操作系统课程设计指导书

（第六版）

田卫东，李琳，罗月童，周红鹃，刘晓平

合肥工业大学 计算机与信息学院

2025 年 10 月 9 日

目 录

第1章 课程设计预备知识

《操作系统》是计算机专业重要的专业基础课程之一，以操作系统的基本概念、技术、原理和重要算法为主要讲述内容。课程侧重于从理论的角度来介绍现代操作系统的核心技术内容，长篇论述较多，课程内容深，难度大。

操作系统课程设计是计算机与信息类专业的重要实践性教学环节。课程设计的主要任务是，在掌握程序的设计技能、专业基础课程和《操作系统》课程的理论知识的基础上，设计和实现操作系统的基本算法、模块与相关的资源管理功能，旨在加深对计算机硬件结构和系统软件的认识，初步掌握操作系统组成模块和应用接口的使用方法，提高进行工程设计和系统分析的能力，为毕业设计和以后的工程实践打下良好的基础。

本课程设计指导书内容分预备、设计题目安排两部分。预备介绍进入课程设计之前的预备知识和有关课程设计的总体概况，设计题目具体规定每个设计任务的内容和要求。

1.1 操作系统内核课程设计的代码规范要求

操作系统课程设计涉及大量的内核代码编写，原则上只允许使用系统级程序设计语言C语言或者汇编语言。主要的操作系统代码部分都使用C语言（不建议使用C++语言），部分设计关键性能和直接与硬件打交道的部分，建议使用汇编语言实现。

每个完成的课程设计包含两个代码组成部分：(1) 操作系统代码；(2) 演示代码；

1.1.1 操作系统代码

操作系统代码部分，完成课程设计的主体设计内容，需要根据操作系统内核原理来进行数据结构的设计和算法的设计与实现。

这一部分设计内容以一系列函数的形式表现出来，设计的最终结果，就是一系列的.C文件，其中定义和实现所有完成设计题目功能的代码。一般将这部分的设计组织成一个Project，Project的目标不是生成应用程序，而是生成一个.lib。

这部分设计强调按照操作系统原理课程的要求，设计和实现相关的数据结构。数据结构需要精心设计，要求紧凑、节省空间。算法的设计也以操作系统原理课程讲授的标准算法为基准。

允许并鼓励同学们设计自己的创新的数据结构和算法，但前提是需要给出实验数据和理论说明，以证明自己设计的新的数据结构的有效性。

这部分代码，没有UI，不带任何界面，没有针对标准I/O的操作。

内核代码设计规范，比如不允许使用标准STL、不允许出现输入输出语句如cin/cout，printf/scanf、窗口操作等，请同学们设计代码时，务必注意。代码的编写规范和风格，具体请同学们读一读Linux源码，学习下如何写。

1.1.2 演示代码

演示代码是展示和验证上述“1.1.1 操作系统代码”中所设计的代码的有效性而额外编写的演示程序。需要通过将上述函数有机地按一定顺序调用，以展示操作系统较大模块的功能和运行过程。

演示代码通常带有UI，可以用标准的Windows/Linux窗口作为UI，也可以使用字符命令作为展示窗口，要求功能展示完整，操作简便。

这部分代码，通常也是以一系列的.C/.CPP文件的形式出现，并组织成一个Project这个

Project的链接结果，是一个应用程序，装入模块。例如Windows环境下的.exe。

1.2 开发与运行环境

1.2.1 操作系统环境

本课程设计开始前必需准备好相应的操作系统软件。由于操作系统软件的特殊性，无法象其它软件一样随时卸载和重装，从操作系统的应用范围、性能、价格和可获得性等因素出发，本指导书选择流行的Windows作为实验操作系统。有条件的读者还可以选用UNIX/LINUX或其它主机系统作为设计环境。

1.2.2 程序开发环境：Microsoft Visual C/C++ 5.0

本书所涉及的课程设计都需要编程，建议使用C/C++作为编程语言，如果课程的设计环境为Windows操作系统，请选用Microsoft Visual C/C++ 5.0或更新的Visual Studio版本，不允许使用Java、Visual Basic等解释执行的语言。以下解释Microsoft Visual C/C++ 5.0开发环境的使用，其他更新的版本请参照对应参考书。

Microsoft C/C++是美国Microsoft公司推出的系列编程环境，是目前Windows平台上使用最为广泛的C/C++开发环境。Microsoft C/C++带有一个具有GUI界面的集成环境，操作非常方便。

Microsoft Visual C/C++支持多种类型程序的开发，读者可根据自己的实际情况选择。如果熟悉Windows编程技术，可以使用MFC型应用程序，否则可使用Win32 Console Application型程序。

Microsoft Visual C/C++的使用较复杂，详细的使用方法请参见相关书籍和联机帮助。下面仅简单介绍一个利用Microsoft Visual C/C++5.0开发Win32 Console Application应用程序的过程。

(1) Microsoft Visual C/C++ 5.0

双击图标：Microsoft Visual C/C++ 5.0

或 选择菜单：Microsoft Visual C/C++ 5.0

(2) 建立新项目 Exp1

选择菜单：File->New

选择页面标签：Project。

选择应用程序类型：Win32 Console Application

选择项目位置：C:\TEMP

输入应用程序名称：Exp1

点击OK按钮，VC系统自动产生一个新项目Exp1；

(3) 建立源程序 maic.c：

选择菜单：File->New

弹出对话框见上图。

选择页面标签：Files。

选择文件类型：C/C++ Source File

选择项目位置：C:\TEMP

输入文件名：main

注意选择Add To Project标签，在其前面添加“√”符号。

点击OK按钮，VC系统自动产生一个新文件main.c，并将其加入项目Exp1中；

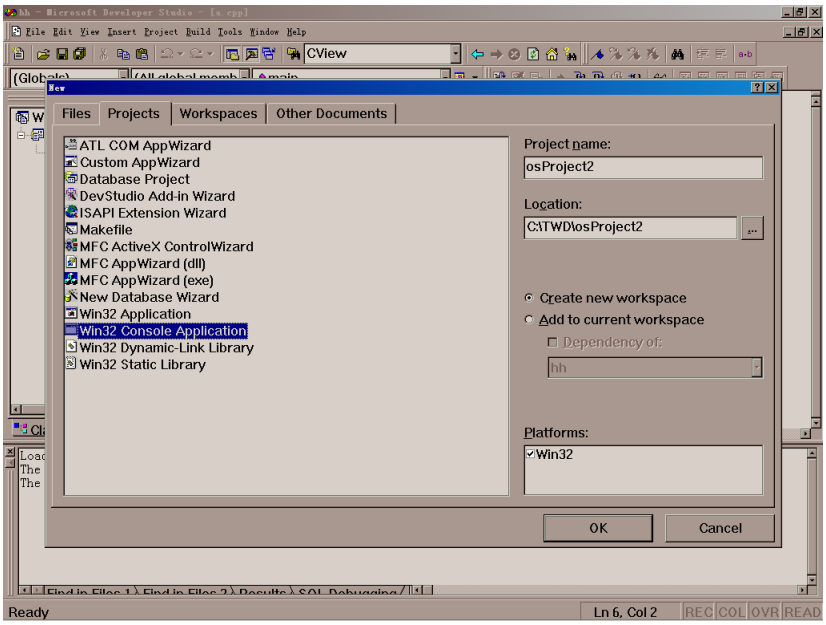

图 1-1 Microsoft C/C++开发环境

(4) 编辑源程序

在窗口main.c中，键入如下程序代码：

```
#include "stdio.h"
main()
{    int i, j ;
     printf( "输入数 i：") ;
      scanf("%d",&i);
     printf( "输入数 j：") ;
      scanf("%d",&j);
     printf( "输出结果为：%d,%d", i, j ) ;
}
```

选择菜单File->Save All存盘。

(5) 编译、链接

选择菜单Build->Rebuild All

强迫编译链接本项目所有的源程序。

(6) 运行

按Ctrl-F5或选择相应菜单运行应用程序。

(7) 调试

程序执行如果出现错误，可以进行调试，方法如下。

- 设置程序断点　移动光标到怀疑出错的位置，单击鼠标右键，从弹出的菜单上选择Insert/Remove Breakpoint。

- 进入调试状态　按F5执行程序，程序将在设置了断点的语句前停下来。
- 查看变量的值　移动鼠标到要查看的变量上方，略停一会，该变量的值会自动显示出来。
- 继续执行程序　按F10/F11/F5继续执行程序。

1.2.3 程序开发环境：Microsoft Visual Studio 2019

以下解释Microsoft Visual Studio 2019开发环境的使用，其他更新的版本请参照对应参考书。Microsoft Visual Studio是美国Microsoft公司推出的系列编程环境，是目前Windows平台上使用最为广泛的C/C++开发环境。Microsoft Visual Studio带有一个具有GUI界面的集成环境，操作非常方便。

Microsoft Visual Studio支持多种类型程序的开发，读者可根据自己的实际情况选择。如果熟悉Windows编程技术，可以使用MFC型应用程序，否则可使用Win32 Console Application型程序。

Microsoft Visual Studio的使用较复杂，详细的使用方法请参见相关书籍和联机帮助。下面仅简单介绍一个利用Microsoft Visual Studio开发Win32 Console Application应用程序的过程。

(1) Microsoft Visual Studio 2019

双击图标：Microsoft Visual Studio 2019

或 选择菜单：Microsoft Visual Studio 2019

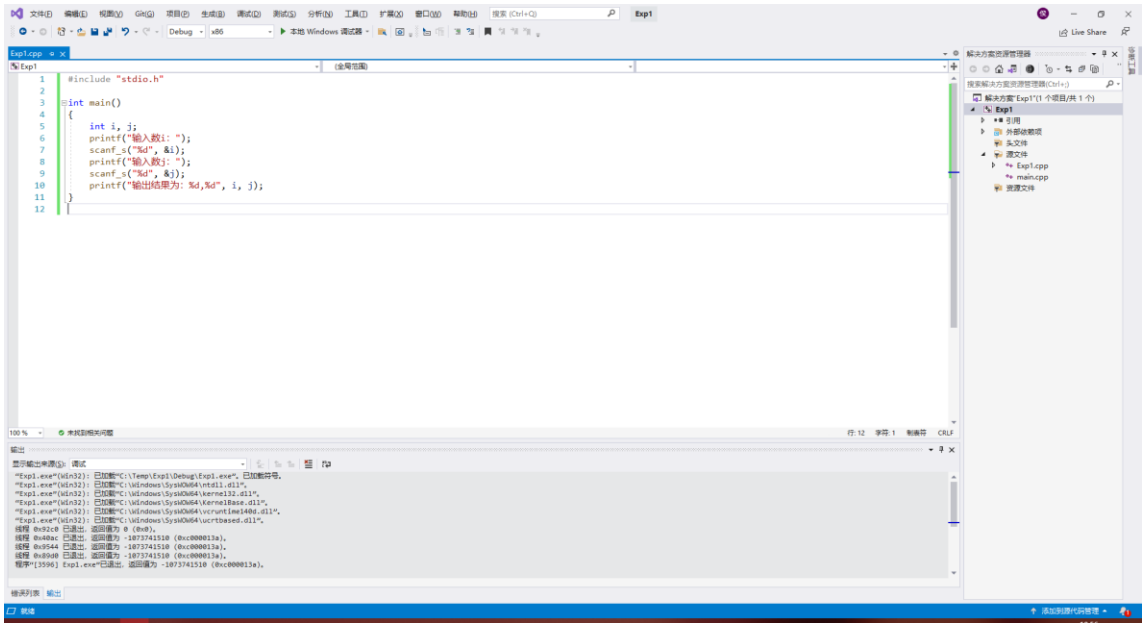

图 1-2 Microsoft Visual 2019 开发环境

(2) 建立新项目 Exp1

选择页面标签：创建新项目

弹出对话框见上图。

选择页面标签：语言->C++ 平台->Windows

选择项目模板：空项目 或 控制台应用

选择页面标签：下一步

选择项目位置：C:\TEMP

输入应用程序名称：Exp1

点击创建按钮，VS系统自动产生一个新项目Exp1；

图 1-3 Microsoft Visual 2019 创建新项目

(4) 编辑源程序 Exp1.cpp

在窗口Exp1.cpp中，键入如下程序代码：

```
#include "stdio.h"
int main()
{    int i, j ;
     printf( "输入数 i：") ;
      scanf_s("%d",&i);
     printf( "输入数 j：") ;
      scanf_s("%d",&j);
     printf( "输出结果为：%d,%d", i, j ) ;
}
```

选择菜单File->Save All存盘。

(5) 编译、链接

选择菜单Build->Rebuild All

强迫编译链接本项目所有的源程序。

(6) 运行

按Ctrl+F5或选择本地Windows调试器运行应用程序。

(7) 调试

程序执行如果出现错误，可以进行调试，方法如下。

- 设置程序断点　移动光标到怀疑出错的位置，单击鼠标右键，从弹出的菜单上选择Insert/Remove Breakpoint。
- 进入调试状态　按F5执行程序，程序将在设置了断点的语句前停下来。
- 查看变量的值　移动鼠标到要查看的变量上方，略停一会，该变量的值会自动显示出来。
- 继续执行程序　按F10/F11/F5继续执行程序。

1.3 课程设计报告

课程设计结束后，要求撰写课程设计报告。课程设计的基本内容包括：

(1) 课程设计任务

依据操作系统课程所介绍的固定分区分配的存储管理方案，按照内河代码的实现原则，设计和实现一个固定分区分配存储管理系统。

(2) 课程设计目的和要求

详细解释课程设计的要求。

(3) 开发环境

描述开发环境。

(4) 相关原理及算法

描述操作系统相关原理和算法等。

(5) 系统结构和主要的算法设计思路

描述所设计系统的系统架构，主要的算法思想、流程图等。

(6) 程序实现---主要数据结构

描述所设计系统的关键数据结构。

(7) 程序实现---程序实现细节

描述所设计系统的核心程序、关键函数的程序实现细节。

(8) 程序运行的主要界面和结果截图

列出系统运行的主要界面和运行结果截图。

(9) 附录

列出所设计系统的程序、函数的程序清单，以及附加的数据、图表、截图。

课程设计报告的封面样式见下图1-4。课程设计报告的具体内容和章节划分，参见图1-5。

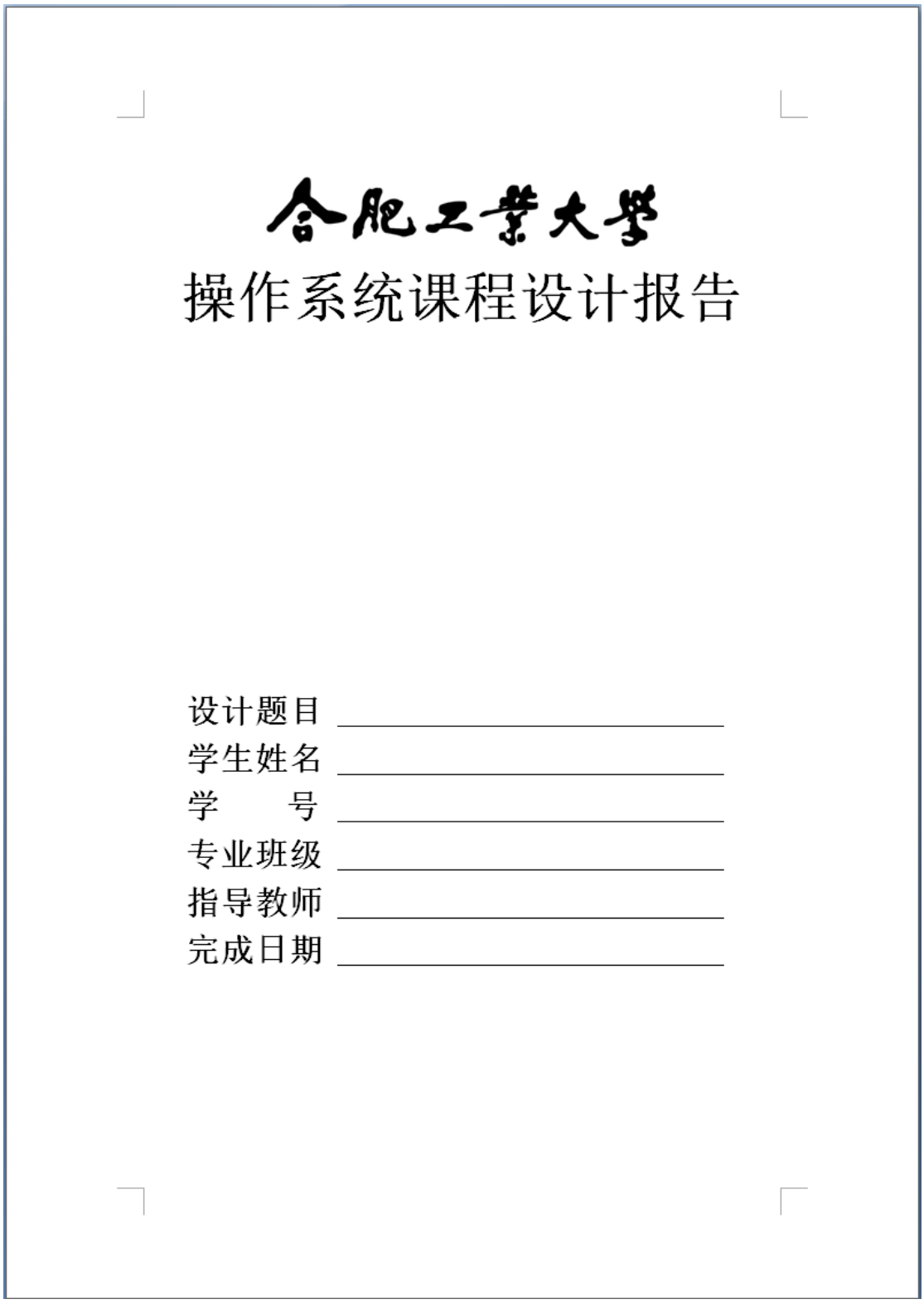

图 1-4　操作系统课程设计报告封面

1. 课程设计任务、要求、目的

1.1 课程设计任务

依据操作系统课程所介绍的固定分区分配的存储管理方案，按照内核代码的实现原则，设计和实现一个固定分区分配存储管理系统。

1.2 课程设计目的和要求

系统应该包含两个部分，一个部分是按内核代码原则设计的固定分区分配存储管理系统，由一系列的函数组成；另一个部分是演示系统，调用固定分区分配存储管理系统的相应函数，以让其运行，同时提供系统的展示界面，可以是 GUI 或者字符界面，以展示系统的运行状态，显示系统的关键数据结构的内容。

具体包括：建立描述内存分配状况的数据结构；建立描述进程的数据结构；使用两种方式产生进程：(a)自动产生，(b)手工输入；在屏幕上显示内存状况、进程执行情况；建立分区分配与回收算法，支持紧凑算法；时间的流逝可用下面几种方法模拟：(a)按键盘，每按一次可认为过一个时间单位；(b)响应 WM_TIMER；

2. 开发环境

使用什么开发环境。

3. 相关原理及算法

简要描述操作系统相关原理和算法等。

4. 系统结构和主要的算法设计思路

描述所设计系统的系统架构，主要的算法思想、流程图等。

5. 程序实现---主要数据结构

描述所设计系统的关键数据结构。

6. 程序实现---程序实现细节描述

描述所设计系统的核心程序、关键函数的程序实现细节。

7. 程序运行的主要界面和实验结果截图

列出系统运行的主要界面和运行结果截图。

8. 总结和感想体会

撰写总结和设计的感想。

参考文献

列出参考文献，格式参见国标 GB/T 7714－2005。

附录 1:程序清单（部分）

列举程序清单，由于一般程序量很大，这里只须列举部分核心代码即可。

附录 2:

其余需要附上的数据、图表、截图等。可以没有。

图 1-5　操作系统课程设计具体内容和结构

第2章 课程设计题目

2.1 针对Windows平台的题目

2.1.1 进程的描述

1. 生成和绘制简单程序片段的前驱图。（1 人，难度：4）
   - 建立前驱图的数据结构描述；
   - 可以从键盘或对话框接收程序片段；
   - 可将程序片段存入磁盘文件或从文件中取出；
   - 对程序片段进行词法分析，得出各语句之间的依赖关系，并据此生成前驱图；
   - 显示生成的前驱图；
   - 提供前驱图的修改功能，包括边和结点的增加和删除。
   - 可将前驱图存入磁盘文件或从文件中取出；

2.1.2 线程/进程技术

2. 多线程编程：临界区控制、线程互斥与同步。（1 人，难度：1）
   - 设置两个线程，一个执行计算 N:=N+1，另一个将 N 的值输出到窗口；
   - 为减慢线程的执行速度，可以在程序中插入类似 sleep(1000)的语句；
   - 在窗口上显示结果；
   - 设法调整两个线程的执行顺序，使之出现教材上所列出的（1）、（2）、（3）三种情况；
   - 设置互斥信号量，保证两线程互斥使用共享变量 N；
   - 设置同步信号量，保证两线程按指定顺序运行；

3. 多线程编程：生产者-消费者问题。（1 人，难度：1）
   - 设置两类线程，一类为生产者，一类为消费者；
   - 建立缓冲区的数据结构；
   - 使用菜单随机启动生产者或消费者；
   - 在窗口上显示缓冲区；
   - 随着线程每次操作缓冲区，更新窗口的显示；

4. 多线程编程：读者-写者问题。（1 人，难度：1）
   - 设置两类线程，一类为读者，一类为写者；
   - 使用菜单随机启动读者或写者；
   - 在窗口上显示读者或写者执行状态；
   - 随着线程的执行，更新窗口的显示；

5. 多线程编程：哲学家问题。（1 人，难度：1）
   - 设置线程，描述哲学家 ；
   - 使用菜单随机启动哲学家 ；
   - 在窗口上显示线程执行状态；
   - 随着线程的执行，更新窗口的显示；
   - 编写正确的哲学家程序，设法延迟线程的执行，使之出现死锁；
   - 编写正确的哲学家程序，保证不出现死锁；

6. 用户级线程库。（2 人，难度：5）
   - 建立用户级线程库；
   - 至少包括线程的创建、撤销、状态转换等，以及线程切换；
   - 库写好后，构建一个程序，演示线程库的使用；

2.1.3 进程间通信IPC

7. 进程通信。（1 人，难度：2）
   - 编写两个进程，一进程负责发送字符 串，另一进程负责接收字符串；
   - 分别支持 Windows 的的共享存储区和消息通讯机制、管道和 Socket；
   - 发送进程应可以从窗口或键盘上输入字符串；
   - 接收进程应可将字符串显示在窗口上；

8. 进程通信-直接消息通信。（1 人，难度：3）
   - 编写支持直接消息通信的例程库
   - 编写测试程序，验证例程库能正常工作

9. 进程通信-间接消息通信。（1 人，难度：3）
   - 编写支持间接消息通信的例程库
   - 设计信箱的数据结构
   - 编写测试程序，验证例程库能正常工作

10. 进程通信-管道通信。（1 人，难度：3）
    - 编写支持管道通信的例程库
    - 包括管道文件的建立和撤销、互斥和共享地读写管道文件
    - 编写测试程序，验证例程库能正常工作

2.1.4 处理机调度与作业调度

11. 多道程序设计技术：建立进程/线程的调度模型。（1 人，难度：3）
    - 建立操作系统内核的多任务并发运行机制，针对进程/线程的调度；
    - 包括 CPU 现场设置、调度的实现（基于硬件中断机制）
    - 可以选取时间片轮转作为调度算法；
    - 使用两种方式产生作业/进程：（a）自动产生， （b）手工输入；
    - 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位； (b) 响应 WM_TIMER；
    - 将一批进程/线程的执行情况存入磁盘文件，以后可以读出并重放；

12. 进程/作业调度算法。（1 人，难度：4）
    - 建立作业的数据结构描述；在屏幕上显示每个作业/进程的执行情况；
    - 使用两种方式产生作业/进程：（a）自动产生， （b）手工输入；
    - 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位； (b) 响应 WM_TIMER；
    - 计算并显示一批作业/进程的周转时间、平均周转时间、带权周转时间、平均带权周转时间。
    - 将一批作业/进程的执行情况存入磁盘文件，以后可以读出并重放；
    - 支持的调度算法：先来先服务、短作业/进程优先、时间片轮转调度算法、优先权调度算法、高响应比优先调度算法、多级反馈队列调度算法。

13. 实现高中低三级调度系统模型的系统内核模块。（1 人，难度：5）
- 建立操作系统内核中的作业调度、进程调度、中级调度的三级队列调度模型；
- 使用两种方式产生作业/进程：（a）自动产生， （b）手工输入；
- 构建作业控制块和进程控制块等核心数据结构；
- 计算并显示一批作业，从创建作业、创建进程，到作业个进程在诸队列上流转的完整过程。
- 将一批作业/进程的执行情况存入磁盘文件，以后可以读出并重放；
- 每类调度需要实现一种调度算法。
- 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位； (b) 响应 WM_TIMER；
- 提供信息转储功能，可将信息存入磁盘，也可从磁盘读入；

2.1.5 死锁的理论与算法

14. 进程死锁的检测：绘制资源分配图。（1 人，难度：3）
- 建立资源分配图的数据结构描述；
- 建立绘制资源分配图的例程，包括结点和有向边；
- 可以删除、添加结点或有向边；
- 可用鼠标在窗口的任意位置指点，确定结点或有向边位置；
- 可以拖动现有结点的位置，与该结点相连的有向边也随之移动；
- 可以将资源分配图存入文件，从文件中取出；

15. 进程死锁的检测：资源分配图化简判断是否有死锁发生。（1 人，难度：3）
- 建立所需数据结构；
- 使用题目 10 存成的资源分配图的文件作为输入（测试时可以使用自定义的文件格式）；
- 编写资源分配图化简算法；
- 每化简一步，在屏幕上显示化简的当前结果；
- 最后给出结论，是否死锁，如思索给出死锁的进程及资源；
- 提供信息转储功能，可将信息存入磁盘，也可从磁盘读入；

16. 进程死锁的避免：银行家算法。（1 人，难度：3）
- 建立银行家算法的数据结构描述；
- 将初始数据放在文件中，算法运行时读出；
- 对给定的资源请求，使用算法判断是否允许；
- 输出每次判断产生的执行序列；
- 注意算法的高效率实现
- 提供信息转储功能，可将信息存入磁盘，也可从磁盘读入；

17. 进程死锁的解除：演示教材展示两种撤销进程解除死锁方法。（1 人，难度：3）
- 建立所需数据结构；
- 设计一个复杂的死锁情况，用前述资源分配图文件表示，并进行展示。
- 分别使用两种方法（宽度、广度）找到应该逐步撤销的 n 个进程，显示搜索和比较过程
- 给出最后结论，设计 2-3 个复杂的死锁情况用于测试算法的正确性。
- 提供信息转储功能，可将信息存入磁盘，也可从磁盘读入；

2.1.6 基本存储管理技术

18. 固定分区分配存储管理。（1 人，难度：3）
- 建立描述内存分配状况的数据结构；
- 建立描述进程的数据结构；
- 使用两种方式产生进程：（a）自动产生，（b）手工输入；
- 在屏幕上显示内存的分配状况、每个进程的执行情况；
- 建立分区的分配与回收算法；
- 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位；(b) 响应 WM_TIMER；
- 提供信息转储功能，可将信息存入磁盘，也可从磁盘读入；

19. 动态分区分配存储管理。（1 人，难度：3）
- 建立描述内存分配状况的数据结构；
- 建立描述进程的数据结构；
- 使用两种方式产生进程：（a）自动产生，（b）手工输入；
- 在屏幕上显示内存的分配状况、每个进程的执行情况；
- 建立分区的分配与回收算法，支持紧凑算法；
- 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位；(b) 响应 WM_TIMER；
- 将一批进程的执行情况存入磁盘文件，以后可以读出并重放；
- 支持算法：首次适应算法、循环首次适应算法、最佳适应算法：最坏适应算法。

20. 分段存储管理系统：建立一个基本分段存储管理系统模型。（1 人，难度：3）
- 首先分配一片较大的内存空间，作为程序运行的可用存储空间；
- 建立应用程序的模型，应该包括相应的分段描述与存储结构；建立进程的基本数据结构及相应算法；建立管理存储空间的基本存储结构。
- 建立管理分段的基本数据结构与算法；设计存储空间的分配与回收算法；
- 提供信息转储功能，可将存储信息存入磁盘，也可从磁盘读入；

21. 分页存储管理系统：建立一个基本分页存储管理系统模型。（1 人，难度：3）
- 首先分配一片较大的内存空间，作为程序运行的可用存储空间；
- 建立应用程序的模型；
- 建立进程的基本数据结构及相应算法
- 建立管理存储空间的基本存储结构。
- 建立管理分页的基本数据结构与算法。
- 设计存储空间的分配与回收算法；
- 提供信息转储功能，可将存储信息存入磁盘，也可从磁盘读入；

22. 段页式存储管理系统：建一个基本段页存储管理系统模型。（1 人，难度：4）
- 首先分配一片较大的内存空间，作为程序运行的可用存储空间；
- 建立应用程序的模型，包括分段结构在内；
- 建立进程的基本数据结构及相应算法
- 建立管理存储空间的基本存储结构。
- 建立管理段页的基本数据结构与算法。
- 设计存储空间的分配与回收算法；
- 提供信息转储功能，可将存储信息存入磁盘，也可从磁盘读入；

2.1.7 虚拟存储管理技术

23. 页面置换算法演示。（1 人，难度：3）
- 建立相应的数据结构；
- 在屏幕上显示页面的状况；
- 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位； (b) 响应 WM_TIMER；
- 将一批页的置换情况存入磁盘文件，以后可以读出并重放；
- 计算页面的缺页次数、缺页后的页面置换次数；
- 支持算法：FIFO、LRU、最佳置换算法、CLOCK。

24. 页式虚拟存储器：内存分配策略。（1 人，难度：4）
- 首先分配一片较大的内存空间，作为程序运行的可用存储空间（页框空间）；
- 建立应用程序的模型，包括分页结构在内；
- 建立进程的基本数据结构及相应算法
- 建立管理请求页式存储管理的基本存储结构。
- 设计给进程分配页框的基本算法，以及页面分配策略，包括固定分配局部置换、可变分配全局置换、可变分配局部置换等；
- 可以展示进程运行过程中的缺页率情况；
- 提供信息转储功能，可将存储信息存入磁盘，也可从磁盘读入；

25. 工作集理论：计算工作集。（1 人，难度：3）
- 建立相应的进程模型，关键是局部区页面划分
- 设置工作集模型和相关数据结构；
- 在屏幕上显示页面的状况；
- 将一批页面的引用情况存入磁盘文件，以后可以读出并重放；
- 计算多进程并发执行是的工作集变化情况；

26. 段式虚拟存储管理系统：建一个请求分段存储管理系统模型。（2 人，难度：5）
- 首先分配一片较大的内存空间和一段磁盘空间，作为程序运行的可用存储空间和外存兑换区；
- 建立应用程序的模型，包括分段结构在内；
- 建立进程的基本数据结构及相应算法
- 建立管理存储空间的基本存储结构。
- 建立管理段的基本数据结构与算法。
- 设计存储空间的分配与回收算法；
- 实现缺段中段支持的逻辑地址到物理地址转换，实现虚拟存储器；
- 提供信息转储功能，可将存储信息存入磁盘，也可从磁盘读入；

27. 页式虚拟存储管理系统：建立一个请求分页存储管理系统模型。（2 人,难度：5）
- 首先分配一片较大的内存空间和一段磁盘空间，作为程序运行的可用存储空间和外存兑换区；
- 建立应用程序的模型，包括分页结构在内；
- 建立进程的基本数据结构及相应算法
- 建立管理存储空间的基本存储结构。
- 建立管理页的基本数据结构与算法。
- 设计存储空间的分配与回收算法；
- 实现缺页中断支持的逻辑地址到物理地址转换，实现虚拟存储器；
- 提供信息转储功能，可将存储信息存入磁盘，也可从磁盘读入；

28. 段页式虚拟存储管理系统：建立一个段页式虚拟存储管理系统模型。（2 人,难度：5）

- 首先分配一片较大的内存空间和一段磁盘空间，作为程序运行的可用存储空间和外存兑换区；
- 建立应用程序的模型，包括分段结构在内；
- 建立进程的基本数据结构及相应算法
- 建立管理存储空间的基本存储结构。
- 建立管理段页的基本数据结构与算法。
- 设计存储空间的分配与回收算法；
- 实现缺页中段支持的逻辑地址到物理地址转换，实现虚拟存储器；
- 提供信息转储功能，可将存储信息存入磁盘，也可从磁盘读入；

2.1.8 I/O设备管理

29. 设备独立性原理。（1 人，难度：5）
    - 构建设备独立性例程库
    - 建立用户模型、设备模型和进程模型；
    - 实现逻辑设备表；
    - 提供逻辑设备和物理设备
    - 编程时使用逻辑设备，运行时始有物理设备；
    - 编写测试程序，验证例程库能正常工作；

30. SPOOLING 技术例程库。（1 人，难度：5）
    - 构建支持 SPOOLING 的例程库
    - 建立一个可以替代输入机或输出机的输入进程/线程或者输出进程/线程；
    - 设置一个磁盘文件作为输入井或输出井；
    - 输入进程/线程或者输出进程/线程从键盘接受输入到输入井或者从输出井输出到屏幕
    - 其他进程/线程不直接操作键盘和屏幕，而是从输入井读取或者写到输出井；
    - 设计至少两个进程/线程执行输入或者输出任务验证方法；
    - 注意进程/线程间的互斥。

2.1.9 磁盘存储管理技术

31. 磁盘调度算法。（1 人，难度：3）
    - 建立相应的数据结构；
    - 在屏幕上显示磁盘请求的服务状况；
    - 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位； (b) 响应 WM_TIMER；
    - 将一批磁盘请求的情况存磁盘文件，以后可以读出并重放；
    - 计算磁头移动的总距离及平均移动距离；
    - 支持算法：FIFO、SSTF、SCAN、CSCAN、FSCAN。

32. 面向磁盘文件系统的磁盘高速缓存系统。（1 人，难度：5）
    - 为磁盘文件的读写建立一个基于磁盘块的缓冲机制，采用缓冲池机制，分为空闲队列、输出队列、输出队列、管理数据结构等部分；
    - 建立相关数据结构，设计相关缓冲区申请、释放等算法；
    - 实现延迟写、提前读等机制；
    - 使用 LRU 等缓冲置换算法；
    - 建立一批进程、文件，以及进程对文件的读写时机、记录；
    - 使用两种方式产生进程和文件：（a）自动产生， （b）手工输入；
    - 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位； (b) 响应 WM_TIMER；

- 显示每次磁盘告诉缓冲池的随时状态和相关数据结构的状态；

33. 空闲磁盘存储空间的管理：建立磁盘空间管理模块。（1 人，难度：3）
    - 建立相应的数据结构；
    - 磁盘上建立一个文件，文件长度设为 10MB，用该文件来模拟一个磁盘，磁盘的物理块大小为 512 字节。
    - 建立进程的数据结构；
    - 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位； (b) 响应 WM_TIMER；
    - 将一批进程对磁盘的请求的情况存磁盘文件，以后可以读出并重放；
    - 使用两种方式产生进程对磁盘的请求：（a）自动产生， （b）手工输入；
    - 显示每次磁盘的请求和空间释放后的相关数据结构的状态；
    - 显示每次磁盘的请求和空间释放后状态；
    - 支持的管理方法：空闲表法、空闲链表法、位示图法。

34. 空闲磁盘存储空间的管理：成组链接法。（1 人，难度：4）
    - 建立相应的数据结构；
    - 磁盘上建立一个文件，文件长度设为 10MB，用该文件来模拟一个磁盘，磁盘的物理块大小为 512 字节。
    - 建立进程的数据结构；
    - 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位； (b) 响应 WM_TIMER；
    - 将一批进程对磁盘的请求的情况存磁盘文件，以后可以读出并重放；
    - 使用两种方式产生进程对磁盘的请求：（a）自动产生， （b）手工输入；
    - 显示每次磁盘的请求和空间释放后的相关数据结构的状态；
    - 显示每次磁盘的请求和空间释放后状态；
    - 支持的管理方法：UNIX 成组链接法。

2.1.10 文件系统技术

35. 逻辑文件到物理文件映射：目录查询，查找 Windows FAT 文件系统指定目录下文件。（1 人，难度：1）
    - 对形如“C:\\Windows\\System\\Telnet.hlp”的文件进行查找；算法为：
    - 首先找到根目录 C:\\；
    - 从根目录找到子目录 Windows；
    - 再找到子目录 System；
    - 再找到文件 Telnet.hlp；
    - 需要在 Windows API 或者系统调用一级实现

36. 内存文件系统：建立基于内存的文件系统。（1 人，难度：5）
    - 首先分配一定容量的内存，建立虚拟磁盘；
    - 在该磁盘上建立相应的文件系统；
    - 解决文件的重名、共享和安全控制；
    - 支持文件的“按名存取”；
    - 为该文件系统设计相应的数据结构来管理目录、虚拟磁盘的空闲空间、已分配空间等。
    - 提供文件的创建、删除、移位、改名等功能；
    - 提供良好的界面，可以显示磁盘文件的状态和空间的使用情况；
    - 提供虚拟磁盘转储功能，可将信息存入磁盘，还可从磁盘读入内存；

37. 磁盘文件系统：建立基于磁盘存储设备的隐式链接文件系统。（1 人，难度：5）
- 首先分配一定容量的磁盘存储空间，作为文件存储空间；
- 建立相应的文件系统，使用隐式链接物理文件的系统；
- 解决文件的重名、共享和安全控制；
- 支持文件的“按名存取”；
- 为该文件系统设计相应的数据结构来管理目录、磁盘空闲空间、已分配空间等。
- 提供文件的创建、删除、移位、改名等功能。
- 提供良好的界面，可以显示磁盘文件系统的状态和空间的使用情况；
- 提供虚拟磁盘转储功能，可将信息存入磁盘，还可从磁盘读入内存；

38. 磁盘文件系统：建立基于磁盘存储设备的 FAT 文件系统。（1 人，难度：5）
- 首先分配一定容量的磁盘存储空间，作为文件存储空间；
- 建立相应的文件系统，使用 FAT 文件系统；
- 解决文件的重名、共享和安全控制；
- 支持文件的“按名存取”；
- 为该文件系统设计相应的数据结构来管理目录、磁盘空闲空间、已分配空间等。
- 提供文件的创建、删除、移位、改名等功能。
- 提供良好的界面，可以显示磁盘文件系统的状态和空间的使用情况；
- 提供虚拟磁盘转储功能，可将信息存入磁盘，还可从磁盘读入内存；

39. 磁盘文件系统：建立基于磁盘存储设备的 UNIX 文件系统。（1 人，难度：5）
- 首先分配一定容量的磁盘存储空间，作为文件存储空间；
- 建立相应的文件系统，使用 UNIX 文件系统，使用索引结点、混合分配方式、成组连接方法管理存储空间；
- 解决文件的重名、共享和安全控制；
- 支持文件的“按名存取”；
- 为该文件系统设计相应的数据结构来管理目录、磁盘空闲空间、已分配空间等。
- 提供文件的创建、删除、移位、改名等功能。
- 提供良好的界面，可以显示磁盘文件系统的状态和空间的使用情况；
- 提供虚拟磁盘转储功能，可将信息存入磁盘，还可从磁盘读入内存；

40. U 盘文件系统：建立基于 U 盘存储设备的 FAT32 文件系统。（1 人，难度：5）
- 首先分配一定容量的 U 盘存储空间，作为文件存储空间；
- 支持长文件名
- 解决文件的重名、共享和安全控制；
- 支持文件的“按名存取”；
- 建立相应的文件系统，使用 FAT32 文件系统；
- 为该文件系统设计相应的数据结构来管理目录、磁盘空闲空间、已分配空间等。
- 提供文件的创建、删除、移位、改名等功能。
- 提供良好的界面，可以显示 U 盘文件系统的状态和空间的使用情况；
- 提供虚拟磁盘转储功能，可将信息存入磁盘，还可从磁盘读入内存；

41. U 盘文件系统格式化命令：建立 U 盘文件系统。（1 人，难度：5）
- 实现针对 U 盘的格式化命令；
- 实现在磁盘上建立标准的 FAT32 文件系统。

42. 事务模型例程库：建立事务的系统安全管理模型。（1 人，难度：5）
- 建立支持事务机制的用户例程库；
- 建立事务的基本数据结构，包括事务日志、事物记录；
- 设计基于事务日志的 Redo 和 Undo 功能。

- 建立一个多线程的测试程序，分别对磁盘上的多个文件进行处理，并利用事务来管理；执行过程中人为安排一些意外的发生，包括断电、异常等，导致文件修改任务无法完成；利用事务日志进行数据的恢复；

43. 访问控制技术：建立基于访问控制表的系统安全管理。（1 人，难度：4）
    - 建立以用户组为管理域的访问控制表基本数据结构；
    - 模拟操作系统中的若干对象，并可添加到访问控制表中；
    - 建立操作系统的用户模型；
    - 模拟用户登录及切换，访问对象时的受限情况。

44. 访问控制技术：建立基于访问权限表的系统安全管理。（1-2 人，难度：4）
    - 实现 UNIX/LINUX 的文件访问权限模型；
    - 模拟文件及目录的数据结构，为文件添加访问权限
    - 建立用户与用户组模型
    - 设计读、写和执行三种指令，模拟不同用户访问文件或目录时所发生的受限情况。

2.1.11 操作系统接口：Unix/Linux

45. 操作系统接口：兼容 Unix/Linux 命令接口 1。（1 人，难度：4）
    - 为 Windows 操作系统建立一个兼容 Unix/Linux 命令的命令接口；
    - 实现关于文件系统的操作命令，包括 ls,cat,cp,rm,mv,mkdir,rmdir,cd,find 等，命令的内容与详细格式请查阅 Unix/Linux 命令手册；
    - 可以字符形式接收命令，执行命令，然后显示命令执行结果；

46. 操作系统接口：兼容 Unix/Linux 命令接口 2。（1 人，难度：4）
    - 为 Windows 操作系统建立一个兼容 Unix/Linux 命令的命令接口；
    - 实现杂项命令包括 login,password,logout,sort,more,print,>,>>,<,<<,|等，命令的内容与详细格式请查阅 Unix/Linux 命令手册；
    - 可以字符形式接收命令，执行命令，然后显示命令执行结果；

47. 操作系统接口：兼容 Unix/Linux 命令接口 3。（1 人，难度：5）
    - 为 Windows 操作系统建立一个兼容 Unix/Linux 的 shell 文件执行命令接口；
    - 核心是实现.sh 格式文件的解读和执行
    - 应该包括常用的 shell 文件的流程控制语句，如顺序、循环和分支转移等，命令的内容与详细格式请查阅 Unix/Linux 命令手册；

2.1.12 操作系统接口:openEuler

48. 操作系统接口：兼容 openEuler 命令接口 1。（1 人，难度：4）
    - 为 Windows 操作系统建立一个兼容 openEuler 命令的命令接口；
    - 实现关于文件系统的操作命令，包括 ls,cat,cp,rm,mv,mkdir,rmdir,cd,find 等，命令的内容与详细格式请查阅 openEuler 命令手册；
    - 可以字符形式接收命令，执行命令，然后显示命令执行结果；

49. 操作系统接口：兼容 openEuler 命令接口 2。（1 人，难度：5）
    - 为 Windows 操作系统建立一个兼容 openEuler 命令的命令接口；
    - 实现杂项命令包括 login,password,logout,sort,more,print,>,>>,<,<<,|等，命令的内容与详细格式请查阅 openEuler 命令手册；
    - 可以字符形式接收命令，执行命令，然后显示命令执行结果；

50. 操作系统接口：兼容 openEuler 命令接口 3。（1 人，难度：5）
- 为 Windows 操作系统建立一个兼容 openEuler 的 shell 文件执行命令接口；
- 核心是实现.sh 格式文件的解读和执行
- 应该包括常用的 shell 文件的流程控制语句，如顺序、循环和分支转移等，命令的内容与详细格式请查阅 Unix/Linux 命令手册；

2.2 针对Unix/Linux平台的题目

2.2.1 进程间通信IPC

51. 进程通信。（1 人，难度：3）
- 编写两个进程，一进程负责发送字符串，另一进程负责接收字符串；
- 分别支持 Unix/Linux 的共享存储区、消息队列、管道和 Socket；
- 发送进程应可以从窗口或键盘上输入字符串；
- 接收进程应可将字符串显示在窗口上；

2.2.2 操作系统接口

52. 操作系统接口：Windows 命令接口 1。（1 人，难度：4）
- 为 Unix/Linux 操作系统建立兼容的 Windows/DOS 命令接口，杂项命令；
- 具体命令：CLS, DATE,TIME,DOSKEY,FIND,FINDSTR,COMP,FC,EXIT,HELP, MORE，命令格式可参照 Windows 的 CMD.EXE 或 MS-DOS 提供的命令；
- 设计命令的名称、参数等格式。
- 可以字符形式接收命令，执行命令，然后显示命令执行结果；

53. 操作系统接口：Windows 命令接口 2。（1 人，难度：4）
- 为 Unix/Linux 操作系统建立兼容的 Windows/DOS 命令接口，文件与目录命令；
- 具体命令：DIR, RD,CD,MD, DEL,MOVE,REN,XCOPY, PROMPT,SORT,TYPE,COPY，命令格式可参照 Windows 的 CMD.EXE 或 MS-DOS 提供的命令；
- 设计命令的名称、参数等格式。
- 可以字符形式接收命令，执行命令，然后显示命令执行结果；

54. 操作系统接口：Windows 命令接口 3。（1 人，难度：5）
- 为 Unix/Linux 操作系统建立兼容的 Windows/DOS 命令接口，批处理文件中的流程命令；
- 批处理文件的扩展名为.BATCH
- 具体命令：IF, FOR,WHILE,SET,SHIFT,ECHO, GOTO,:（标号），命令格式可参照 Windows 的 CMD.EXE 或 MS-DOS 提供的命令；
- 设计命令的名称、参数等格式。
- 可以执行扩展名为.BATCH 的批处理文件。

2.3 针对openEuler平台的题目

2.3.1 处理机调度与作业调度

55. CFS 调度算法。（1 人，难度：4）

- 建立进程的数据结构描述；在屏幕上显示每个进程的执行情况；
- 使用两种方式产生进程：（a）自动产生， （b）手工输入；
- 时间的流逝可用下面几种方法模拟：（a）按键盘，每按一次可认为过一个时间单位； (b) 响应时钟中断；
- 设计并实现完整的 CFS 调度算法；
- 计算并显示一批进程的周转时间、平均周转时间、带权周转时间、平均带权周转时间。
- 将一批进程的执行情况存入磁盘文件，以后可以读出并重放；
- 支持的调度算法：先来先服务、短作业/进程优先、时间片轮转调度算法、优先权调度算法、高响应比优先调度算法、多级反馈队列调度算法。

2.3.2 进程间通信IPC

56. 进程通信。（1 人，难度：3）
    - 编写两个进程，一进程负责发送字符串，另一进程负责接收字符串；
    - 分别支持 openEuler 的共享存储区、消息队列、管道和 Socket；
    - 发送进程应可以从窗口或键盘上输入字符串；
    - 接收进程应可将字符串显示在窗口上；

2.3.3 存储管理

57. 分页存储管理系统：建立一个 openEuler 基本分页存储管理系统模型。（1 人，难度：4）
    - 首先分配一片较大的内存空间，作为程序运行的可用存储空间；
    - 建立应用程序的模型；
    - 建立进程的基本数据结构及相应算法
    - 建立管理存储空间的基本存储结构,包括 3 级页表机制。
    - 设计存储空间的分配与回收算法；
    - 提供信息转储功能，可将存储信息存入磁盘，也可从磁盘读入；

58. 页式虚拟存储管理系统：建立一个 openEuler 请求分页存储管理系统的模型。（1 人，难度：5）
    - 首先分配一片较大的内存空间和一段磁盘空间,作为程序运行的可用存储空间和外存兑换区；
    - 建立应用程序的模型，包括分页结构在内；
    - 建立进程的基本数据结构及相应算法
    - 建立管理存储空间的基本存储结构,包括 3 级页表机制。
    - 建立管理页的基本数据结构与算法,。
    - 设计存储空间的分配与回收算法；
    - 实现缺页中断支持的逻辑地址到物理地址转换，实现虚拟存储器；
    - 提供信息转储功能，可将存储信息存入磁盘，也可从磁盘读入；

2.3.4 文件系统

59. 磁盘文件系统：建立基于磁盘存储设备的 openEuler Ext4 文件系统。（1 人，难度：5）
    - 首先分配一定容量的磁盘存储空间，作为文件存储空间；
    - 建立相应的 Ext4 文件系统，使用索引结点、混合分配方式、成组连接方法管理存储空间；
    - 解决文件的重名、共享和安全控制；

- 支持文件的“按名存取”；
- 为该文件系统设计相应的数据结构来管理目录、磁盘空闲空间、已分配空间等。
- 提供文件的创建、删除、移位、改名等功能。
- 提供良好的界面，可以显示磁盘文件系统的状态和空间的使用情况；

提供虚拟磁盘转储功能，可将信息存入磁盘，还可从磁盘读入内存；

2.3.5 操作系统接口

60. 操作系统接口：Windows 命令接口 1。（1 人，难度：3）
    - 为 openEuler 操作系统建立兼容的 Windows/DOS 命令接口，杂项命令；
    - 具体命令：CLS, DATE,TIME,DOSKEY,FIND,FINDSTR,COMP,FC,EXIT,HELP, MORE，命令格式可参照 Windows 的 CMD.EXE 或 MS-DOS 提供的命令；
    - 设计命令的名称、参数等格式。
    - 可以字符形式接收命令，执行命令，然后显示命令执行结果；

61. 操作系统接口：Windows 命令接口 2。（1 人，难度：4）
    - 为 openEuler 操作系统建立兼容的 Windows/DOS 命令接口，文件与目录命令；
    - 具体命令：DIR, RD,CD,MD, DEL,MOVE,REN,XCOPY, PROMPT,SORT,TYPE,COPY，命令格式可参照 Windows 的 CMD.EXE 或 MS-DOS 提供的命令；
    - 设计命令的名称、参数等格式。
    - 可以字符形式接收命令，执行命令，然后显示命令执行结果；

62. 操作系统接口：Windows 命令接口 3。（1 人，难度：5）
    - 为 openEuler 操作系统建立兼容的 Windows/DOS 命令接口，批处理文件中的流程命令；
    - 批处理文件的扩展名为.BATCH
    - 具体命令：IF, FOR,WHILE,SET,SHIFT,ECHO, GOTO,:（标号），命令格式可参照 Windows 的 CMD.EXE 或 MS-DOS 提供的命令；
    - 设计命令的名称、参数等格式。
    - 可以执行扩展名为.BATCH 的批处理文件。

第3章 课程设计实施和验收

3.1 课程设计实施方式、时间和地点

课程设计原则上需要在指定实验室完成，鼓励提前作好，到实验室验收。验收安排在专业实验室进行。请参加验收的同学，按指定时间到指定实验室点验收，一个小组的同学须全部到场，才能验收。课程设计程序运行正确，回答验收教师质询正确，验收才能通过。

多个班同学参加验收的，原则上同一个设计题目的各班的小组，须由同一教师验收。

验收成绩、课程设计报告成绩，合在一起形成最终成绩。

3.2 课程设计验收和评分

1、题目大多数为一人一题，存储管理系统和文件管理系统比较复杂，可以两个人完成，且分值较高。题目分数代表难度等级，完成基本功能要求的题目只能得到相应的等级分。各种难度题目对应的最终评分最高如下（特殊情况除外）：

难度 1—中等，难度 2—良好，难度 3，4，5—优秀。

2、题目都需要算例验证后才能进行验收，并且提前准备好算例向老师演示与讲解。

3、操作系统课程设计题目的设计目标在于让观看者更加清晰明确的了解操作系统理论方法的作用，因此在题目基本功能要求上设计出更有利于展示介绍的内容，比如一目了然的结果、使用图形界面展示、更先进的算法、更能说明方法的算例都可以使得分数等级+1。

4、如不能完全理解或者达到题目的基本要求，但也完成了部分功能，可以进行完整展示，则视情况不扣分或等级-1。

5、2 人一题的题目每个人在设计任务中都要有具体任务，验收的时候进行说明，根据情况具体评分。

6、题目目前的等级是根据以往呈现出结果的暂定方案，主要以设计难度和代码量定义，但也有可能新的理解和设计下出现与以往不同的结果，可能会根据具体情况进行少数调整。

7、如发现抄袭，或者对所述程序完全不能理解，按及格或不及格处理。

8、最后上交材料的整理是决定是否以验收成绩作为最后成绩的补充，如课程设计报告或材料整理糟糕，代码混乱，则成绩降档处理。

3.2 课程设计材料提交

完整课程设计材料如下：

(1) 课程设计报告。名称：学号-人名.DOCX；

(2) 以学号-人名为文件夹组织电子资料（1）课程设计报告（2）可执行文件（3）测试用例（4）源工程文件文件夹（5）说明.txt，说明文档是用来说明该项目开发环境，使用第三方工具情况，如何使用。名称：学号-人名.RAR；

上述资料电子版，分别提交到操作系统课程网站。