我坚信，在考研过程中培养的品质，一定会在今后闪闪发辉

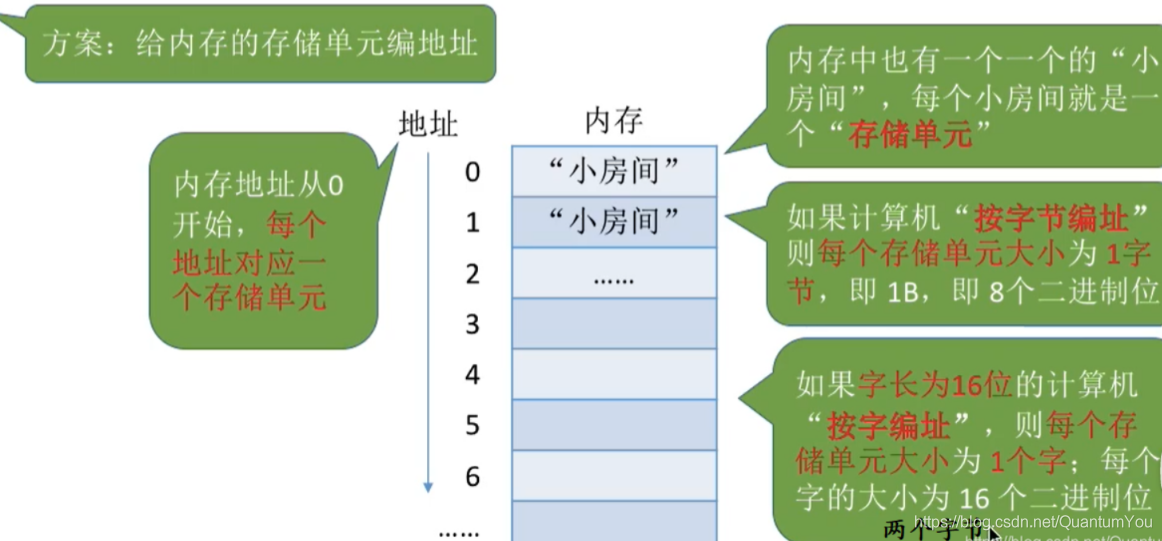
@[toc]

# 内存的基本知识

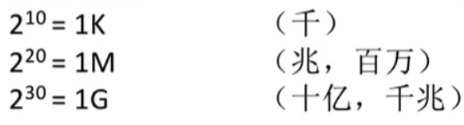
* 内存是用于存放数据的硬件。程序执行前需要先放到内存中才能被CPU处理。😇
* 在多道程序环境下，糸统中会有多个程序并发执行，也就是说会有多个程序的数据需要**同时**放到内存中。那么，如何区分各个程序的数据是放在什么地方的呢？

解法: 给内存的存储单元编地址

* 内存地址从0开始，每个地址对应一个存储单元
* 内存中有一个个“小房间”，这就是存储单元



* 根据内存大小确定地址的长度



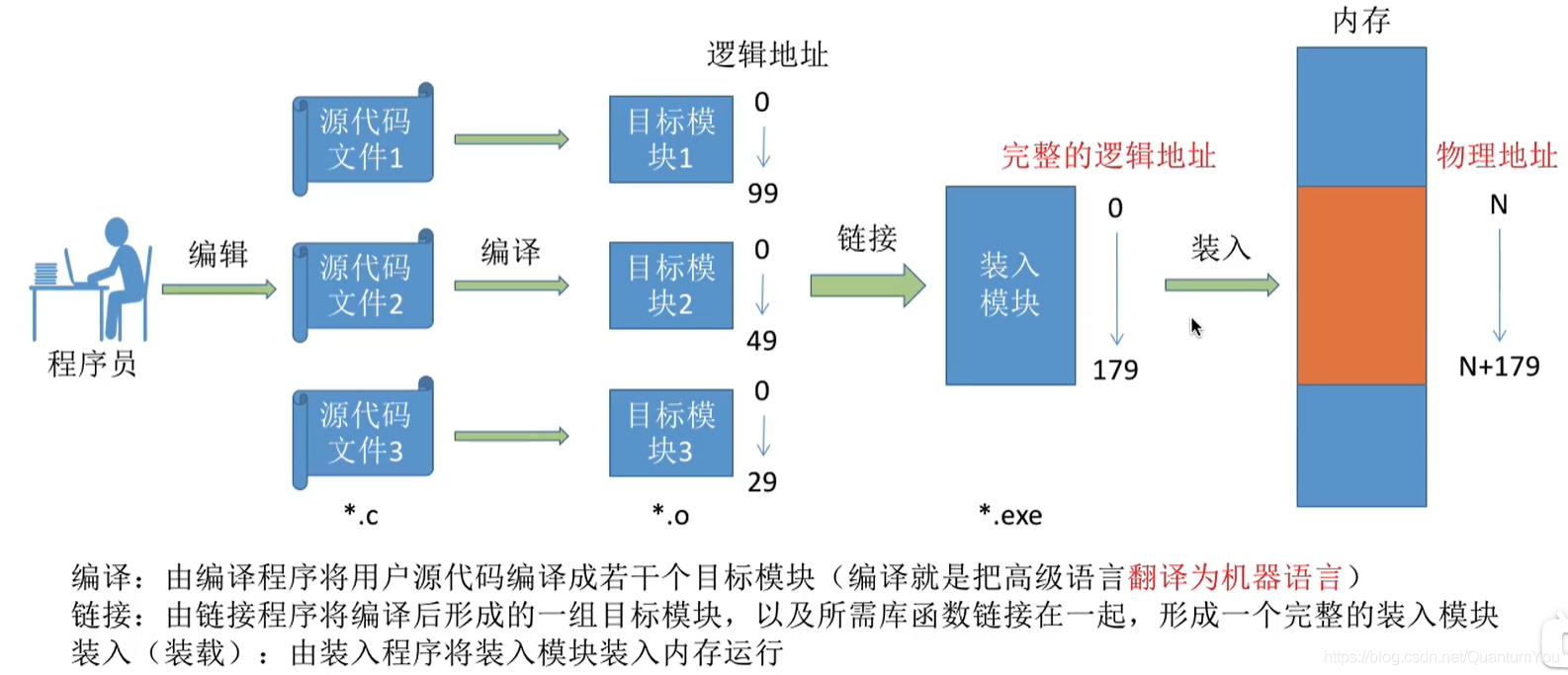
Eg: 4GB 内存 --> 存放4\* 230个字节 --> 2\* 232 个存储单元，所以地址需要32个二进制位进行表示（0~232-1）

回顾：

* 程序段（存放指令）、数据段（存放数据）、PCB
* 相对地址又称逻辑地址，绝对地址又称物理地址

## 写程序至程序运行

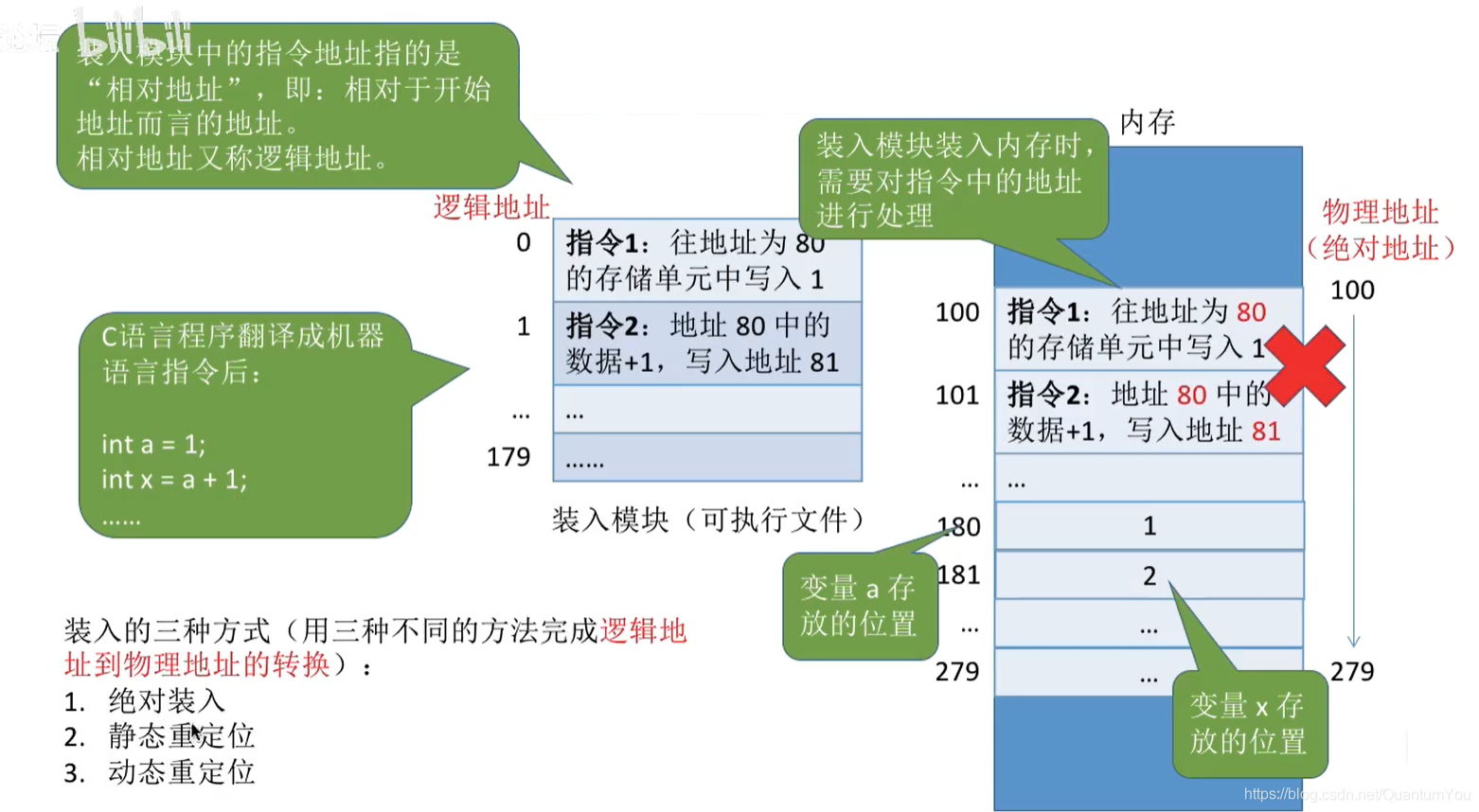
进程运行的基本原理

* 编译、链接、装入  
  

## 装入模块装入内存

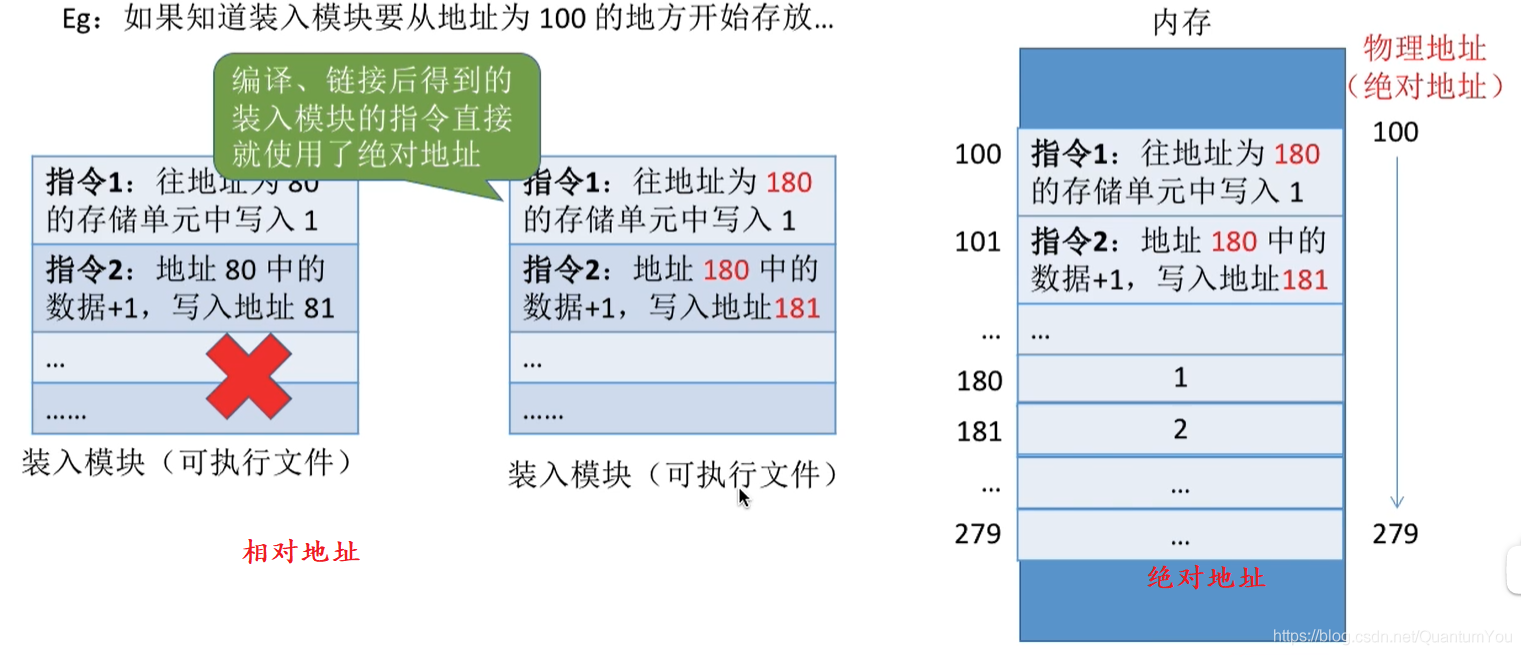
**问题引出**

* 逻辑地址与物理地址的对应关系



### 绝对装入

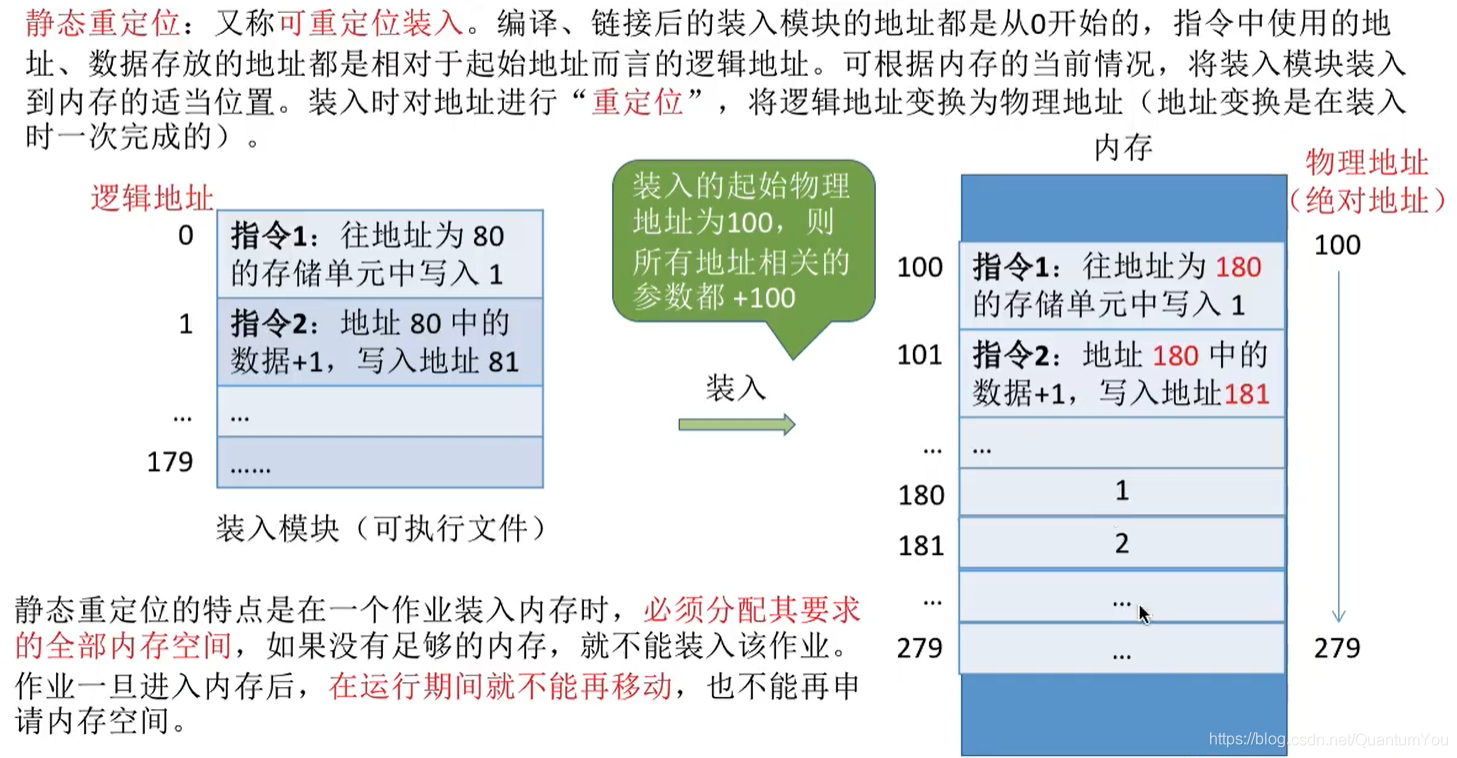
* 绝对装入：在编译时，如果知道程序将於到内存中的哪个位置，编译程序将产生绝对地址的目标代码。装入程序按照装入模块中的地址，将程序和数据装入内存。
* 绝对转入只适用于单道程序环境（同一时刻内存中只有一道程序运行）



* 程序中使用的绝对地址，可在编译或汇编时给出，也可由程序  
  员直接赋予。通常情况下都是编译或汇编时再转换为绝对地址

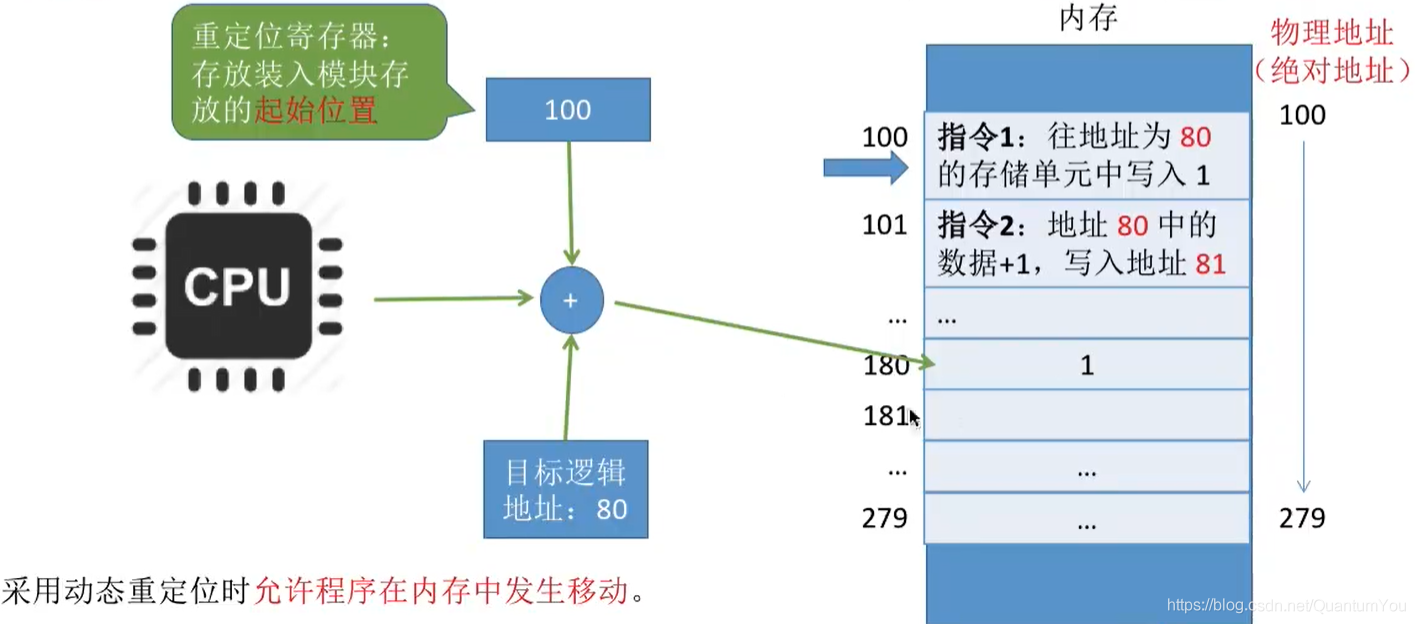
### 静态重定位

* 静态重定位：又称可重定位装入。编译、链接后的装入模块的地址都是从0开始的，指令中使用的地址、数据存放的地址都是相对于起始地址而言的逻辑地址。可根据内存的当前情况，将裝入模块装入到内存的适当位置。装入时对地址进行“重定位”，将逻辑地址变换为物理地址（地址变换是在装入时一次完成的）.



### 动态重定位

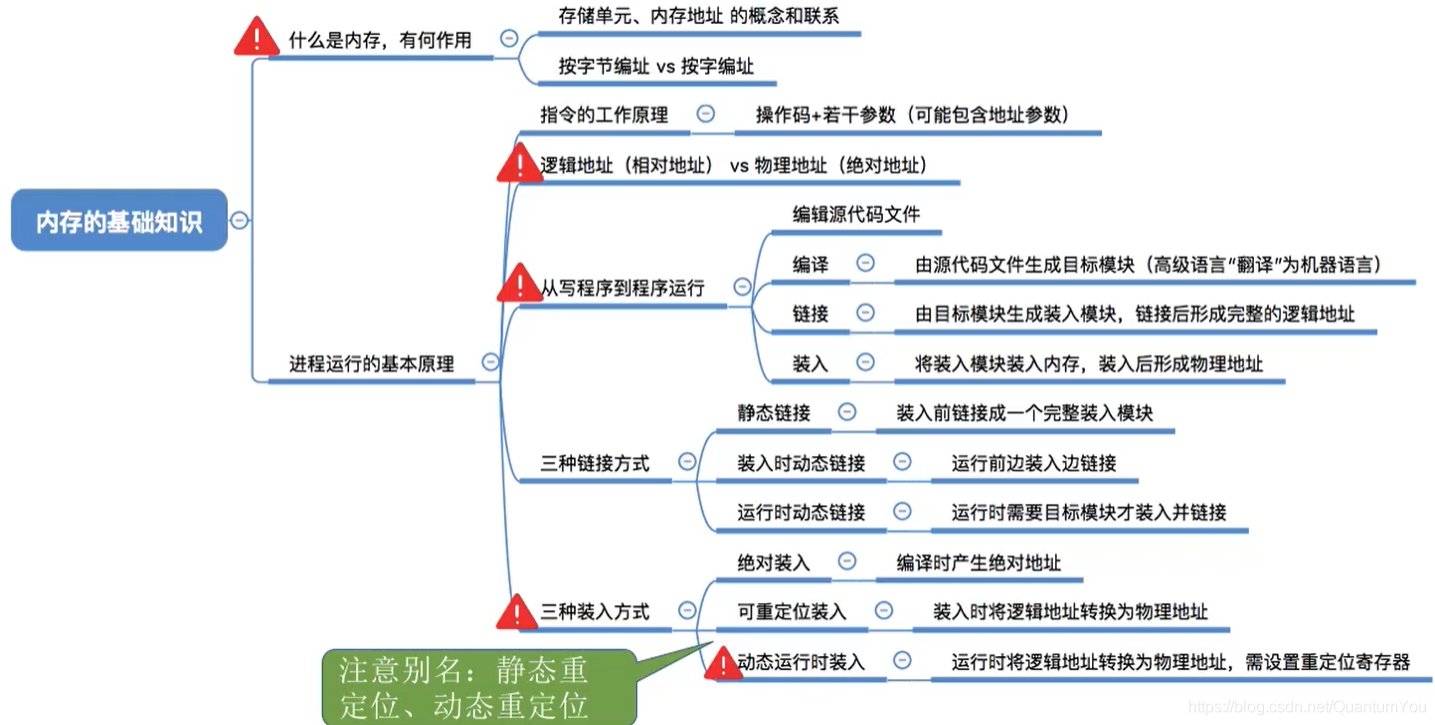
* 动态重定位：又称动态运行时装入。编译、链接后的装入模块的地址都是从0开始的。装入程序把装入模块裝入内存后，并不会立即把逻辑地址转换为物理地址，而是把地址转换推迟到程序真正要执行时才进行。因此装入内存后所有的地址依然是逻辑地址。这种方式需要一个重定位寄存器的支持。



**优点**

* 可将程序分配到不连续的存储区中；在程序运行前只需装入它的部分代码即可投入运行，然后在程序运行期间，根据需要动态申请分配内存；
* 便于程序段的共享，可以向用户提供一个比存储空间大得多的地空间。

## 小结



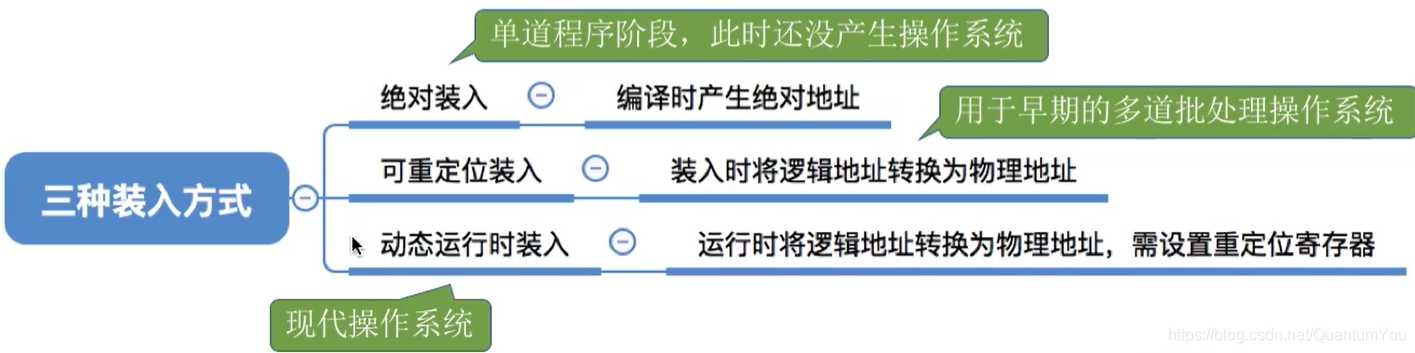
# 内存管理

## 内存空间的分配与回收

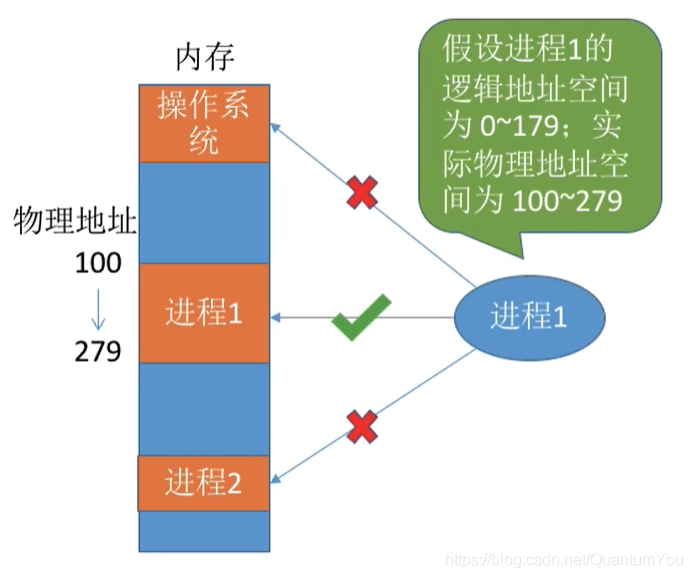
引入：操作系统作为系统资源的管理者，当然也需要对内存进行管理，要管些什么呢？

解法：

* 1、操作系统负责**内存空间的分配与回收**
* 2、操作系统需要提供某种技术从逻辑上**对內存空间进行扩充**
* 3、操作系统需要提供地址转换功能，负责程序的**逻辑地址**与**物理地址**的转换



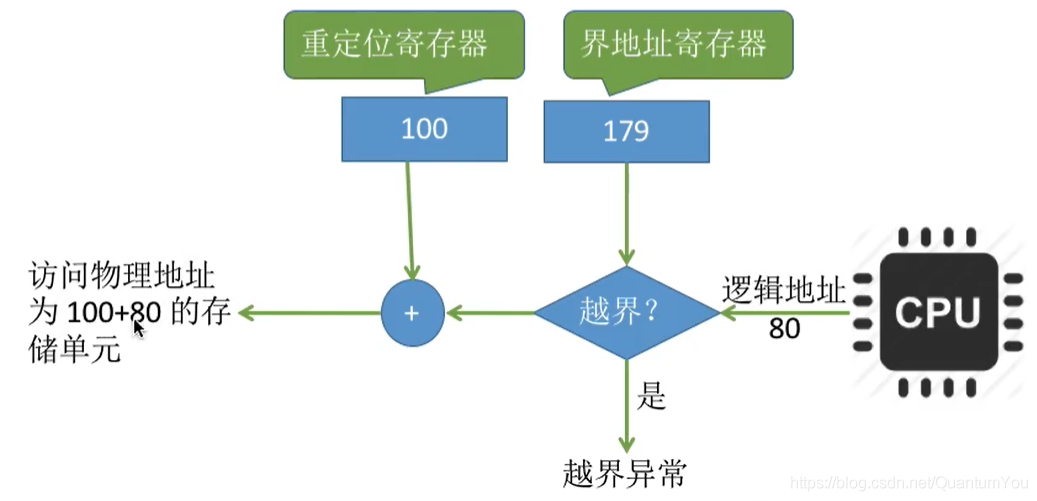
## 内存保护



* **方法一**：在CPU中设置一对上、下限寄存器，存放进程的上、下限地址。进程的指令要访问某个地址时，CPU检查是否越界。



* **方法二**：采用重定位寄存器（又称基址寄存器）和界地址寄存器（又称限长寄存器）进行越界检査。重定位寄存器中存放的是进程的起始物理地址。界地址寄存器中存放的是进程的最大逻辑地址。



## 小结

