算法），需要在逻辑上把此简

单文件系统管理过程讲解一遍。

如果你用 C 语言实现了该系统，你需要把系统运行截图粘贴到作业中