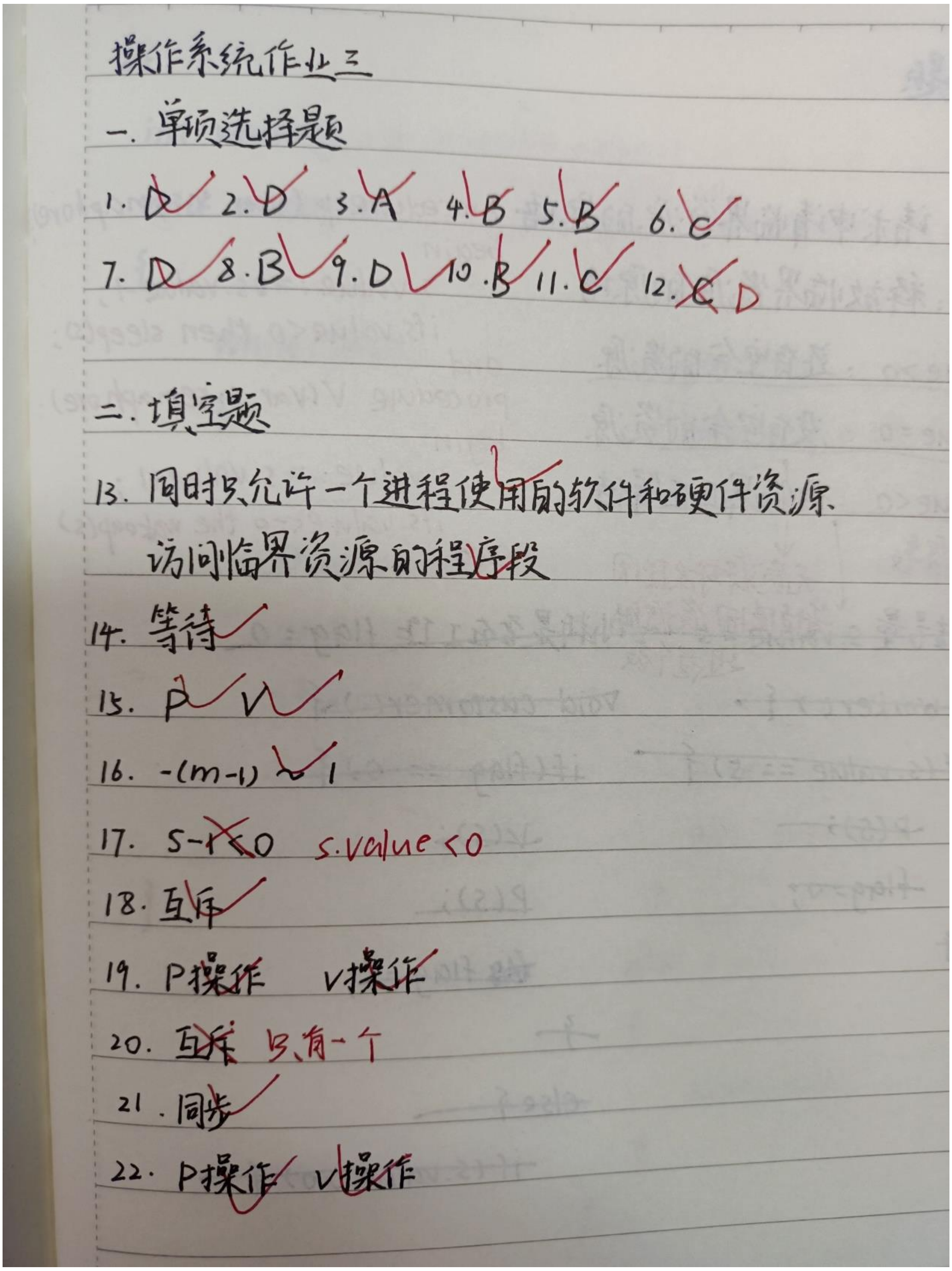


操作系统作业三

一. 单项选择题

1. D ✓ 2. D ✓ 3. A ✓ 4. B ✓ 5. B ✓ 6. C ✓

7. D ✓ 8. B ✓ 9. D ✓ 10. B ✓ 11. C ✓ 12. ~~C~~ D

二. 填空题

13. 同时只允许一个进程使用的软件和硬件资源 ✓
访问临界资源的程序段 ✓

14. 等待 ✓

15. P ✓ V ✓

16. -(m-1) ~ 1 ✓

17. ~~S-1<0~~ s.value<0

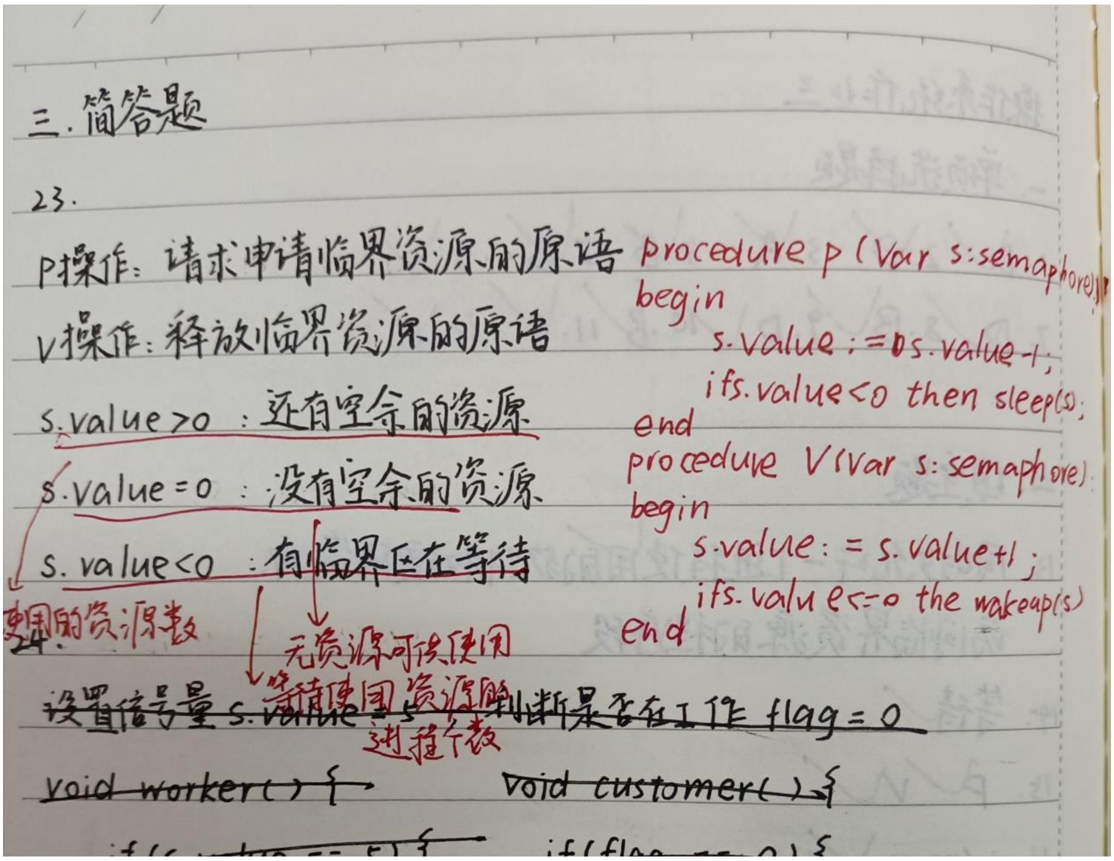
18. 互斥 ✓

19. P操作 V操作 ✓

20. ~~互斥~~ 只有一个

21. 同步 ✓

22. P操作 ✓ V操作 ✓



三. 简答题

23.

P操作：请求申请临界资源的原语

V操作：释放临界资源的原语

s.value>0 ：还有空余的资源

s.value=0 ：没有空余的资源

s.value<0 ：有临界区在等待

使用的资源数

无资源可供使用

等待使用资源的进程个数

```
procedure p (Var s:semaphore);
begin
    s.value := s.value -1;
    if s.value<0 then sleep(s);
end
procedure V(Var s: semaphore)
begin
    s.value: = s.value+1 ;
    if s.value<=0 the wakeup(s)
end
```

24.

~~设置信号量 s.value = 5 判断是否在工作 flag = 0~~

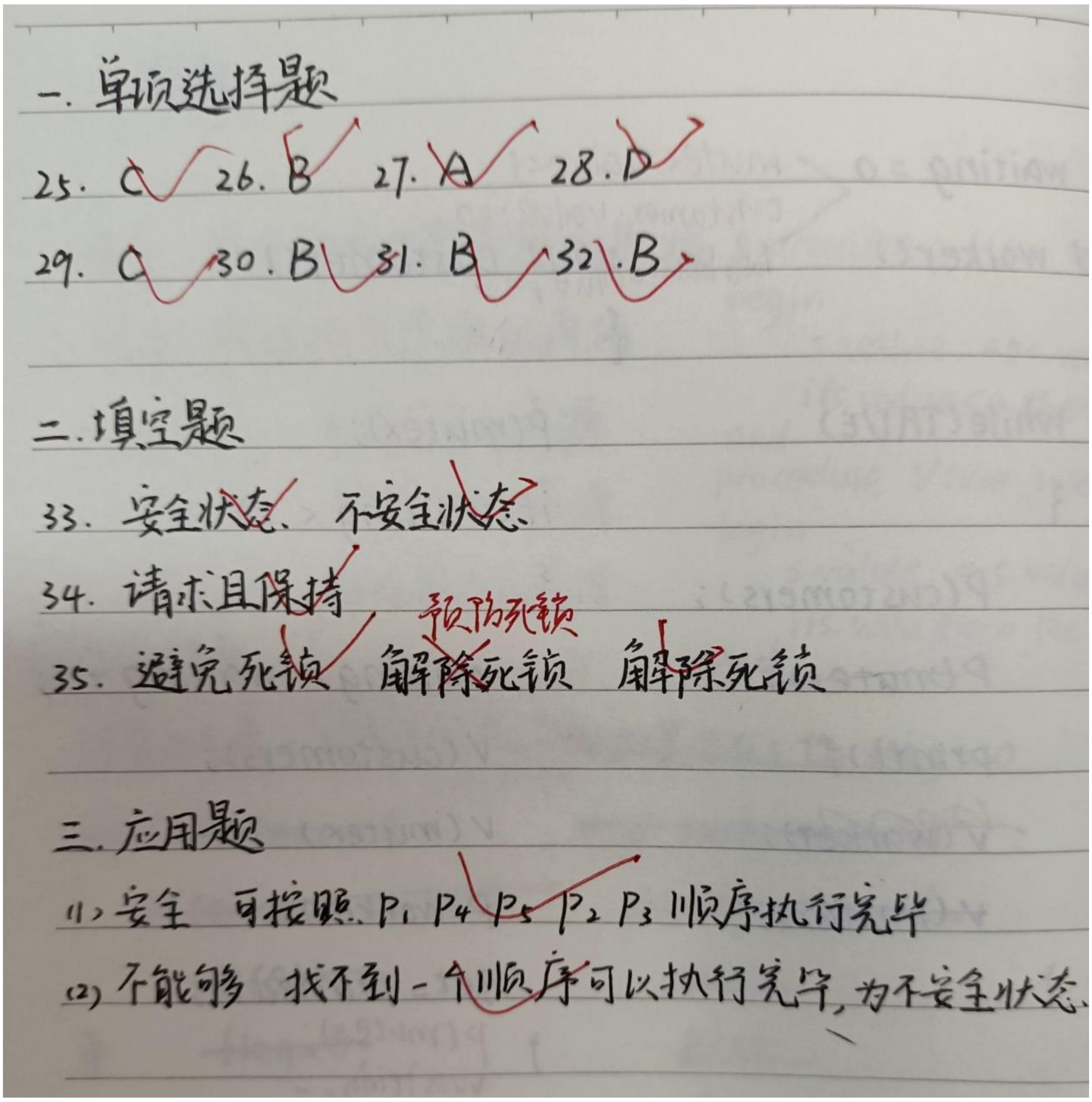
~~void worker() {~~ ~~void customer() {~~

~~if(s.value == 5) {~~ if(flag == 0) {

24.

```
int waiting = 0   mutex.value:=1
                  customer·value:=0
                  barbers.value:=5
void worker()
{
    while(TRUE)
    {
        P(customers);
        ~~P(mutex);~~
        print();
        ~~V(worker)~~;
        ~~V(mutex);~~
    }
}
```

```
void customer()
{
    P(mutex);
    if(waiting < 5)
    {
        waiting = waiting + 1;
        V(customers);
        V(mutex)
        P(worker);
        get_printed();
    }   P(mutex);
        waiting--
    else V(mutex);
    {
        V(mutex);
    }
}
```



一. 单项选择题

25. C 26. B 27. A 28. D

29. C 30. B 31. B 32. B

二. 填空题

33. 安全状态、不安全状态

34. 请求且保持

预防死锁

35. 避免死锁 解除死锁 解除死锁

三. 应用题

(1) 安全 可按照 P_1 P_4 P_5 P_2 P_3 顺序执行完毕

(2) 不能够 找不到一个顺序可以执行完毕，为不安全状态。