第5章 内存管理

5.1 例题解析

例题 1 根据作业在本次分配到的内存起始地址将目标代码装到指定内存地址中，并修改所有有关地址部分的值的方法称为_____方式。

A. 固定定位　　B. 静态重定位　　C. 动态重定位　　D. 单一连续重定位

分析：答案 B。在可重定位装入方式中，如果逻辑地址转换成物理地址的过程发生在程序装入到内存时进行，在程序装入时一次完成地址转换称为静态重定位，如果地址转换过程是在指令运行过程中进行的称为动态重定位。单一连续分配是内存分配的一种管理方式，跟程序重定位无关。

例题 2 在下列存储管理算法中，内存的分配和释放平均时间之和为最大的是_____。

A. 首次适应法　　B. 循环首次适应法

C. 最佳适应法　　D. 最差适应法

分析：答案 C。最佳适应算法要找到一块既满足程序需要又是空闲区域中最小的空闲块，因此最佳适应算法的分配算法速度比首次适应法、循环首次适应法和最差适应法差得多，如用链表实现，释放算法要在链表中找上、下邻空闲区，修改过或新加入的空闲区还要有序地插入到链表中。

例题 3 以下分配方案中，_____不适于多道系统。

A. 单一连续区管理　　B. 固定分区管理

C. 可变分区管理　　D. 页式存储管理

分析：答案 A。采用单一连续分配管理方式时，内存的用户区一次只能分配给一个用户程序使用，因此无法满足多道程序同时运行。

例题 4 可变式分区又称为动态分区，它是在系统运行过程中_____时动态建立的。

A. 在作业未装入　　B. 在作业装入　　C. 在作业创建　　D. 在作业完成

分析：答案 B。在存储器管理技术中引入多道程序系统后，多个作业可以同时放在内存中，这就需要把存储器分成若干区域，每个区域分配给一道程序，这就是分区分配，分区管理分为固定式分区和可变式分区两种。在可变式分区管理中，当装入作业时从空闲区中划出一个与作业需求量相适应的分区分配给该作业，在作业运行完毕后，收回释放的分区。

例题 5 采用可重入程序是通过使用_____的方法来改善响应时间的。

A. 减少用户数目　　B. 减少对换信息量

C. 改变时间片长短　　D. 加快对换速度

分析：答案 B。所谓可重入程序，是指当多个用户共享程序时，在内存中仅保存一份副本。而在没有采用可重入代码的系统中，每个用户都必须各备一套程序。由于可重入代码的采用减少了内外存的对换信息量，这也就为采用更短的时间片来缩短响应时间创造了条件。

例题 6 下面内存管理方法中有利于程序动态链接的是_____。

A. 分段存储管理　　B. 分页存储管理

C. 可变分区分配　　D. 固定分区分配

分析：答案 A。引入分段存储管理方式，主要是为了满足用户和程序员的下述一系列需要：①方便编程；②信息共享；③信息保护；④动态增长；⑤动态链接。动态链接是指在作业运行之前，并不把几个目标程序段链接起来。要运行时，先将主程序所对应的目标程序装入内存并启动运行，当运行过程中又需要调用某段时，才将该段(目标程序)调入内存并进行链接。可见，以段作为管理单位有利于动态链接。

例题 7 什么是动态重定位？它有什么特点？

答：动态重定位是指在程序执行期间，由系统硬件完成从逻辑地址到物理地址的转换。

动态重定位的特点：具有给用户程序任意分配内存区的能力；可以将程序分配到不连续的存储区中；在程序运行之前可以只装入它的部分代码即可投入运行，不执行的代码就不做地址映射的工作，这样节省了 CPU 的时间；可实现虚拟存储，向用户提供一个比主存的存储空间大得多的地址空间；重定位寄存器的内容由操作系统用特权指令来设置，比较灵活；实现动态重定位必须有硬件支持，并有一定的执行时间延迟，且实现存储管理的软件算法比较复杂。现代计算机系统中都采用动态地址映射技术。

例题 8 某操作系统采用分区存储管理技术。操作系统在低地址占用了 100KB 的空间，用户区主存从 100KB 处开始占用 512KB。开始时，用户区全部空闲，分配时截取空闲分区的低地址部分作为已分配区。在执行申请、释放操作序列后，请求 300KB、请求 100KB、释放 300KB、请求 150KB、请求 50KB、请求 90KB，请回答以下问题。

(1) 若采用首次适应分配算法，此时主存中有哪些空闲分区？

(2) 若采用最佳适应分配算法，此时主存中有哪些空闲分区？

(3) 若随后又要请求 80KB，采用以上两种方法分别会产生什么情况？

答：(1) 采用首次适应分配算法。

请求 300KB；地址块 100～399 被占用；地址块 400～612 空闲。

请求 100KB；地址块 100～399、400～499 被占用；地址块 500～612 空闲。

释放 300KB；地址块 400～499 被占用；地址块 100～399、500～612 空闲。

请求 150KB；地址块 100～249、400～499 被占用；地址块 250～399、500～612 空闲。

请求 50KB；地址块 100～249、250～299、400～499 被占用；地址块 300～399、500～612 空闲。

请求 90KB；地址块 100～249、250～300、300～389、400～499 被占用；地址块 390～399、500～612 空闲。

产生空闲块两个：块 1，首地址 390，块大小为 10KB；块 2，首地址 500，块大小为 112KB。

(2) 采用最佳适应分配算法。

请求 300KB；地址 100～399 被占用；地址 400～612 空闲。

请求 100KB；地址 100～399、400～499 被占用；地址 500～612 空闲。

释放 300KB；地址 400～499 被占用；地址 100～399、500～612 空闲。

请求 150KB；地址 100～249、400～499；地址 250～399、500～612 空闲。

请求 50KB；地址 100～249、400～499、500～549 被占用；地址 250～399、550～612 空闲。

请求 90KB；地址 100～249、250～339、400～499、500～549 被占用；地址 340～399、550～612 空闲。

产生空闲块两个：块 1，首地址 340，块大小为 60KB；块 2，首地址 550，块大小为 62KB。

(3) 若随后又要请求 80KB，则采用首次适应分配算法可以分配，因为块 2 空闲空间为 112KB，大于 80KB；采用最佳适应分配算法不可分配，因为块 1 和块 2 空闲空间均小于 80KB。

例题 9 比较分别采用数组和链表两种数据结构实现最佳适应算法和最差适应算法的优缺点(要考虑分配和释放两个过程)。

答：实现最佳适应算法时，空闲存储区管理表的组织方法可以采用顺序结构，也可采用链接结构。如采用顺序结构，空闲分区按地址由小到大的顺序登记在表中，分配时需要搜索所有的空闲分区，在其中挑出一个满足分配大小的最小分区，其算法的时间复杂度为 $O(N)$。此种管理结构的释放算法可用顺序结构的首次适应法，不需要插入或删除一个空闲分区表项时，其时间复杂度为 $O(1)$，否则其算法的时间复杂度为 $O(N)$。

当采用链接结构时，空闲区也可按由小到大的非递减次序排列。分配时总是从最小的第一项开始，这样第一次找到的满足条件的空闲区必定是最合适的。平均而言，只要搜索一半数目的空闲区表项就能找到最佳配合的空闲区，但寻找较大空闲区比较费时，其算法的时间复杂度为 $O(N)$。采用按存储区大小排序的链接表会降低释放算法的效率。由于空闲区是按大小而不是按地址序号排序的，因此释放回收空闲区时要在整个链表上寻找地址相邻的前、后空闲区，合并后又要插入合适的位置，因此释放算法比首次适应法和循环首次适应法耗时得多，尽管其算法的时间复杂度也为 $O(N)$，但其常数 C 要大得多。

实现最差适应算法时的空闲存储区表的组织方法一般都是采用按空闲块由大到小排序的链接表，因为如果采用按地址大小的顺序结构，那么该算法与首次适应法和最佳适应法比较起来就没有什么优点可言了。采用按存储区大小顺序排列的链表形式，虽然释放一个空闲块时速度较慢，算法的时间复杂度也为 $O(N)$，但分配时一次查找就行，成功不成功在此一举，算法的时间复杂度为 $O(1)$，其效率是一切算法中最高的一种，很适合实时系统。

例题 10 用可变分区分配的存储管理方案中，基于链表的存储分配算法有哪几种？它们的思想是什么？

答：可变分区分配的存储管理方案中，基于链表的存储分配算法主要有三种：首次适应分配算法、循环首次适应分配算法和最佳适应分配算法。

(1) 首次适应分配算法。

在首次适应算法中，每个空白区按其位置的顺序链接在一起，即每个后继的空白区其起始地址总是比前者的大。当系统要分配一个存储块时，就按照空白块链的顺序，依次查询，直到找到第一个满足要求的空白块为止。由这种算法确定的空白块，其大小不一定刚好满足要求。如果找到的这个空白区比要求的大，则把它分成两个分区，一个为已分配分区，其

大小刚好等于所要求的,另一个仍然为空白块,且留在链中原来的位置上。如果在空白链中从头到尾找了一遍,找不到满足要求的空白块,则返回“暂不能分配”。系统回收一个分区,首先检查该分区是否有邻接的空白块,如果有,则应将这个分区与之合并,并将该空白块保留在链中原来的位置;如果回收的分区不和空白块邻接,则应根据其起始地址大小,把它插在链中的相应位置上。

首次适应分配算法的实质是,尽可能地利用存储器低地址部分的空白块,尽量保存在高地址部分的大空白块。其优点在于:当需要一个较大的分区时,便有希望找到足够大的空白块以满足要求。其缺点是:在回收一个分区时,需要花费较多的时间去查找链表,确定它的位置。

(2) 循环首次适应分配算法。

循环首次适应分配算法与首次适应分配算法类似,只是在每次分配分区时,系统不是从第一个空白块开始查找,而是从上次分配的空白块处查找。当查找至链尾时,便从链首继续查找,直到查找完整个链表。在系统回收一个分区时,为了减少在插入一个空白区时花在查寻链表的时间,如果这个分区不和空白块邻接,则把它插入到前向指针链的最后一个空白块后;如果和空白块相邻,则根据情况做相应处理。由此可见,这些空白块在链中的位置没有一定的规则。

这种循环首次适应分配算法的实质是,使得小的空白块均匀地分布在可用存储空间内。这样,当回收一个分区时,它和一个较大空白块相邻的可能性比较大,因而合并后可得到更大的空白块。与首次适应算法相比,它产生过小空白块现象的可能性比较小。

(3) 最佳适应分配算法。

在最佳适应分配算法中,空白块按大小顺序链接在一起。系统在寻找空白块时,总是从最小的一个开始。这样,第一次找到的满足要求的空白块必然是最适合的,因为它最接近于要求的大小。这种算法的优点是:如果存储空间中具有正好是所要求大小的空白块,则必然被选中;如果不存在这样的空白块,也只对比要求稍大的空白块划分,而绝不会去划分一个更大的空白块。此后,遇到大的存储要求时,就比较容易满足了。最佳适应算法的缺点在于:寻找一个较大空白块时花费的时间较长;回收一个分区时,把它插入到空白块链中合适的位置上也较为费时;此外,由于每次都划分一个与要求大小最接近的空白块,系统中小的空白块较多。这种算法的实质是,在系统中寻找与要求的空间大小最接近的一个空白块。

5.2 课后自测题

一、选择题

1. 对内存的访问是以_____为单位。

A. 字节或字　　B. 页　　C. 段　　D. 内存块

2. 将逻辑地址转变为内存的物理地址的过程称为_____。

A. 编译　　B. 连接　　C. 运行　　D. 重定位

3. 如果一个程序为多个进程所共享,那么该程序的代码在执行过程中不能被修改,即程序应该是_____。

A. 可执行码　　B. 可重入码　　C. 可改变码　　D. 可再现码

4. 源程序经过编译或者汇编生成的机器指令集合，称为_____。

A. 源程序　　B. 目标程序　　C. 可执行程序　　D. 非执行程序

5. 装入到地址寄存器的地址为_____。

A. 符号名地址　　B. 虚拟地址　　C. 相对地址　　D. 物理地址

6. 动态重定位是在程序的_____中进行的。

A. 编译过程　　B. 连接过程　　C. 装入过程　　D. 执行过程

7. 静态地址重定位的结果是得到_____。

A. 源程序　　B. 静态代码　　C. 目标代码　　D. 执行代码

8. 静态地址重定位的对象是_____。

A. 源程序　　B. 编译程序　　C. 目标程序　　D. 执行程序

9. 下面几条中，_____是动态重定位的特点。

A. 需要一个复杂的重定位装入程序　　B. 存储管理算法比较简单

C. 不需地址变换硬件机构的支持　　D. 在执行时将逻辑地址变换成内存地址

10. 采用动态重定位技术优点之一是_____。

A. 在程序执行期间可动态地变换映像在内存空间的地址

B. 程序在执行前就可决定装入内存的地址

C. 能用软件实现地址变换

D. 动态重定位的程序占用的内存资源较少

11. 使用_____，目标程序可以不经过任何改动而装入主存直接执行。

A. 静态重定位　　B. 动态重定位　　C. 编译或汇编　　D. 连接程序

12. 固定分区存储管理一般采用_____进行主存空间的分配。

A. 首次适应分配算法　　B. 循环首次适应分配算法

C. 最佳适应分配算法　　D. 顺序分配算法

13. 在可变分区存储管理中，回收一个空闲区后，空闲区管理表中不可能_____。

A. 增加一个表项　　B. 减少一个表项

C. 表项数不变　　D. 表项内容不变

14. 在可变式分区存储管理中，当释放和回收一个空闲区时，造成空闲表项区数减 1 的情况是_____。

A. 无上邻空闲区，也无下邻空闲区　　B. 有上邻空闲区，但无下邻空闲区

C. 无上邻空闲区，但有下邻空闲区　　D. 有上邻空闲区，也有下邻空闲区

15. 在可变分区式存储管理中，倾向于优先使用低地址部分空闲区的算法是_____。

A. 首次适应法　　B. 循环首次适应法

C. 最佳适应法　　D. 最差适应法

16. 在以下存储管理方案中，不适用于多道程序设计系统的是_____。

A. 单一连续区分配　　B. 固定式分区分配

C. 可变式分区分配　　D. 页式存储管理

17. 在固定分区分配中，每个分区的大小可以_____。

A. 相同　　B. 随作业长度变化

C. 不同但预先固定　　D. 不同但根据作业长度固定

18. 以下分配方案中,_____不适于多道系统。

A. 单一连续区管理　　B. 固定分区管理

C. 可变分区管理　　D. 页式存储管理

19. 最佳适应分配算法的空白区一般是_____。

A. 按大小递减顺序连在一起　　B. 按大小递增顺序连在一起

C. 按地址由小到大排列　　D. 按地址由大到小排列

20. 首次适应分配算法的空闲区一般是_____。

A. 按地址递增顺序连在一起　　B. 始端指针表指向最大空闲区

C. 按大小递增顺序连在一起　　D. 寻找从最大空闲区开始

21. 在可变分区管理方式下,在释放和回收空闲区时,若已判定"空闲区表第 j 栏中的始址=释放的分区始址+长度",则表示_____。

A. 归还区有上邻空闲区　　B. 归还区有下邻空闲区

C. 归还区有上下邻空闲区　　D. 归还区无相邻空闲区

22. _____方案要求程序在主存必须连续存放。

A. 可变分区存储管理　　B. 页式存储管理

C. 段式存储管理　　D. 段页式存储管理

二、填空题

1. 存储管理的主要功能是:主存空间的分配与回收,_____,主存空间的共享和_____,以及_____。

2. 程序员编写程序时所使用地址称为_____或称为_____。

3. 内存分配主要通过两种途径来实现,分别是_____和_____。

4. 将逻辑地址转换成物理地址的工作称为_____。

5. 源程序的名地址与目标程序的逻辑地址的转换是在_____过程中实施的。

6. 由装入程序实施的程序逻辑地址与物理地址转换的地址重定位方式称为_____。

7. 在执行时要靠硬件机构实现地址变换的方法是_____。

8. 系统初始化时就把存储空间划分成若干个分区的存储管理系统称为_____管理系统。

9. 在采用首次适用策略的可变分区存储管理中,某作业完成后要收回其主存空间并修改空闲区表。使空闲区始址不改变,空闲区数也不变的情况是_____。

10. 将内存中可分配的用户区预先划分为数量固定并且大小固定的分区,这种存储管理方案称为_____。

11. 在_____方式中,内存不预先划分,而是按照作业大小来划分分区,但分区的大小、位置和划分时间都是动态的,当作业装入时,根据作业的需求和内存空间的使用情况来决定是否分配。

12. 推迟到进程执行时才进行的地址重定位,称为_____。

13. 固定分区存储管理中的地址重定位方式主要采用_____。

14. 在可变分区存储管理中,主要采用_____重定位方式。

15. 存储保护主要包括防止_____和防止_____两方面的内容。

16. 可变分区存储管理中的地址转换需要硬件地址映射机制的支持,主要采用一对寄

存器，_____和_____。其中，前者用来存放作业或进程所占分区的起始地址；后者用来存放作业或进程所占连续存储空间的长度。

17. 可变分区存储管理中的地址转换采用的机制是：用户程序运行时，每访问一次_____，该机制就将逻辑地址与_____中的值进行比较，如果越界，则终止该进程；否则，与_____中的值相加得到物理地址。

18. 常用的可变分区分配算法中，_____算法有利于大作业的装入，但会使主存低地址和高地址两端的分区利用不均衡；_____算法会使分割后的空闲区不会太小，有利于中小型作业的装入。

19. 固定分区管理中，分区中未被利用的一部分区域称为_____；可变分区管理中，因为分割太小而无法利用的空闲分区称为_____。

三、问答题

1. 什么是符号名地址、相对地址和绝对地址？什么是地址重定位？

2. 地址重定位方式分为哪几种？各具有什么特点？

3. 简述固定分区存储管理的基本思想。

4. 简述可变分区存储管理算法中首次适应法的释放算法，其空闲存储区表是用连续线性结构实现的。

5. 简述可变分区存储管理中最佳适应算法的分配算法和释放算法。

6. 什么是内部碎片和外部碎片？各种主要的存储管理方法可能产生何种碎片？

7. 用户程序的地址结构只能是一维的吗？

5.3 自测题答案及分析

一、选择题

1. A　2. D　3. B　4. B　5. D　6. D　7. D　8. C　9. D　10. A

11. B　12. D　13. D　14. D　15. A　16. A　17. C　18. A　19. B　20. A

21. B 分析："空闲区表第 j 栏中的始址＝释放的分区始址＋长度"，释放的分区始址作为起始地址说明释放区在前面，释放区与它后面的空闲块合并。

22. A 分析：可变分区存储管理每次按程序需要的内存大小为其分配一片连续的存储空间。

二、填空题

1. 地址重定位　保护　内存空间的扩充
2. 逻辑地址　相对地址
3. 静态分配　动态分配
4. 地址重定位(或地址转换、地址映射)
5. 编译
6. 静态重定位
7. 动态重定位
8. 固定分区
9. 释放区与前空闲区相邻
10. 固定分区存储管理
11. 可变分区存储管理
12. 动态重定位
13. 静态重定位
14. 动态
15. 地址越界　非法操作(或操作越权)
16. 基址寄存器　限长(长度)寄存器
17. 内存　限长(长度)寄存器　基址寄存器

18. 首次适应分配算法　最坏适应分配算法

19. 内部碎片　外部碎片

三、问答题

1. **答**：对程序员来说，数据的存放地址是由变量符号决定的，因此称符号名地址或简称名地址，源程序的地址空间称为符号名空间或简称名空间。源程序经汇编或编译后得到的是目标代码程序，由于编译程序无法确定目标代码在执行时所驻留的实际内存地址，因此一般总是从零号单元开始为其编址，并顺序分配所有的符号名所对应的地址单元。由于目标代码中所有的地址值都相对于以"0"为起始的地址，而不是真实的内存地址，因此称这类地址为相对地址、逻辑地址或虚拟地址。

当装入程序将可执行代码装入内存时，程序的逻辑地址与程序在内存的物理地址是不相同的，必须通过地址转换将逻辑地址转换成内存地址，该地址称为绝对地址、物理地址或实地址，这个地址转换过程称为地址重定位。

2. **答**：地址重定位有静态重定位方式和动态重定位方式。当需要执行时，由装入程序运行重定位程序模块，根据作业在本次分配到的内存起始地址，将可执行目标代码装到指定内存地址中，并修改所有有关地址值的方式称为静态地址重定位。修改的方式是对每一个逻辑地址的值加上内存区首地址(或称基地址)值。静态重定位方式的主要优点是无须硬件地址变换机构支持，因此可以在一般的计算机上实现。其主要缺点是必须给作业分配一个连续的存储区域，在作业的执行期间不能扩充存储空间，也不能在内存中移动，多个作业也难以共享内存中同一程序副本和数据。

采用动态重定位方式，将程序装入内存时，不必修改程序的逻辑地址值，程序执行期间在访问内存之前，再实时地将逻辑地址变换成物理地址。动态重定位要靠硬件地址变换机构实现：当程序开始执行时，系统将程序在内存的起始地址送入地址变换机构中的基地址寄存器(BR)中。在执行指令时，若涉及逻辑地址，则先将该地址送入虚地址寄存器(VR)，再将BR和VR中的值相加后送入地址寄存器(MR)，并按MR中的值访问内存。

动态重定位方式的优点是程序在运行期间可以换出和换进内存，以便缓和内存紧张状态；也可在内存中移动，把内存中的碎片集中起来，以充分利用内存空间，这也便于进行多道程序设计。采用动态重定位方式，系统不必给程序分配连续的内存空间，这样就可将程序分成较小的部分，能充分利用内存中的较小片段。动态重定位方式又为信息共享和虚拟存储器的实现创造了条件。

3. **答**：固定分区管理系统在系统初始化时就把存储空间划分成若干个分区，这些分区的大小可以不同，以支持不同的作业对内存大小需求的不同。系统要建立一个分区说明表，用以记录每个分区的大小 、起始地址和状态等信息。当一个作业需要装入内存运行时，系统就从分区表中找出一个最合适的分区分配给该作业。

4. **答**：根据释放区与原空闲区相邻情况可归纳为4种情况。

(1) 仅与前空闲区相连：合并前空闲区和释放区，该空闲区的m_addr仍为前空闲区的首地址，修改表项的长度域m_size为前空闲区与释放区长度之和。

(2) 与前空闲区和后空闲区都相连：将三块空闲区合并成一块空闲区。修改空闲区表中前空闲区表项，其始地址为前空闲区始址，其大小m_size等于三个空闲区长度之和，这块大的空闲区由前空闲区表项登记。接下来还要在空闲区表中删除后项。

(3) 仅与后空闲区相连：与后空闲区合并，使后空闲区表项的 m_addr 为释放区的始址，m_size 为释放区与后空闲区的长度之和。

(4) 与前、后空闲区皆不相连：在前、后空闲区表项中间插入一个新的表项，其 m_addr 为释放区的始址，m_size 为释放区的长度。

5. **答**：采用最佳适应算法，空闲存储区管理表的组织方法可以采用顺序结构，也可采用链表结构。如采用顺序结构，空闲分区按地址由小到大的顺序登记在表中，分配时需要搜索所有的空闲分区，以在其中挑出一个满足分配需求的最小分区。此种管理结构的释放算法可采用顺序结构的首次适应法。

当采用链表结构时，空闲区可按由小到大的非递减次序排列。分配时总是从最小的第一项开始，这样第一次找到满足条件的空闲区必定是最合适的。采用按存储区大小排序的链表会降低释放算法的效率。由于空闲区是按空间大小而不是按地址排序的，因此释放回收空闲区时要在整个链表上寻找地址相邻的前、后空闲区，合并后又要插入到合适的位置，因此释放算法比首次适应法和循环首次适应法耗时得多。

6. **答**：当一个程序分配到比它所要求的更大的内存块时，剩余的空间没有利用，而其他程序也不能使用该内存空间，这就是内部碎片。

另一方面，在各个被分配出去的分区之间也会存在很多小的空闲区，其他程序很难利用该小内存空间，这就是“外部碎片”。

固定分区存储管理会产生内部的剩余碎片。在可变分区管理算法中，系统按照程序的申请大小分配内存，不会产生内部碎片，但在各个已分配出去的内存分区之间会留下一些小的分区，由于作业在主存要占用一个连续的存储区，当一个作业的程序空间大于主存中这些小的空闲存储区时，就不能装入运行，会产生外部碎片。

在分页存储管理中，内存划分为与程序虚页相同大小的块，程序的虚页可以占用主存中任意的内存块，故不会产生外部碎片，但程序所要求的内存不一定是整数块，最后一块可能没有全部利用，就会产生内部碎片，但内部碎片较小，平均来说只有半块大小。在纯段式管理系统中，作业需要分配一块连续的内存，与可变分区管理算法一样，会产生外部碎片。

在段页式存储管理系统中，由于段内再分为页，故与页式存储管理系统类似，但每一个段内都会产生一个页内碎片。

7. **答**：用户程序的结构可以是一维空间(一个用户程序就是一个程序，并且程序和数据是不分离的)，也可以是二维空间(程序由主程序和若干个子程序或函数组成，并且程序与数据是分离的)，还可以是 N 维空间(一个大型程序，由一个主模块和多个子模块组成，其中的各子模块又由主程序和子程序或函数组成)。