

系统架构设计师—开学第一课

姜美荣



目录

1.1 欢迎新同学

1.2 考试时间安排

1.3 系统架构设计师课程配置

1.4 关于备考

1.1 欢迎新同学

欢迎新同学

欢迎大家来到极客时间学习平台，报考系统架构设计师这个专业，我是这门课程的主讲老师姜美荣，从今天开始我们将一起学习，一起成长，我真诚的希望大家愉快且认真的学习、坦诚的交流，争取一次通过考试，拿到证书，实现梦想。

与君远相知,不道云海深。愿你努力，祝你幸运。

——姜美荣

1.2 考试时间安排



考试时间

2025 年下半年计算机技术与软件专业技术
资格（水平）考试工作安排

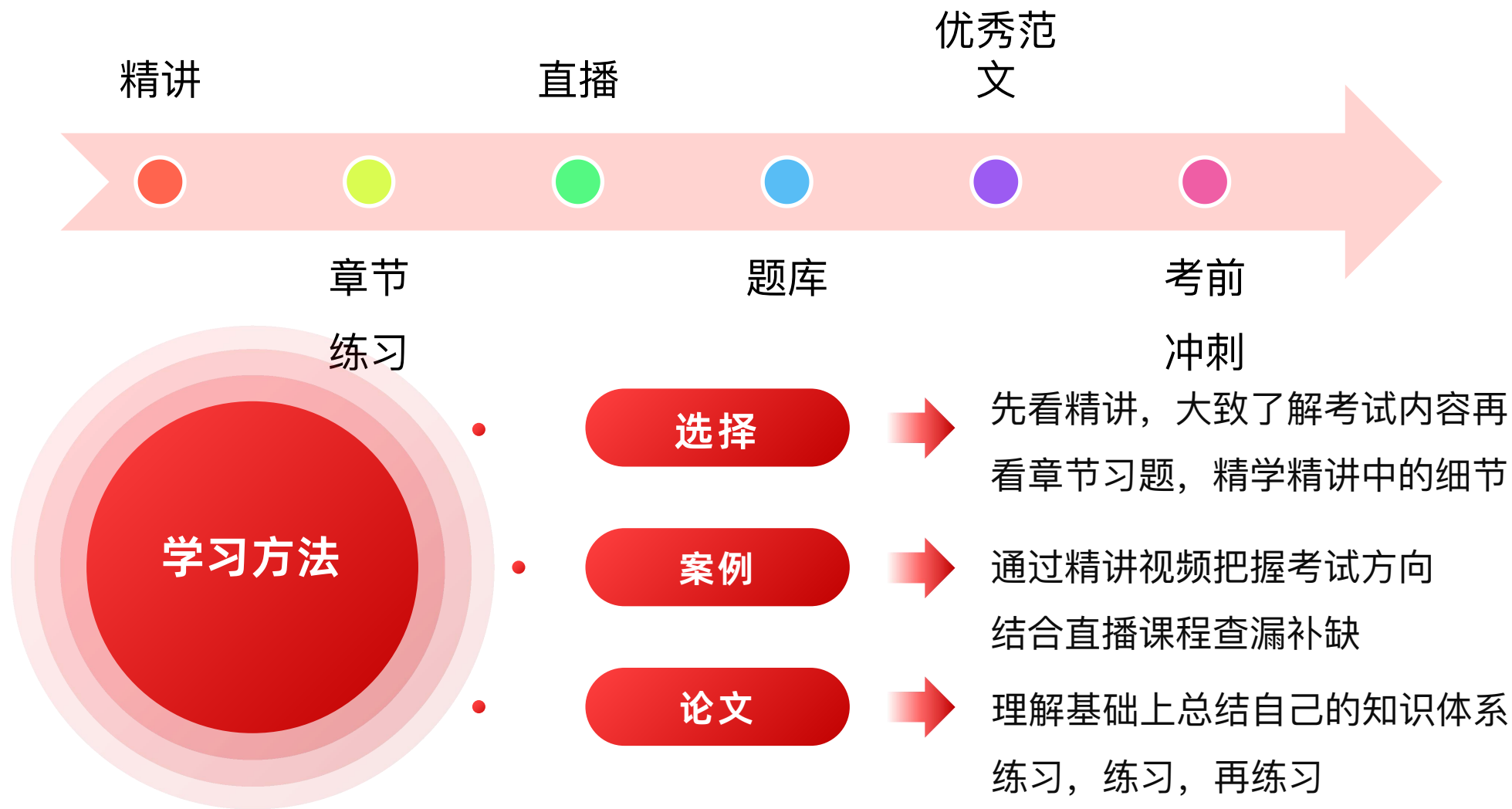
考试日期	级别	资格名称
11 月 8 日至 11 日	高级	系统分析师
		系统架构设计师
		网络规划设计师
		系统规划与管理师
	中级	软件设计师
		网络工程师
		信息安全工程师
		信息系统管理工程师
		多媒体应用设计师
		系统集成项目管理工程师
		网络管理员

考试时间

题型	数量	考试时间	做题时间	分数	合格分
单选题	75题	8: 30~12: 30	150分钟	75分	45
案例题	1+4选2		90分钟	75分	45
论文	4选1	14: 30~16: 30	120分钟	75分	45

1.3 系统架构设计师课程配置

课程内容配置



课程平台配置

课程平台

囊括所有课程资料、
录播&直播视频、重
难点及真题资料。



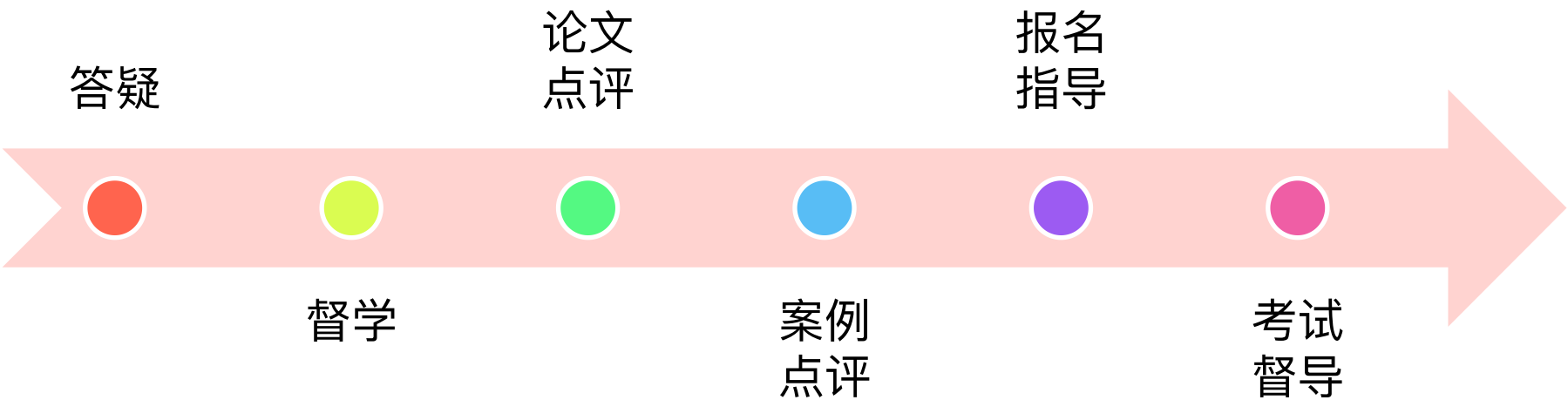
题库平台

提供章节练习、历年
真题试卷、模拟试卷
等多种练习形式。





课程服务配置



课程服务指导	
班主任	平台使用，学习计划，报名通知，报考通知，考试通知，证书领取等，除学习疑问，均可联系班主任。 (不知道找谁——找班主任)
助教 讲师	答疑解惑，学习指导，论文批改，直播授课，直播答疑等

1.4 关于备考

如何一次拿下证书？

- 录播为主
- 教材为辅导
- 直播为纲
- 刷题为王
- 及时总结错题
- 论文练习要充足

姜老师：一次通过考试，就是最低成本，3个月能考过，但仓促，现在报名正当时。

祝大家顺利上岸 成为优秀的系统架构设计师



系统架构设计师

计算机系统基础知识1

姜美荣

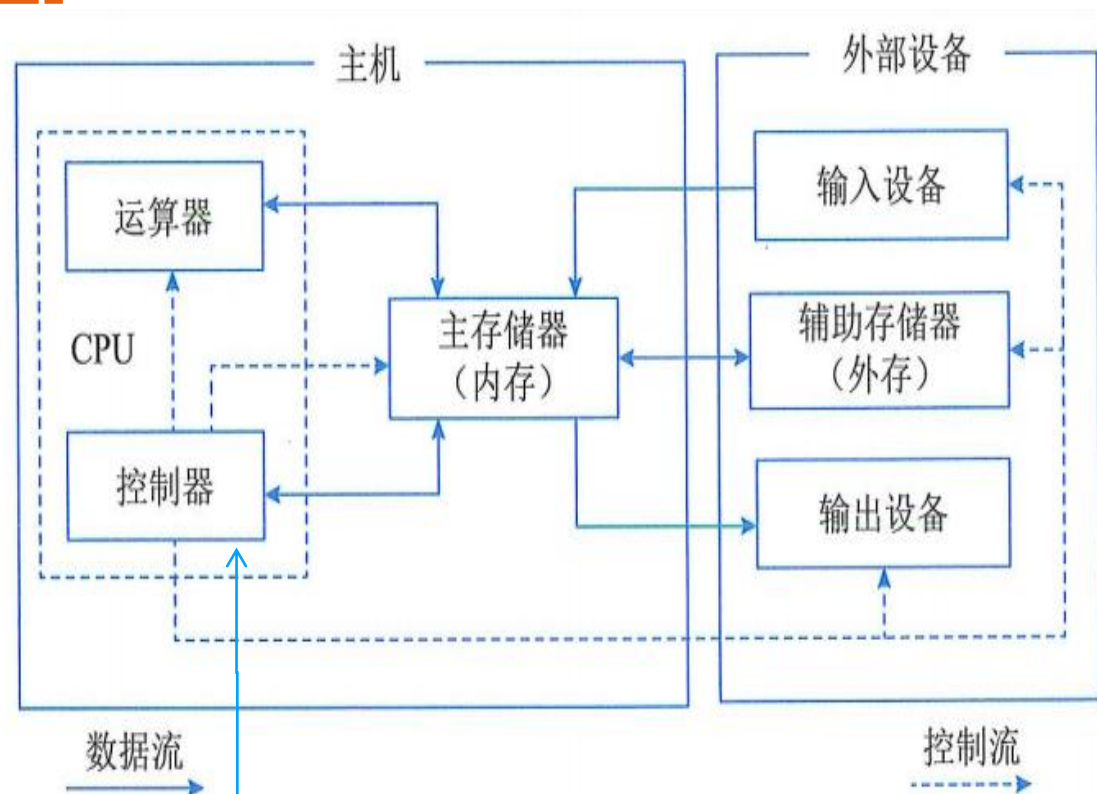


目录

计算机系统基础



计算机系统概述



程序控制、操作控制
时间控制、数据处理
其他如中断响应

计算机的冯·诺依曼体系结构

(1) 控制器：是分析和执行指令的部件，也是统一指挥并控制计算机各部件协调工作的中心部件，所依据的是机器指令。控制器的组成包含程序计数器、指令寄存器、指令译码器、时序部件、微操作控制信号形成部件和中断机构。

(2) 运算器：也称为算术逻辑单元，其主要功能是在控制器的控制下完成各种算术运算和逻辑运算。

(3) 主存储器：也称为内存储器（通常简称为“内存”或“主存”）。存储现场操作的信息与中间结果，包括机器指令和数据。

(4) 辅助存储器：也称为外存储器，通常简称为外存或辅存。存储需要长期保存的各种信息。

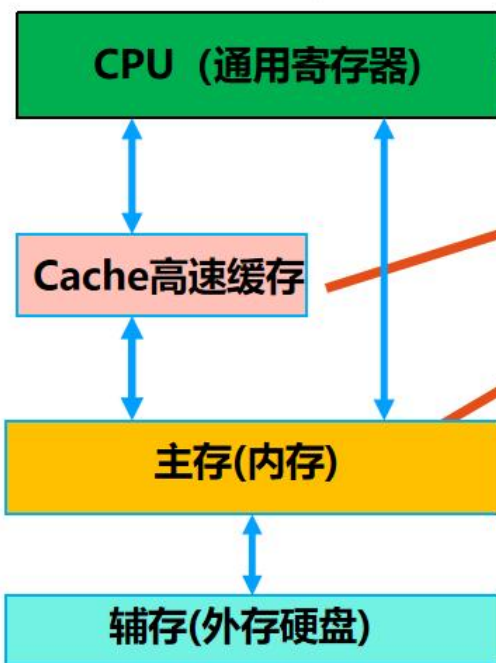
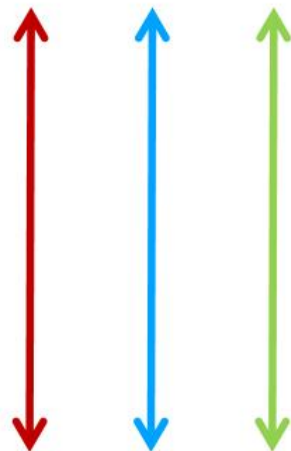
(5) 输入设备：任务是把人们编好的程序和原始数据送到计算机中去，并且将它们转换成计算机内部所能识别和接受的信息方式。

(6) 输出设备：任务是将计算机的处理结果以人或其他设备所能接受的形式送出计算机。目前，最常用的输出设备是打印机和显示器。

计算机系统

存储器是用来存放程序和数据部件，它是一个记忆装置，也是计算机能够实现“存储程序控制”的基础。在计算机系统中，规模较大的存储器往往分成若干级，称为存储器系统。传统的存储器系统一般分为**高速缓冲存储器(Cache)**、**主存储器**、**辅助存储器**三级。**存储器系统主要有4个性能指标**，分别是**存取时间、存储器带宽、存储器周期和数据传输率**。

容量 价格 速度



寄存器：容量小、速度快但是成本高

按照内容存取，主存到cache的映射由**硬件**完成。

分两类：DRAM

▲ 离CPU越近的存储器，速度越快，每字节的成本越高，同时容量也因此越小

▲ 局部性原理是层次化存储结构的支撑和理论依据

▲ 主存和辅存构成虚拟存储器

▲ 目前计算机三级存储体系为：Cache、主存和辅存。

典型真题

202505 系分

主存储器简称主存或内存，存放计算机运行时的程序和数据，并由 CPU 直接随机进行读 / 写。

主存储器不包括（ ）。

A. SRAM B. Cache C. DRAM D. ROM

答案：B

解析：主存储器（内存）是 CPU 能直接访问的存储部件，用于存放运行中的程序和数据，主要包括随机存取存储器（RAM）和只读存储器（ROM）

典型真题

202505 架构

一次可编程只读存储器的英文缩写是（ ）。

A.ROM B.PROM C.EPROM D.EEPROM

【解析】：

计算机中的PROM，全称为Programmable Read-Only Memory，即可编程只读存储器，也被称为一次可编程只读存储器（One-Time Programmable ROM，OTP ROM）。这是一种特殊的存储芯片，允许用户在制造完成后通过特定的编程设备将数据写入，但一旦写入，数据在正常操作下便不可更改。

ROM全称为Read-Only Memory，中文意思是“只读存储器”。

EPROM就是Erasable Programmable Read Only Memory，中文含意为“可擦除可编程只读存储器”。它是一种可重写的存储器芯片，并且其内容在掉电的时候也不会丢失。换句话说，它是非易失性的。它通过EPROM编程器进行编程，EPROM编程器能够提供比正常工作电压更高的电压对EPROM编程。一旦经过编程，EPROM只有在强紫外线的照射下才能够进行擦除。

EEPROM (Electrically Erasable Programmable read only memory)是指带电可擦可编程只读存储器。是一种掉电后数据不丢失的存储芯片。EEPROM 可以在电脑上或专用设备上擦除已有信息，重新编程。一般用在即插即用。

【参考答案】： B

计算机系统

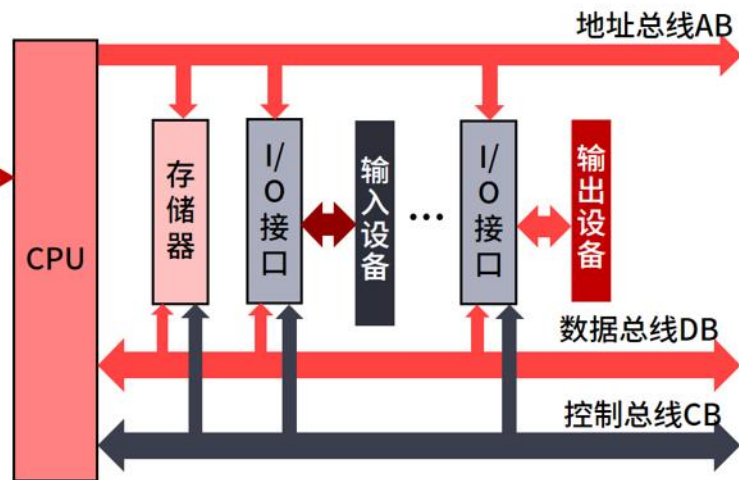
1、总线的分类

- 按总线相对于 CPU 或其他芯片的位置：内部总线、外部总线。
- 按总线功能：**地址总线、数据总线、控制总线**。
- 按连部件不同：片内总线、系统总线、通信总线。

片内总线	CPU内部的总线
系统总线	CPU、主存、I/O设备等
通信总线	计算机系统之间或与其他系统之间

- 按照总线中数据线的多少：并行总线和串行总线。

名称	数据线	特点	应用
并行总线	多条双向数据线	有传输延迟，适合近距离连接。	系统总线 (计算机各部件)
串行总线	一条双向数据线 或两条单向数据线	速率不高，但适合长距离连接。	通信总线 (计算机之间或 计算机与其他系统间)



计算机系统

计算机中，系统总线用于（46）连接。

- (46) A. 接口和外设
B. 运算器，控制器和寄存器
C. 主存及外设部件
D. DMA控制器和中断控制器

答案：C 解析 按连接部件不同，总线可分为：片内总线、系统总线、通信总线。片内总线是芯片内部的总线，如CPU内部的总线。系统总线是CPU、主存、I/O设备各大部件之间的信息传输线。通讯总线用于计算机系统之间或与其他系统之间的通信。

计算机系统

程序控制(查询)方式:分为无条件传送和程序查询方式两种。方法简单，硬件开销小，但I/O能力不高，严重影响CPU的利用率。

程序中断方式:与程序控制方式相比，中断方式因为CPU无需等待而提高了传输请求的响应速度。**CPU与I/O传输可并行**

DMA方式:DMA方式是为了在主存与外设之间实现高速、批量数据交换而设置的。

DMA方式比程序控制方式与中断方式都高效。**CPU与I/O传输可并行**

通道方式

I/O处理机

计算机系统

202505系分

如果 I/O 设备与存储设备间的数据交换不经过 CPU 来完成，则这种数据交换方式是（ ）。

A. 中断方式 B. DMA 方式 C. 无条件存取方式 D. 程序查询方式

答案：B

解析：直接内存访问（Direct Memory Access），通过 DMA 控制器（DMAC）实现 I/O 设备与内存的数据直接传输，全程无需 CPU 介入。CPU 仅在传输开始前配置 DMA 控制器，传输完成后处理中断，大幅提高效率。

计算机系统

指令系统类型	指令	寻址方式	实现方式	其它
CISC (复杂)	数量多，使用频率相差很大、可变定长	多种寻址方式	微程序控制技术	周期长，指令直接在主存处理，执行速度慢
RISC (精简)	数量少，使用频率相近，定长格式，大部分为单周期指令，只有LOAD/Store操作内存	支持方式少	增加了通用寄存器；硬布线逻辑控制为主；适合采用流水线。	优化编译，对编译的要求高，支持高级语言

典型真题

202505系分.RISC 指令系统的特点不包括（ ）。

- A. 寻址方式种类尽量减少，指令功能尽可能强
- B. 指令长度固定，指令种类少
- C. 增加寄存器的数目，以尽量减少访存次数
- D. 选取使用频率最高的一些简单指令，以及很有用但不复杂的指令

答案：A

解析：RISC（精简指令集计算机）指令系统强调指令的简单性，指令功能相对单一，而不是指令功能尽可能强。CISC（复杂指令集计算机）才倾向于使指令功能尽可能强，一条指令能

操作系统

操作系统(OS, Operating System)

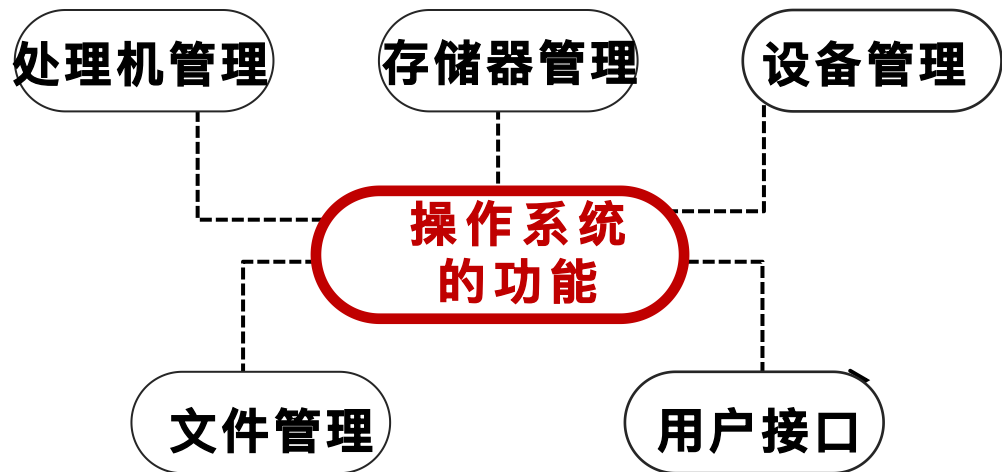
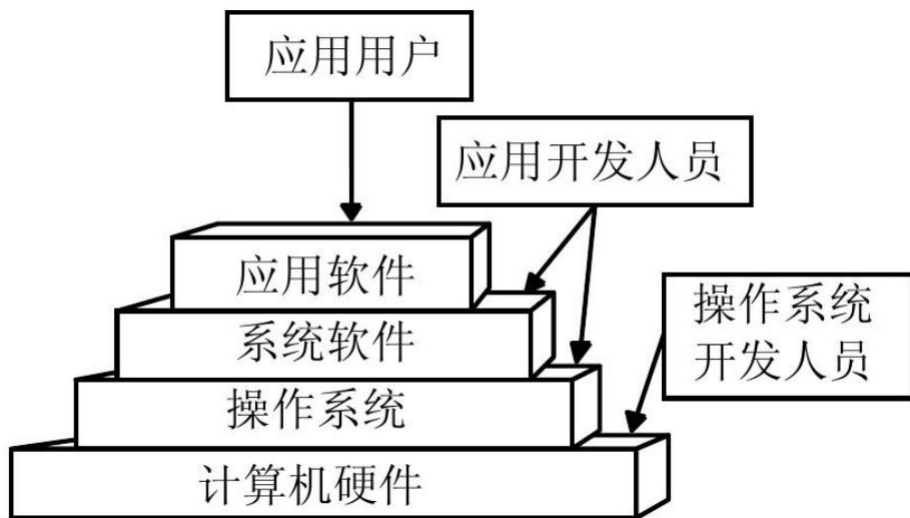
人机之间的接口

应用软件与硬件之间的接口

为应用程序的开发和运行提供一个高效率的平台

管理系统的硬件、软件、数据资源

控制程序运行



202505系分.操作系统关心的问题不包括()。

- A. 管理计算机系统资源
- B. 提供高级语言的编译
- C. 提供用户程序与硬件系统的界面
- D. 管理应用软件的运行

答案: B

解析: 提供高级语言的编译是编译器的功能。编译器将高级语言编写的源程序翻译成机器语言或目标代码, 操作系统并不直接参与高级语言的编译过程。

操作系统

操作系统的分类

分类	特点
批处理操作系统	单道批：一次一个作业入内存，作业由程序、数据、作业说明书组成 多道批：一次多个作业入内存，特点：多道、宏观上并行微观上串行(提高CPU和外部设备的利用效率)
分时操作系统	采用时间片轮转的方式为多个用户提供服务，每个用户感觉独占系统 特点：多路性、独立性、交互性和及时性
实时操作系统	实时控制系统和实时信息系统 交互能力要求不高，可靠性要求高(规定时间内响应并处理)
网络操作系统	方便有效共享网络资源，提供服务软件和有关协议的集合 主要的网络操作系统有：Unix、Linux和Windows Server系统
分布式操作系统	任意两台计算机可以通过通信交换信息 是网络操作系统的更高级形式，具有透明性、可靠性和高性能等特性
微机操作系统	Windows：Microsoft开发的图形用户界面、多任务、多线程操作系统 Linux：免费使用和自由传播的类Unix操作系统，多用户、多任务、多线程和多CPU的操作系统
嵌入式操作系统	运行在智能芯片环境中 特点：微型化、可定制(针对硬件变化配置)、实时性、可靠性、易移植性(HAL和 BSP支持)

操作系统

操作系统的分类

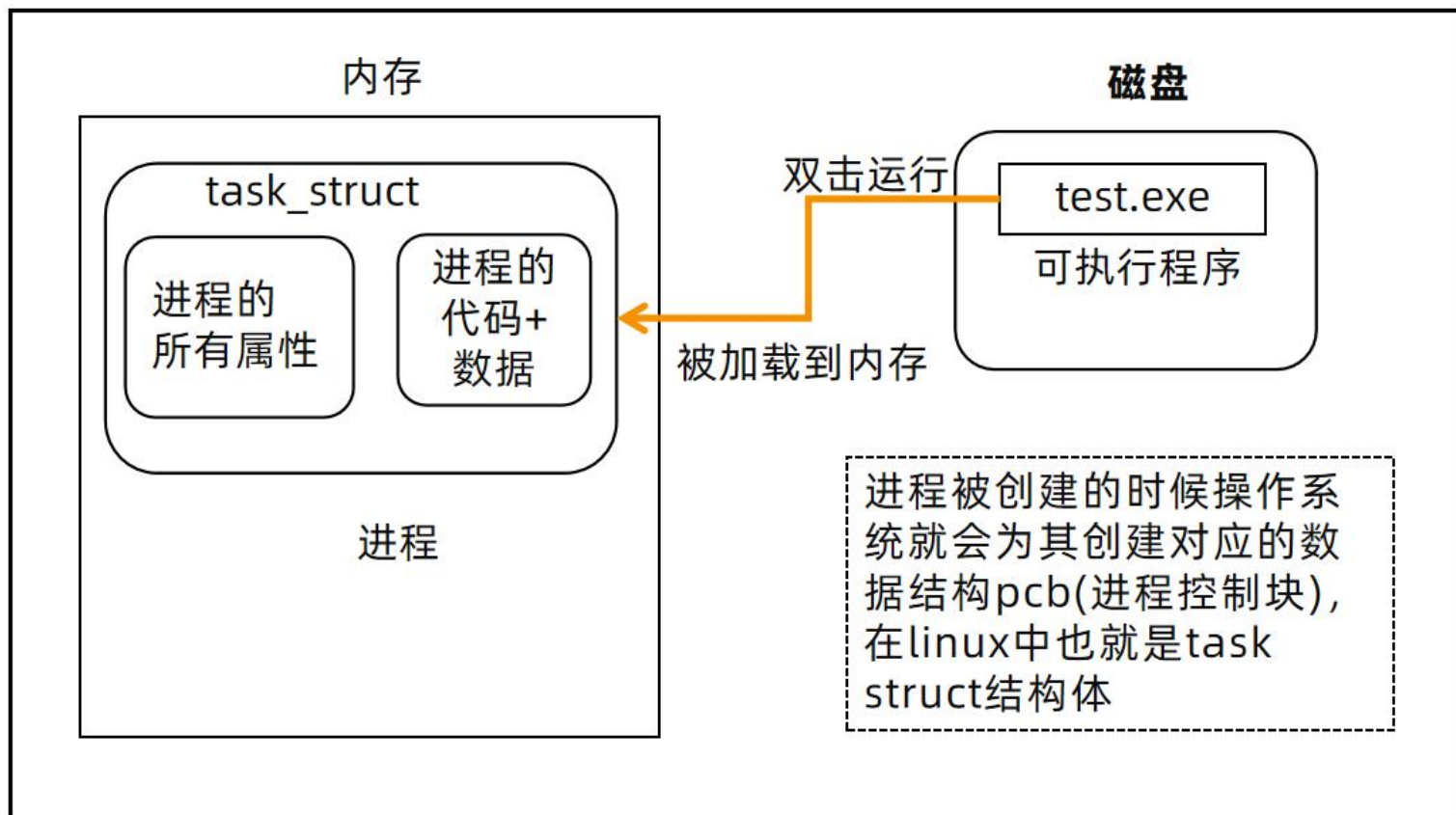
操作系统中采用多道程序设计技术可以提高CPU和外部设备的()。

A、利用率. B、可靠性 C、兼容性 D、稳定性

答案A

进程管理-基本概念

1、**进程**是程序在一个数据集合上运行的过程，进行资源分配和调度的基本

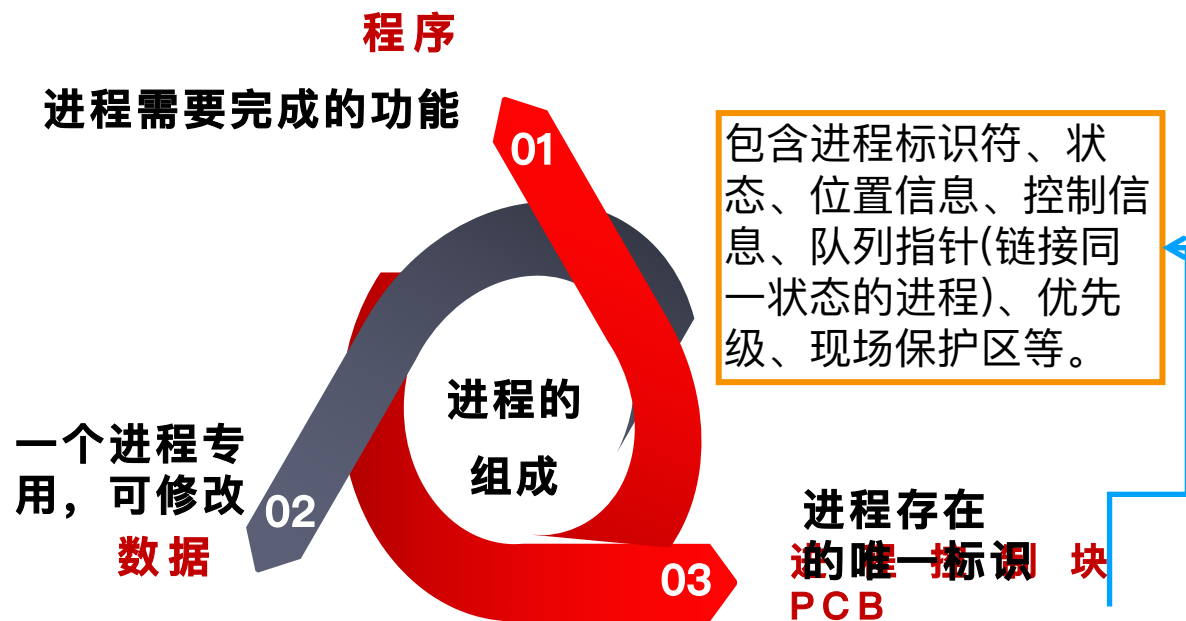
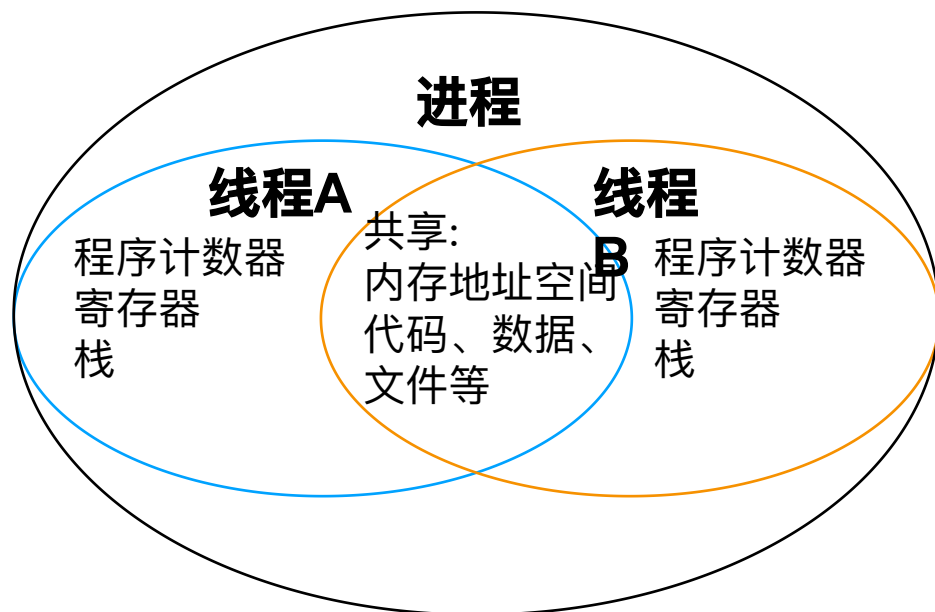


进程管理-基本概念

1、**进程**是程序在一个数据集合上运行的过程，进行资源分配

和调度的基本单位。 (2) **动态特征（最基本特征）**

- 程序：是一组有序指令的集合，是静态的概念。先有程序后进程。
- 进程：进程实体的一次执行过程，有生命周期。是动态性（.exe）。
- 线程：是进程中的一个实体，是被系统独立分配和调度的基本单位



典型真题

202505架构.下列选项中会导致线程从执行态变为就绪态的是

- A.键盘输入
- B.主动让出CPU
- C.执行信号量的wait () 操作
- D.缺页异常

【解析】：

任务的工作状态最简单的可分为三种：执行态、就绪态和阻塞态，其转换关系见图16-10。①执行态：当任务已获得处理机，其程序正在处理机上执行，此时的任务状态称为执行状态。

②就绪状态：当任务已分配到除CPU 以外的所有必要的资源，只要获得处理机便可立即执行，这时的任务状态称为就绪状态。

③阻塞状态：正在执行的任务，由于等待某个事件发生而无法执行时，便放弃处理机而处于阻塞状态。引起进程阻塞的事件可有多种，例如，等待I/O完成、申请缓冲区不能满足、等待信件（信号）等。

三种基本状态转换：

(1) 就绪—执行：处于就绪状态的任务，当任务调度程序为之分配了处理机后，该任务便由就绪状态转变成执行状态。

(2) 执行—就绪：处于执行状态的任务在其执行过程中，因分配给它的一个时间片已用完而不得不让出处理机，于是任务从执行状态转变成就绪状态。

(3) 执行—阻塞：正在执行的任务因等待某种事件发生而无法继续执行时，便从执行状态变成阻塞状态。

(4) 阻塞—就绪：处于阻塞状态的任务，若其等待的事件已经发生，于是任务由阻塞状态转变为就绪状态。

【参考答案】：B



图 16-10 任务三种状态的转换

典型例题

2025系分.关于进程和线程说法不正确的是（ ）。

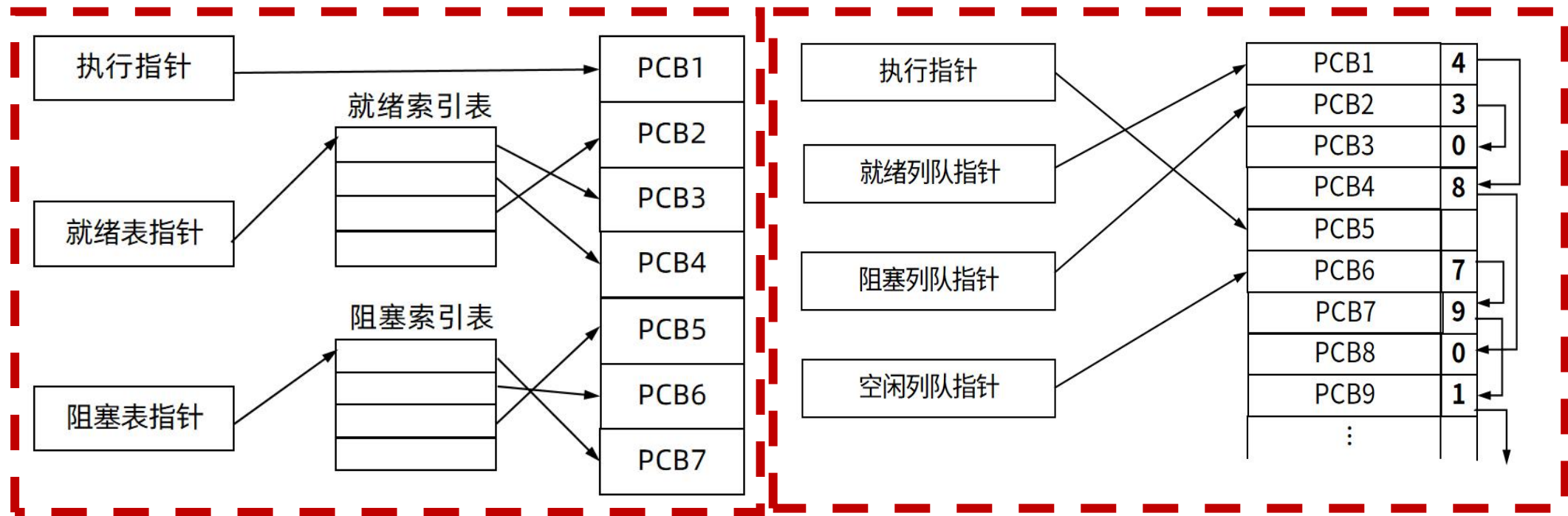
- A.一个进程可以创建一个或多个线程
- B.一个进程可以创建一个或多个进程
- C.一个线程可以创建一个或多个线程
- D.一个线程可以创建一个或多个进程

答案：D

解析：线程不能直接创建进程，创建进程的主体是进程本身。

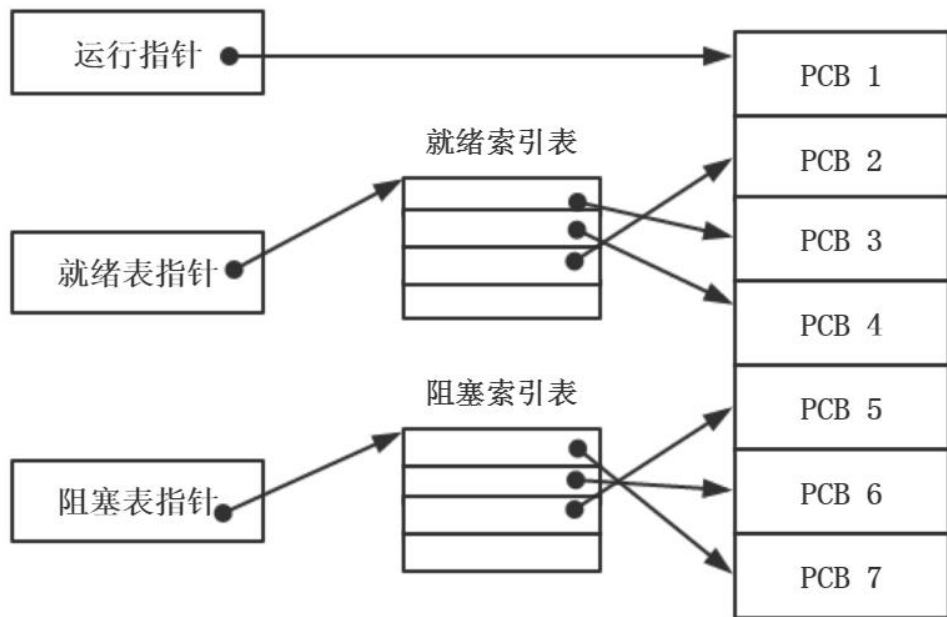
进程管理-PCB的组织方式

线性方式	把所有PCB组织在一张线性表中，每次查找是需要扫描全表。
链接方式	把具有同一状态的 PCB，用其中的链接字链接成一个队列，PCB存储在一个连续的区域。
索引方式	同一状态的进程归入一个索引表，多个状态对应多个不同的索引表。



典型例题

下图所示的PCB（进程控制块）的组织方式是（1），图中（2）。



- (1) A. 链接方式 B. 索引方式 C. 顺序方式 D. Hash
- (2) A. 有1个运行进程、2个就绪进程、4个阻塞进程
B. 有2个运行进程、3个就绪进程、2个阻塞进程
C. 有1个运行进程、3个就绪进程、3个阻塞进程
D. 有1个运行进程、4个就绪进程、2个阻塞进程

答案：BC

进程间的通信-同步与互斥

1、基本概念

进程间两种形式的制约关系

- (1) 间接相互制约关系(**互斥**): 源于资源共享(临界资源), 同类资源的竞争关系。
- (2) 直接相互制约关系(**同步**): 源于进程合作(进程之间的顺序或逻辑关系), 速度有差异, 一定情况下停下等待, 进程间的协作关系。
- **临界资源**: 把**一段时间内只允许一个进程访问的资源**称为临界资源或独占资源所以进程间需要互斥地访问共享的资源, 如打印机。
- **临界区**: 每个进程中访问临界资源的那段**代码**称为临界区。

进程间的通信-PV操作-信号量机制

2、信号量机制

信号量： $S \geq 0$ 表示某资源的可用数（全局变量）；若 $S < 0$ ，则其绝对值表示阻塞队列中等待该资源的进程数。

- 利用信号量进行进程互斥
- 利用信号量进行进程同步
- 利用信号量进行进程前驱关系

为使多个进程互斥的访问某个临界资源，需为该资源设置一互斥的信号量（mutex），并设置其初始值为1（有几个临界资源就设置为几，默认为1），然后将各进程访问资源的临界区CS设置于PV之间即可。

进程间的通信-PV操作（互斥）

进入临界区之前先执行P操作（可能阻塞当前进程）

离开临界区之后执行V操作（可能唤醒某个进程）

```
main( )  
{  
    /*信号量mutex*/  
    int mutex=1;  
    cobegin//并发  
    Pa( );  
    Pb( );  
    Pc( );  
    coend//并发结束  
}
```

```
Pa( )  
{
```

```
    P(mutex);  
    CSa  
    V(mutex);  
}
```

```
Pc( )  
{
```

```
    P(mutex);  
    CSc  
    V(mutex);  
}
```

```
Pb( )
```

```
{  
    P(mutex);  
    CSb  
    V(mutex);  
}
```

进程间的通信-PV操作

PV操作：（操作系统原语）P是荷兰语的Passeren，V是荷兰语的Verhoog。

P操作：

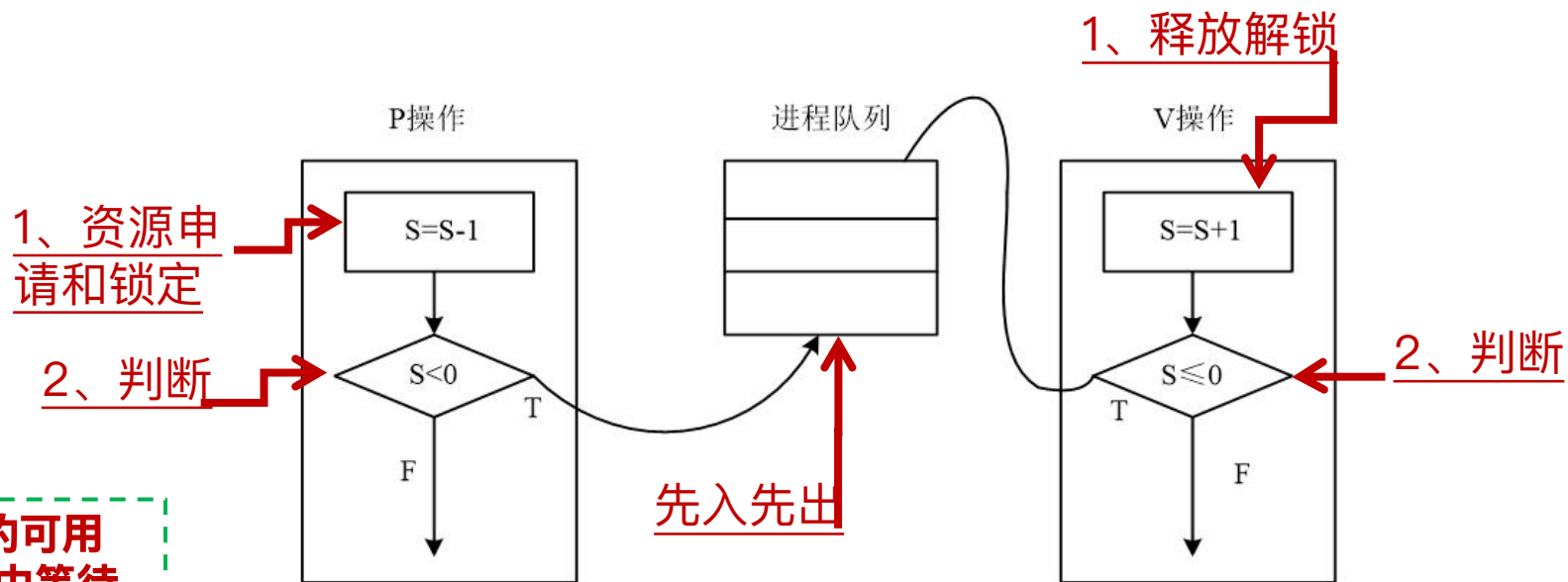
- ①将信号量S的值减1，即 $S=S-1$ ；
- ②若 $S \geq 0$ ，则执行P操作的进程继续执行；
若 $S < 0$ ，则置该进程为阻塞状态（因为无可用资源），并将其插入阻塞队列。

V操作：

- ①将信号量S的值加1，即 $S=S+1$ ；
- ②若 $S > 0$ ，则执行V操作的进程继续执行；
若 $S \leq 0$ ，则从阻塞状态唤醒一个进程，并将其插入就绪队列，
然后执行V操作的进程继续。

信号量S的物理意义： $S \geq 0$ 表示某资源的可用数，若 $S < 0$ ，则其绝对值表示阻塞队列中等待该资源的进程数，即资源不足数。

- P操作是先加锁申请资源，然后自减1，判断信号量是否小于0（阻塞的操作）
- V操作是释放资源，增量操作 $S=S+1$ ，然后唤醒阻塞列表的等待进程。
- PV操作必须成对出现，但是没有严格的先后顺序。



典型真题

202505系分.某系统使用信号量实现进程互斥访问共享资源，信号量的初始值应设置为（ ）。

A. 1 B. 2 C. 0 D. -1

答案 A

解析：信号量（Semaphore）是用于进程同步的机制，初始值的设置取决于具体场景：

互斥访问：当信号量用于实现互斥（如临界区保护）时，初始值应为 1（即二元信号量）。

进程进入临界区前执行 P 操作（信号量减 1），若初始值为 1，则减 1 后变为 0，表示资源被占用；

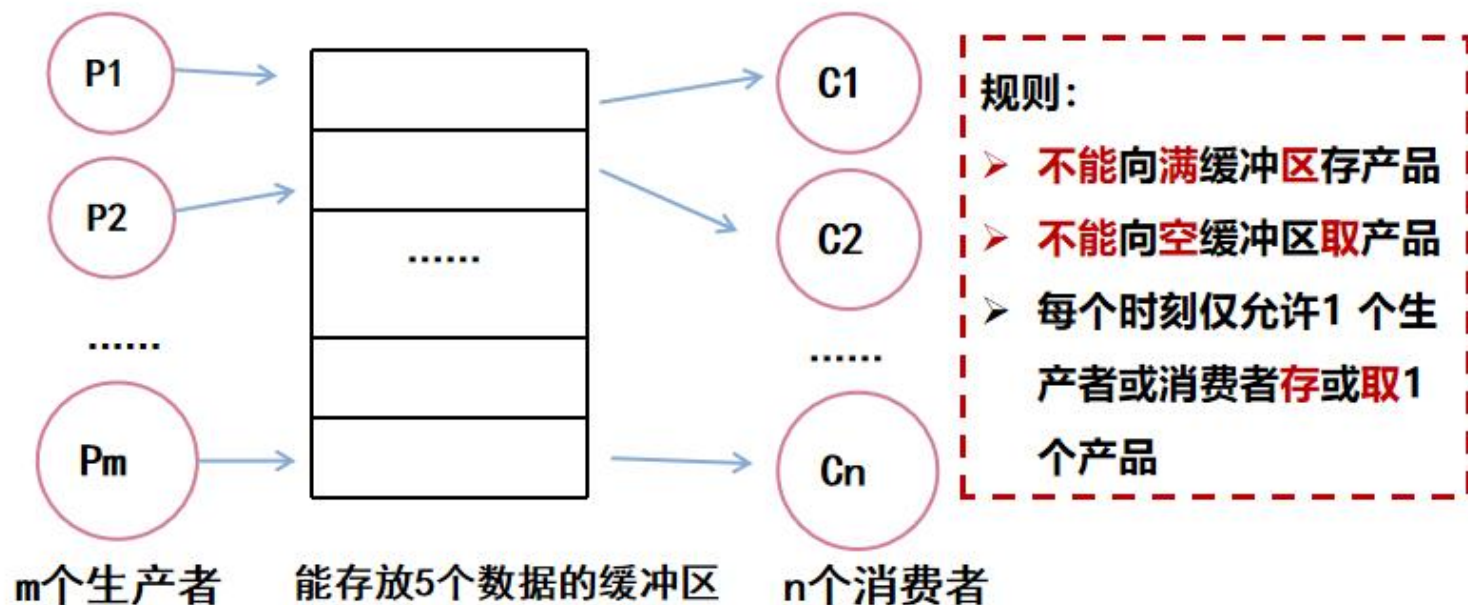
进程离开临界区时执行 V 操作（信号量加 1），使信号量恢复为 1，允许其他进程进入。

进程间的通信-PV操作（同步）

同步问题：

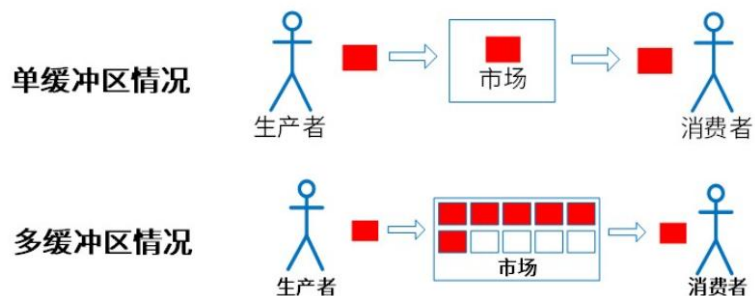
运行条件不满足时，能让进程暂停（在关键操作之前执行P操作）

运行条件满足时，能让进程继续（在关键操作之后执行V操作）



进程间的通信-PV操作（同步）

未使用PV操作：



生产者
生产一个产品，送产品到缓冲区。--检查空间是否充足。

消费者
从缓冲区取产品，消费产品。--检查产品是否充足。

使用 PV操作

生产者：

生产一个产品；

P(S1);

送产品到缓冲区；

V(S2);

消费者：混合关系

P(S2);

从缓冲区取产品；

V(S1);

消费产品；

互斥与同步

生产者：

生产一个产品；

P(S1);

P(S0);

送产品到缓冲区；

V(S0);

V(S2);

消费者：

P(S2);

P(S0);

从缓冲区取产品；

V(S0);

V(S1);

消费产品；

S1初值为1，S2初值为0

S1初值为1，S2初值为0，S0的初值为1

典型真题

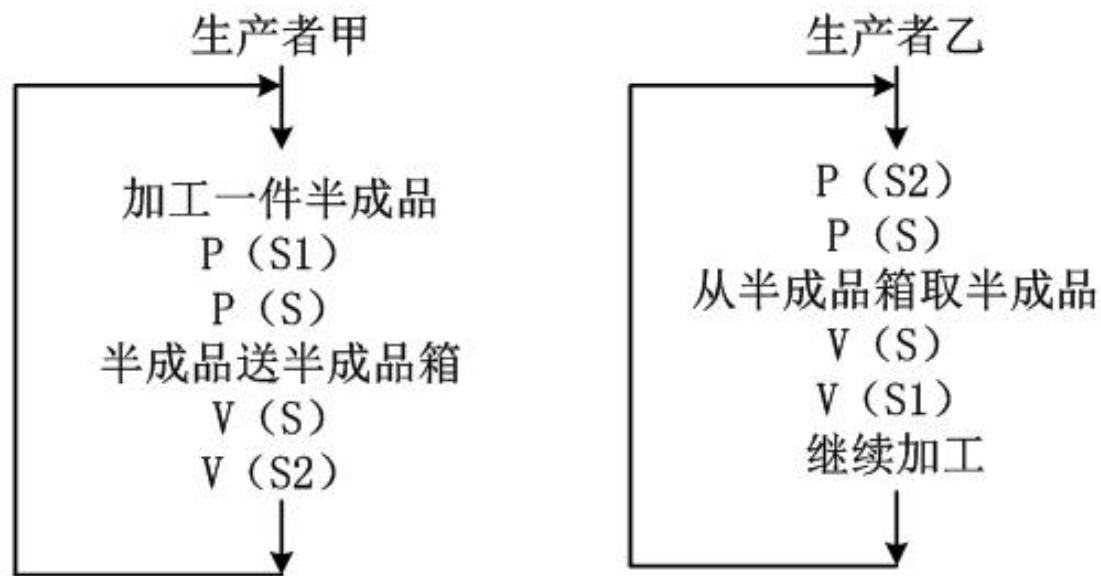
某企业生产流水线M共有两位生产者，生产者甲不断地将其工序上加工的半成品放入半成品箱，生产者乙从半成品箱取出继续加工。假设半成品箱可存放n件半成品，采用PV操作实现生产者甲和生产者乙的同步可以设置三个信号量S、S1和S2，其同步模型如下图所示。

信号量S是一个互斥信号量，初值为（ ）；S1、S2的初值分别为（ ）。

A.0 B.1. C.n D.任意正整数

A.n、0. B.0、n C.1、n D.n、1

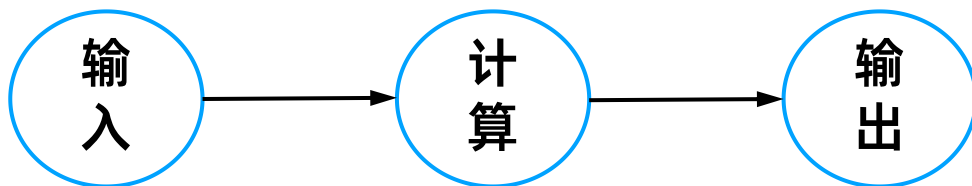
答案：BA



进程管理-前趋图

程序顺序执行

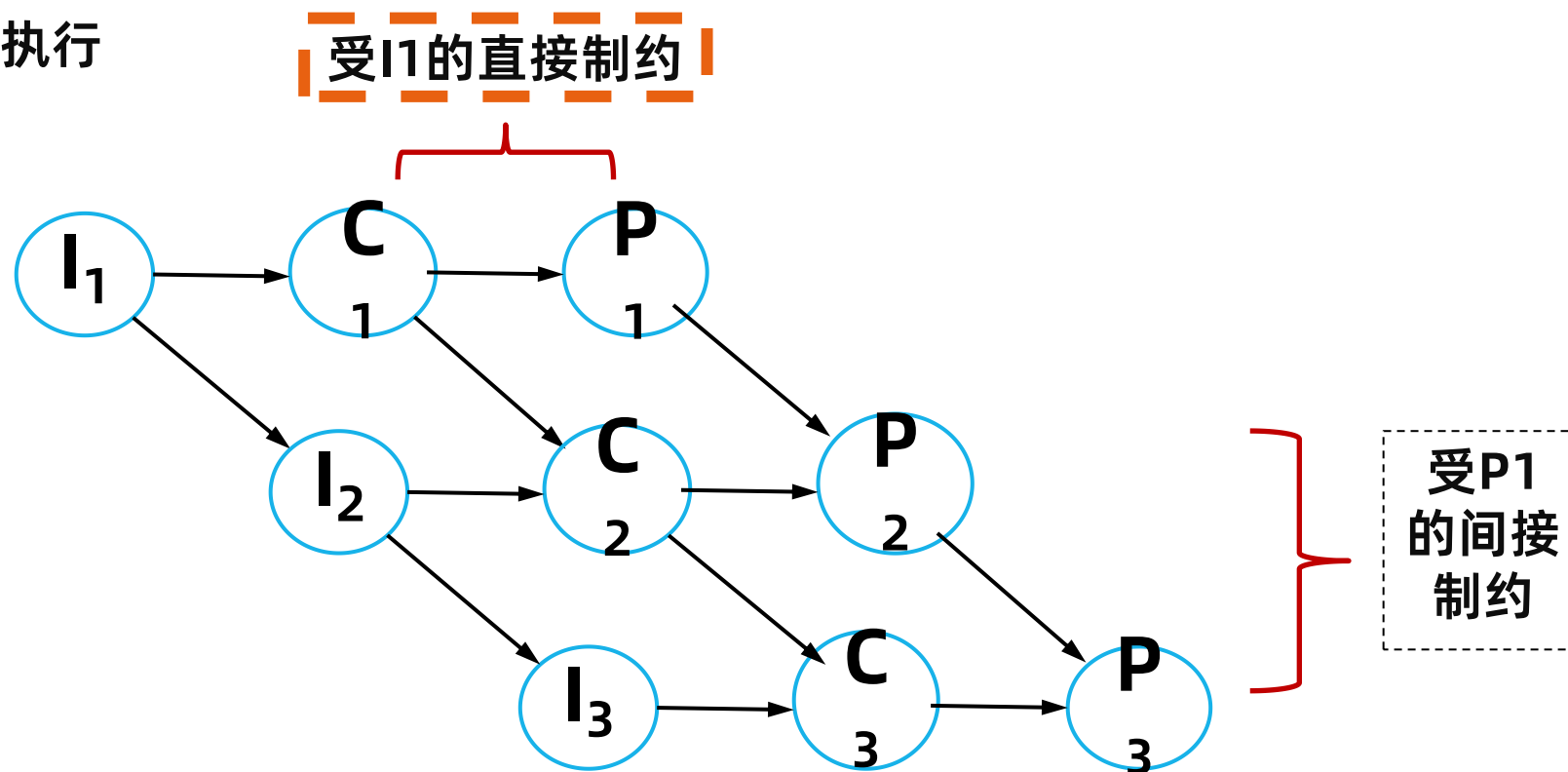
前趋图：是一个有向无循环图，由结点和有向边组成，结点代表各程序段的操作，而结点间的有向边表示两个程序段操作之间存在的前趋关系(\rightarrow)。程序段 P_i 和 P_j 的前趋关系表示成 $P_i \rightarrow P_j$ ，其中， P_i 是 P_j 的前趋， P_j 是 P_i 的后继，其含义是 P_i 执行结束后 P_j 才能执行。例如，



特征：顺序性、封闭性、可再现性

进程管理-前趋图

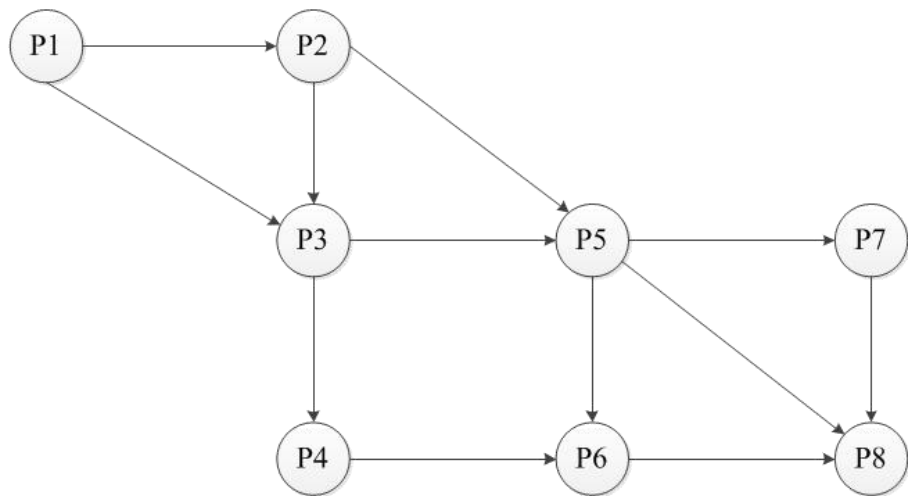
程序并发执行



特征：无封闭性、互相制约、不再一一对应

典型例题

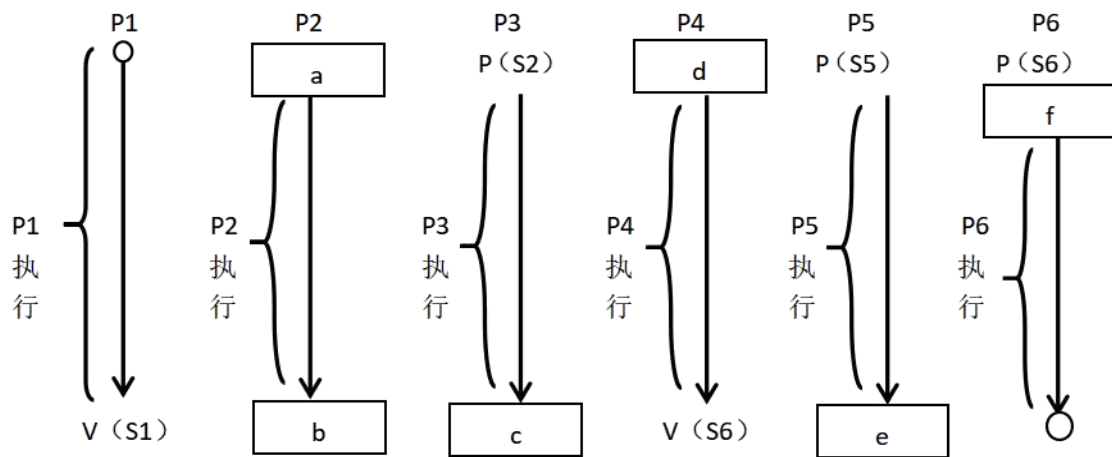
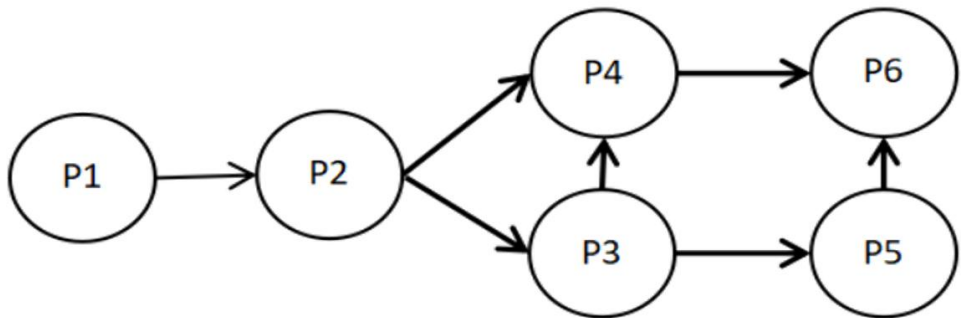
前趋图(Precedence Graph) 是一个有向无环图,记为: $\rightarrow = \{(P_i, P_j) | P_i \text{ must complete before } P_j \text{ may start}\}$, 假设系统中进程 $P = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8\}$, 且进程的前趋图如下, 该前趋图的前趋关系是 ()。



- A. $\rightarrow = \{ (P_1, P_2), (P_3, P_1), (P_4, P_1), (P_5, P_2), (P_5, P_3), (P_6, P_4), (P_7, P_5), (P_7, P_6), (P_5, P_6), (P_4, P_5), (P_6, P_7), (P_7, P_6) \}$
- B. $\rightarrow = \{ (P_1, P_2), (P_1, P_3), (P_2, P_5), (P_2, P_3), (P_3, P_4), (P_3, P_5), (P_4, P_5), (P_5, P_6), (P_5, P_7), (P_8, P_5), (P_6, P_7), (P_7, P_8) \}$
- C. $\rightarrow = \{ (P_1, P_2), (P_1, P_3), (P_2, P_3), (P_2, P_5), (P_3, P_4), (P_3, P_5), (P_4, P_6), (P_5, P_6), (P_5, P_7), (P_5, P_8), (P_6, P_8), (P_7, P_8) \}$
- D. $\rightarrow = \{ (P_1, P_2), (P_1, P_3), (P_2, P_3), (P_2, P_5), (P_3, P_6), (P_3, P_4), (P_4, P_7), (P_5, P_6), (P_6, P_7), (P_6, P_5), (P_7, P_5), (P_7, P_8) \}$

典型真题

若用PV操作控制进程P1、P2、P3、P4、P5和P6并发执行的过程，则需要设置7个信号量S1、S2、S3、S4、S5、S6和S7，且信号量S1~S7的初值都等于零。如右边的进程执行图中，a和b处应分别填写（ ）；c和d处应分别填写（ ），e和f处应分别填写（ ）。



A. $P(S1)$ 和 $V(S2)$ $V(S3)$

B. $V(S1)$ 和 $P(S2)$ $V(S3)$

C. $P(S1)$ 和 $P(S2)$ $V(S3)$

D. $V(S1)$ 和 $V(S2)$ $V(S3)$

A. $V(S4)$ 和 $P(S3)$ $P(S4)$

B. $P(S4)$ 和 $P(S3)$ $P(S4)$

C. $P(S3)$ 和 $V(S4)$ $V(S5)$

D. $V(S4)$ $V(S5)$ 和 $P(S3)$ $P(S4)$

A. $P(S6)$ 和 $V(S7)$

B. $V(S6)$ 和 $P(S6)$

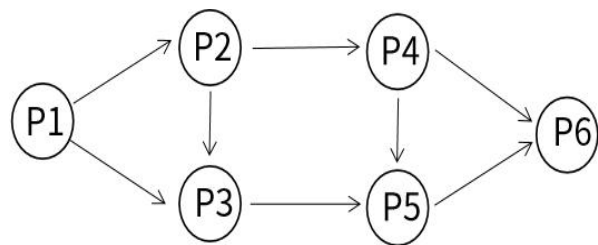
C. $V(S7)$ 和 $P(S7)$

D. $V(S6)$ 和 $V(S7)$

答案: ADC

典型真题

进程P1、P2、P3、P4、P5和P6的前趋图如下所示。假设用PV操作来控制这6个进程的同步与互斥的程序如下，程序中的空①和空②处应分别为 (27)。空③和空④处应分别为 (28)，空⑤和空⑥处应分别为 (29)。



- (27) A. V (S1) V (S2) 和P (S2) P (S3)
 B. V (S1) P (S2) 和V (S3) P (S4)
 C. V (S1) V (S2) 和V (S3) V (S4)
 D. P (S1) P (S2) 和V (S2) V (S3)

- (28) A. V (S5) 和V (S6) V (S7)
 B. V (S5) 和V (S6) P (S7)
 C. P (S3) 和V (S6) V (S7)
 D. P (S5) 和P (S6) V (S7)

- (29) A. V (S6) 和P (S7) P (S8)
 B. P (S8) 和P (S7) P (S8)
 C. P (S8) 和P (S7) V (S8)
 D. V (S8) 和P (S7) P (S8)

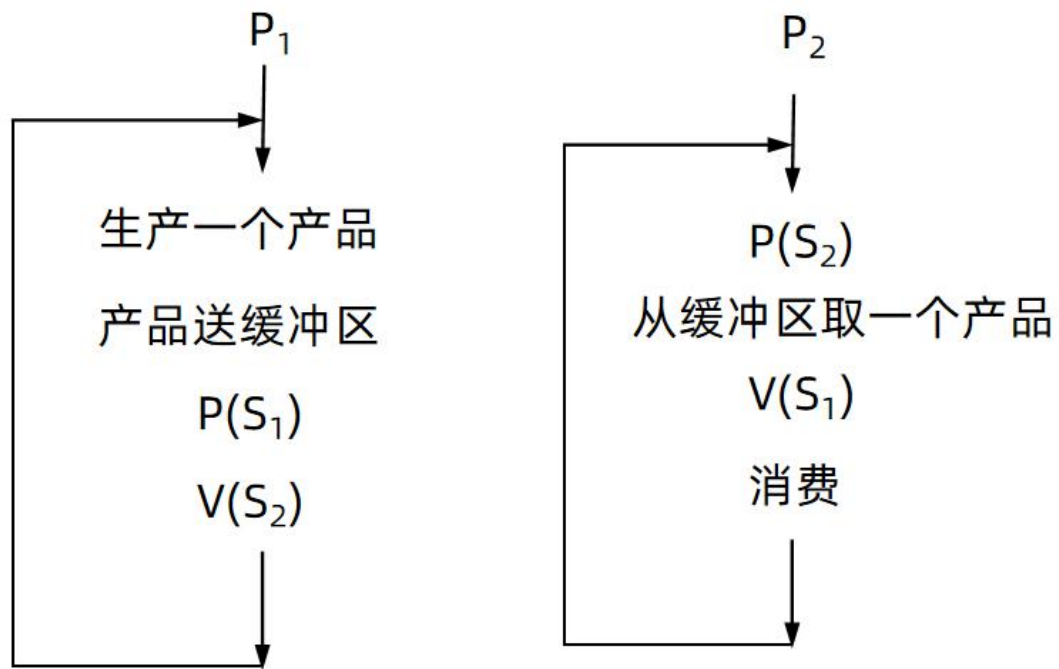
答案：CCD

```

Begin
S1, S2, S3, S4, S5, S6, S7, S8; semaphore;           //定义信号量
S1:=0; S2:=0; S3:=0; S4:=0; S5:=0; S6:=0; S7:=0; S8:=0;
Cobegin
  process P1      process P2      process P3      process P4      process P5      process P6
  Begin          Begin          Begin          Begin          Begin          Begin
                P(S1)          P(S2)          P(S4)          P(S5)          P(S6)
                P1执行;        P2执行;        P4执行;        P5执行;
                ①              ②              ③              ④              ⑤              ⑥
                End;          End;          P3执行;        End;          End;          P6执行;
                End;          End;          V(S5);        End;          End;          End;
                End;          End;          End;
  Coend;
End.
  
```


进程管理-死锁

死锁：死锁是指**两个以上的进程互相都要求使用对方已经占有的资源**而导致无法继续运行下去的现象；

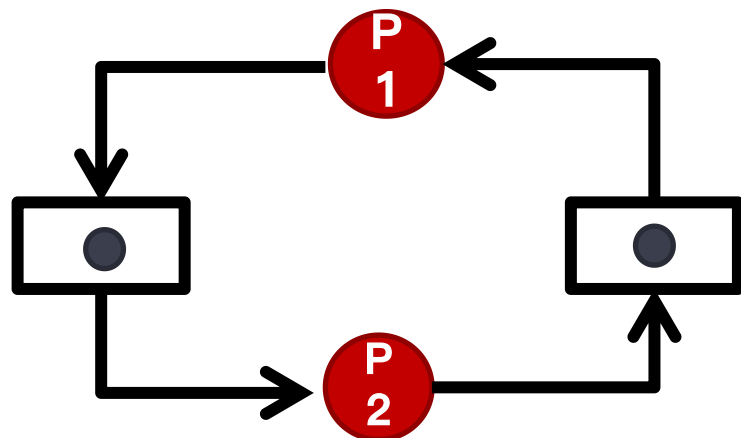
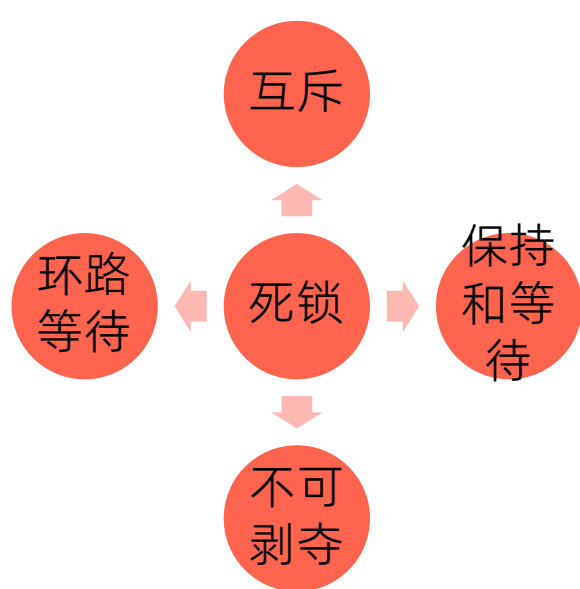


PV操作引起的死锁

进程管理-死锁

死锁产生的四个必要条件

- a) 保持和等待：进程在请求其余资源时，不主动释放已经占有的资源。
- b) 不可剥夺：进程已经占用的资源，不会被强制剥夺
- c) 环路等待：环路中每一条边是进程在请求另一个进程已经占有的资源
- d) 互斥：任一时刻只允许一个进程使用资源（临界资源的访问）



进程资源有向图

24.11

下列选项中不能作为预防死锁措施的是()。

- A、破坏“循环等待”条件
- B、破坏“不可抢占”条件
- C、破坏“互斥”条件
- D、破坏“请求和保持”条件

答案：C

● → □ 表示请求的资源
□ → ● 表示分配的资源

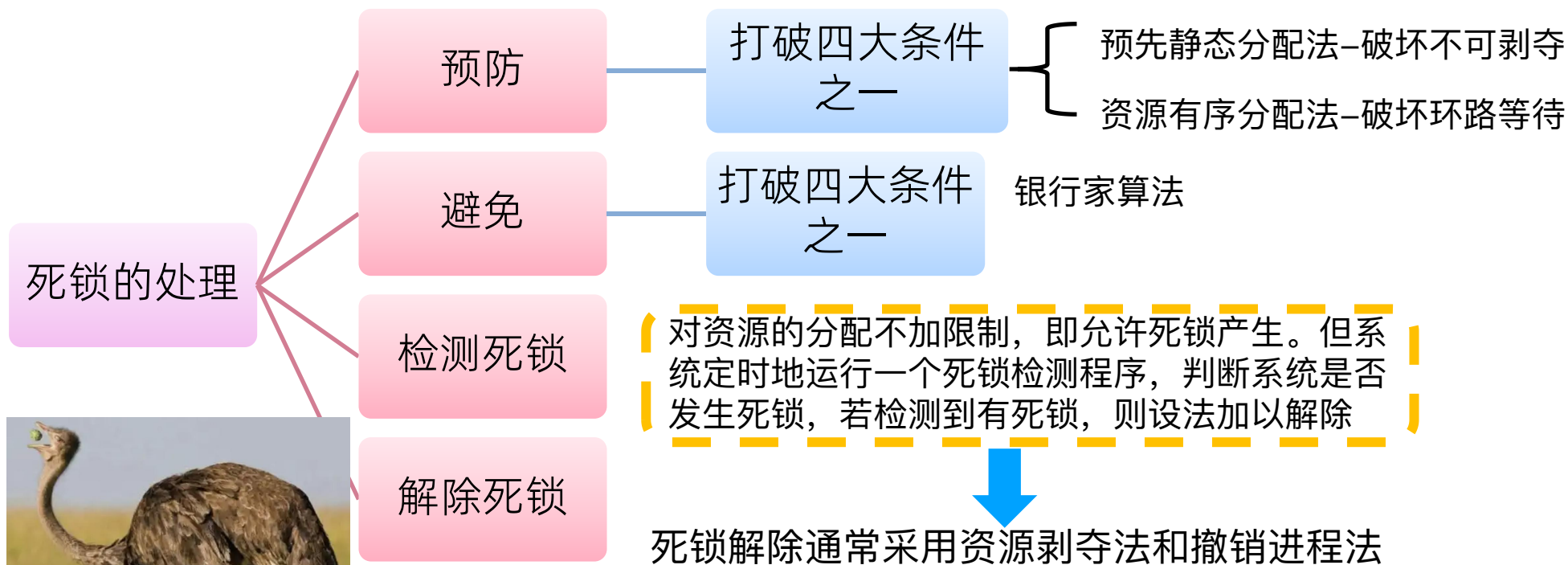
25.05 5个进程各需要2个资源，至少提供（）个资源才能避免死锁。

- A.5个 B.6个 C.8个 D.10个

【解析】：

R个进程，每个进程需要N个资源，避免死锁需要的资源数 = $R \times (N - 1) + 1 = 5 \times (2 - 1) + 1 = 6$ 。

进程管理-死锁



银行家算法:分配资源的原则

当一个进程对资源的最大需求量不超过系统中的资源数时可以接纳该进程；
进程可以分期请求资源，但请求的总数不能超过最大需求量；
当系统现有的资源不能满足进程尚需资源数时，对进程的请求可以推迟分配，但总能使进程在有限的时间里得到资源。

典型真题

银行家算法

假设系统中互斥资源R的可用数为25。T0时刻进程P1、P2、P3、P4 对资源R的最大需求数、已分配资源数和尚需资源数的情况如表a所示，若P1和P3 分别申请资源R数为1和2，则系统()。

- A.只能先给P1进行分配，因为分配后系统状态是安全的
- B.只能先给P3进行分配，因为分配后系统状态是安全的
- C.可以先后 P1、P3.进行分配，因为分配后系统状态是安全的
- D.不能给P3进行分配，因为分配后系统状态是不安全的

答案： B

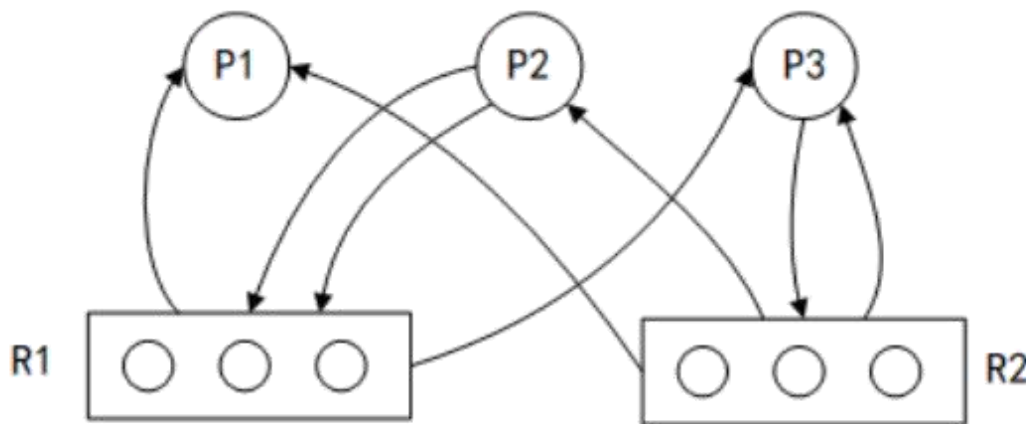
解析： 银行家算法。T0时刻系统可用资源数为 $25 - (6 + 4 + 7 + 6) = 2$ ，此时只能满足P3，因此必须第一个分配给P3。

表 A T0 时刻进程对资源的需求情况

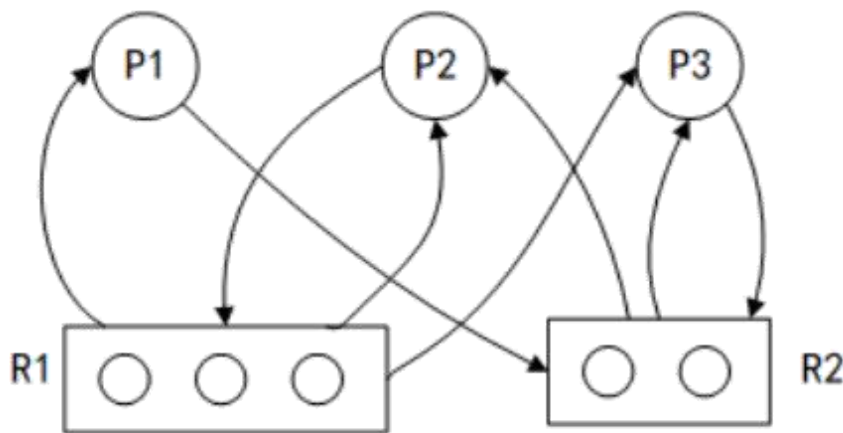
进程	最大需求数	已分配资源数	尚需资源数
P1	10	6	4
P2	11	4	7
P3	9	7	2
P4	12	6	6

典型真题

假设系统中有三个进程P1、P2和P3，两种资源R1、R2。如果进程资源图如图①和图②所示，那么（ ）。



图①



图②

A.图①和图②都可化简

B.图①和图②都不可化简

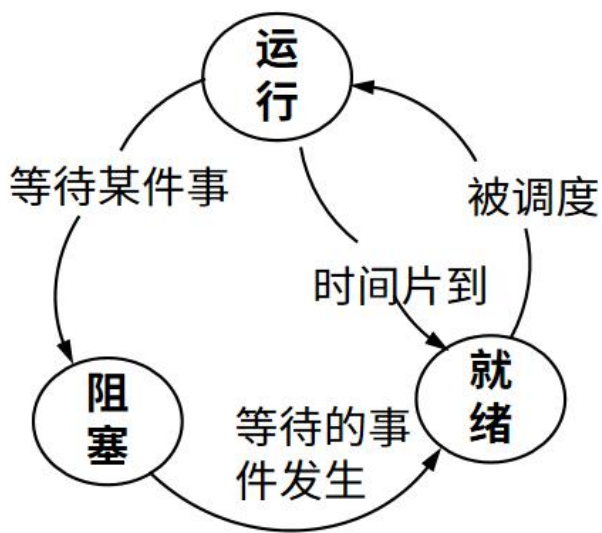
C.图①可化简，图②不可化简

D.图①不可化简，图②可化简

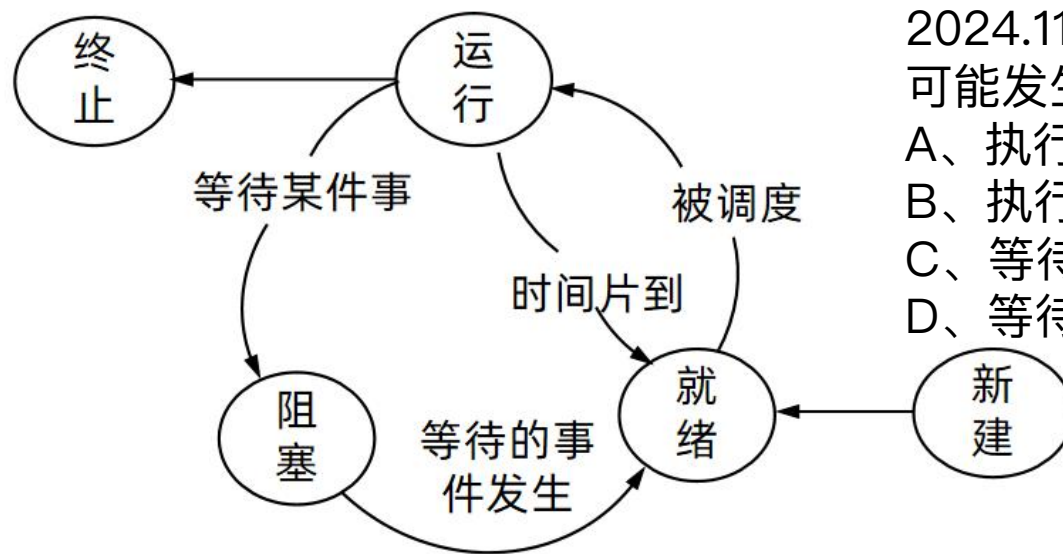
答案：C

进程管理-进程的状态及调度

- (1)运行。当一个进程在处理机上运行时，则称该进程处于运行状态。显然，对于单处理机系统，处于运行状态的进程只有一个。
- (2)就绪。一个进程获得了除处理机外的一切所需资源，一旦得到处理机即可运行
- (3)阻塞。阻塞也称等待或睡眠状态，一个进程正在等待某一事件发生(例如请求I/O 等待 I/O 完成等)而暂时停止运行，这时即使把处理机分配给进程也无法运行



进程的三态模型



进程的五态模型

2024.11以下进程状态变化中，不可能发生的是()

- A、执行状态到就绪状态
- B、执行状态到等待状态
- C、等待状态到执行状态
- D、等待状态到就绪状态

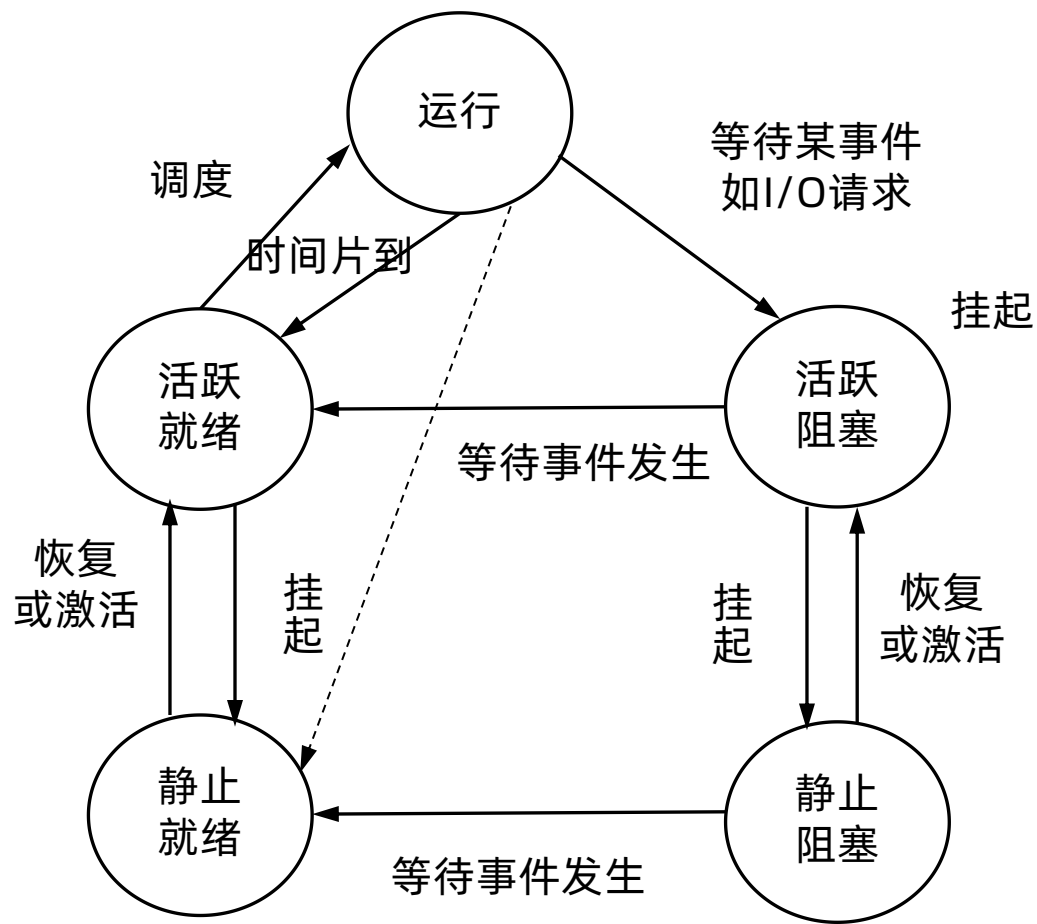
典型真题

在单处理机计算机系统中，有1台打印机、1台扫描仪，系统采用先来先服务调度算法。假设系统中有进程P1、P2、P3、P4，其中P1为运行状态，P2为就绪状态，P3等待打印机，P4等待扫描仪。此时，若P1释放了扫描仪，则进程P1、P2、P3、P4的状态分别为 (7) 。

- (7) A. 等待、运行、等待、就绪 B. 运行、就绪、等待、就绪
 C. 就绪、就绪、等待、运行 D. 就绪、运行、等待、就绪.

答案：D

进程管理-进程的状态及调度



- 优先级调度
- 抢占式&非抢占式

时间片轮转
先来先服务
短作业优先
高响应比优先

()进程调度算法每次从就绪队列中选择一个最先进入该队列的进程为其分配处理机，该进程一直运行到完成或发生阻塞后才放弃处理机。

- A、高响应比优先 B、时间片轮转
C、先来先服务 D、短作业优先

典型真题

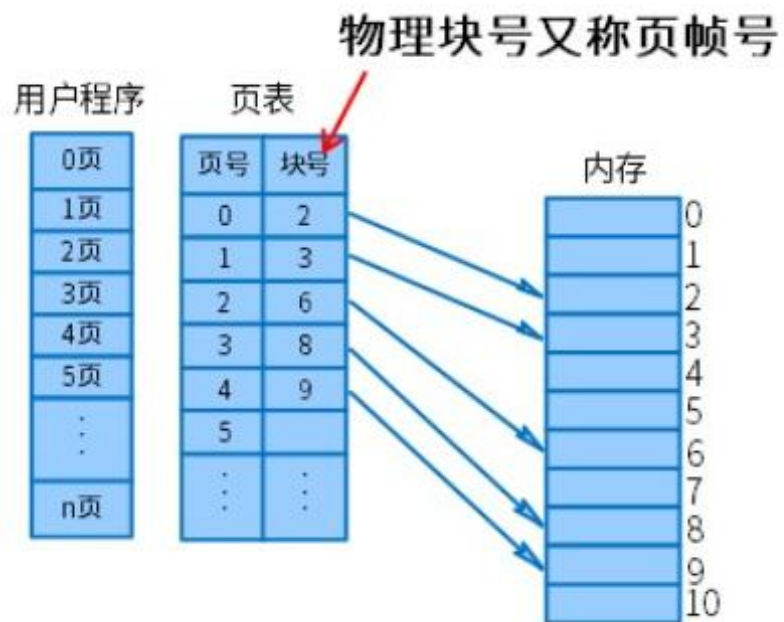
实时操作系统主要用于有实时要求的过程控制等领域。因此，在实时操作系统中，对于来自外部的事件必须在()。

- A. 一个时间片内进行处理
- B. 一个周转时间内进行处理
- C. 一个机器周期内进行处理
- D. 被控对象允许的时间范围内进行处理

【答案】 D

页式存储

页式存储：将程序与内存划分成同样大小的块，以**页为单位**将程序调入内存。



高级程序语言使用逻辑地址；
运行状态，内存中使用物理地址。

逻辑地址=页号+页内地址

物理地址=页帧号+页内地址

例如，页式存储系统中，每个页的大小为4KB。

逻辑地址：

10 1100 1101 1110

对应的物理地址为：

110 1100 1101 1110

优点：利用率高，碎片小，分配及管理简单

缺点：增加了系统开销，可能产生抖动现象。

典型真题

某计算机系统页面大小为4K，若进程的页面变换表如下所示，逻辑地址为十六进制1D16H。该地址经过变换后，其物理地址应为十六进制（ ）。

页号	物理块号
0	1
1	3
2	4
3	6

- A. 1024H B. 3D16H. C. 4D16H D. 6D16H

试题分析

页面大小为4K，而 $4K=2^{12}$ ，因此逻辑地址的低12位对应页内地址，高位对应页号。题目中逻辑地址为十六进制1D16H，一位十六进制数对应4位二进制数，3位十六进制数则对应12位二进制数，因此D16H为页内地址，页号为1。查页面变换表，页号1对应的物理块号为3，将物理块号与页内地址D16H拼接起来即可得到物理地址3D16H。

参考答案：B

典型真题

某操作系统采用分页存储管理方式，下图给出了进程A和进程B的页表结构。如果物理页的大小为512字节，那么进程A逻辑地址为1111(十进制)的变量存放在（ ）号物理内存页中。假设进程A的逻辑页4与进程B的逻辑页5要共享物理页8，那么应该在进程A页表的逻辑页4和进程B页表的逻辑页5对应的物理页处分别填（ ）。

进程 A 页表		进程 B 页表		物理页
逻辑页	物理页	逻辑页	物理页	0
0	9	0	1	1
1	2	1	3	2
2	4	2	5	3
3	6	3	7	4
4		4	2	5
5		5		6
				7
				8
				9

- A. 9 B. 2 C. 4. D. 6
- A. 4、5 B. 5、4 C. 5、8 D. 8、8.

典型真题

试题分析

十进制1111转化为二进制10001010111。物理页的大小为512字节,这说明页内地址为9个二进制位,进程A的逻辑址中,右边的9位是页内地址,左边的2位是页号,即:10 001010111。页号为二进制的10.即十进制的2,对应的物理页号为4.

若A页表的逻辑页4和进程B页表的逻辑页5共享物理页8,则说明他们都对应物理页8.所以均填8。

参考答案: C、D

页面淘汰算法问题

某系统采用请求页存式存储管理方案，假设某进程有6个页面，系统给该进程分配了4个存储块，其页面变换表如下图所示，表中的状态位等于1和0分别表示页面在内存或不在内存，当该进程访问的第4号页不在内存时，应该淘汰表中页面号为（）的页面。

页面号 逻辑	页帧号 (物理)	状态位 0不在内存; 1在内存	访问位 1为最近访问过; 0 为最近未被访问	修改位 1: 内容被修改过 0: 内容未被修改过
-----------	-------------	-----------------------	------------------------------	--------------------------------

0	—	0	0	0
1	5	1	1	1
2	6	1	1	1
3	8	1	0	1
4	—	0	0	0
5	12	1	1	0

淘汰原则

- n 首先淘汰未被访问过的页面（局部性原理）
- n 当都被访问过时，应淘汰未修改过的页面
- n 未修改的页面与辅存内容一致，所以代价较小

202505系分真题

假设某计算机主存的读/写时间为 100ns，有一个指令和数据合一的 Cache，已知该 Cache 的读 / 写时间为 20ns，取指令的命中率为 98%，取数的命中率为 95%。在执行某类程序时，约有 1/5 指令需要存 / 取一个操作数。假设指令流水线在任何时刻都不阻塞，则设置 Cache 后，每条指令的平均访存时间约为（ ）。

A. 26.4ns B. 32ns C. 14.7ns D. 18.6ns

答案：A

解析：取指令访问时间 $20 \times 98\% + 100 \times 2\%$ ，取数访问时间 $(20 \times 95\% + 100 \times 5\%) \times 1/5$ ，二者相加为 26.4ns

THANKS

 极客时间 | 训练营