第一章 导论

操作系统是管理计算机硬件并提供应用程序运行环境的软件。

多道程序设计：允许多个作业同时位于内存，从而保证CPU总有一个作业可以执行。

分时系统是多道程序系统的扩展，它采用调度算法，以快速切换作业，好像每个作业同时执行。

第二章 操作系统结构

操作系统服务大致分为三大类：程序控制、状态请求、I/O请求。

系统调用比API更为注重细节且更加难用，大致可以分为六大类：进程控制、文件管理、设备管理、信息维护、通信、保护。

简单结构、分层法、微内核法；动态加载模块；混合方法。

第三章 进程

进程由文本段、寄存器、栈、数据段、堆构成。

```
新的 --允许--> 就绪 <--中断---------- 运行 -- 退出 --> 终止
               ||                    ||
               |---调度器调度-------> |
               |                     |
                <-I/O完-等待<-I/O等---
```

操作系统内的每个进程表示，采用进程控制块(PCB)表示：
进程状态、程序计数器、CPU寄存器、CPU调度信息、内存管理信息、记账信息、I/O状态信息。

操作系统内主要有两种队列：就绪队列和设备队列。两种调度程序：长期调度程序和短期调度程序。

允许并发执行有多个原因：信息共享、计算加速、模块化和方便。`init` 的 pid 永远是 1。父进程 pid>0，子进程 pid=0。

```
            |---------------wait()------>父进程重启
pid=fork()--|                 |
            |----exec()---->exit()
```

进程间通信（IPC）有两种基本模型：共享内存、消息传递。

客户机/服务器系统三种策略：直接{远程程序调用（RPC）、本地程序调用（LPC）} 间接{套接字（TCP和UDP）、管道：`fd[0]`读出，`fd[1]`写入}。

第四章 多线程编程

线程是进程内的控制流，优点：用户相应的改进、进程内资源的共享、经济和可扩展性因素。

用户线程对程序员来说是可见的，而对内核来说是未知的。

第五章 进程调度

FCFS、SJF[SRTF]、Priority、RR、多级队列调度、CFS

实时系统：单调速率、最单截止期限优先

第六章 同步

互斥、进步、有限等待。

Peterson

```
do {
    flag[i] = true;
    turn = j;
    while(flag[j] && turn == j) ;
        // Critical
    flag[i] = false;
        // Remaining
} while (true);
```

互斥锁

```
wait(S){
    while(S<=0)
        ;   // busy wait
    S--;
}

signal(S){
    S++;
}
```

生产者消费者问题

```
semaphore mutex = 1;
semaphore empty = n;
semaphore full = 0;

do {
    // produce
    wait(empty);
    wait(mutex);
    // add
    signal(mutex);
    signal(full);
}

do {
    wait(full);
    wait(mutex);
    // remove
    signal(mutex);
    signal(empty);
    // consume
}
```

读者作者问题

```
semaphore rw_mutex = 1;
semaphore mutex = 1;
```

```
int read_count = 0;

do {
    wait(reader);
    wait(writer);
    // write
    signal(writer);
    signal(reader);
} while (true);

do {
    wait(reader);
    wait(mutex);
    read_count++;
    if(read_count==1)
        wait(writer);
    signal(mutex);
    signal(reader);
    // read
    wait(mutex);
    read_count--;
    if(read_count==0)
        signal(writer);
    signal(mutex);
}
```

哲学家就餐问题 管程

```
monitor DiningPhilosophers{
    enum {THINKING, HUNGRY, DININING} state[5];
    condition self[5];

    void pickup(int i){
        state[i] = HUNGRY;
        test(i);
        if(state[i]!=EATING)
            self[i].wait();
    }

    void putdown(int i){
        state[i] = THINKING;
        test((i+4)%5);
        test((i+1)%5);
    }

    void test(int i){
        if((state[(i+4)%5]!=EATING) &&
            (state[i]==HUNGRY) &&
            (state[(i+1)%5]!=EATING)){
                state[i] = EATING;
                self[i].signal();
        }
    }

    initialization_code(){
        for(int i = 0; i<5; ++i)
            state[i] = THINKING;
```

```
    }
}
```

第七章 死锁

必要条件：互斥、占有并等待、非抢占、循环等待。

银行家算法

第八章 内存管理策略

连续分配、分页、分段及分页分段组合

硬件支持、性能、碎片、重定位、交换、共享、保护。

第九章 虚拟内存管理

页面置换：FIFO、OPT、LRU

伙伴系统、slab分配

第十章 文件系统

文件是逻辑记录的一个序列。

连续分配、链接分配、索引分配

号码（每个字的位数）*（值为0的字数）+第一个值为1的位的偏移

传输速率，定位时间，旋转延迟

FCFS、SSTF、SCAN、C-SCAN、（LOOK,C-LOOK）