一、数据结构部分

1、绪论

（1）了解时间复杂度、空间复杂度，理解怎么表示。这两个知识点考查体现在算法设计中，算法设计的过程中要考虑时间复杂度和空间复杂度。

2、线性表

（1）顺序表、单链表的优缺点比较。顺序表就直观地理解为一维数组。
（2）顺序表考查的更多是算法的思想。比如求第 k 大的，连续子数组的和的最大值。做这类题目时，告诉了你顺序表，就要知道是已知了顺序表的长度。
（3）单链表注意带头结点和不带头结点，看清题目。一定要掌握单链表的创建、查找、删除的代码。一定要会两个数的交换代码，类似于单链表的转置代码。
（4）注意题目中的循环单链表，双链表，循环双链表，要充分利用这些链表的特性。否则题目做不出来，这些链表的特点是：给定链表中的任意一个结点的指针，可以遍历到链表中的任何一个结点。
（5）算法题中要注意元素的顺序，递增还是递减，要充分利用这些条件。有递增递减时，就要考虑到是否要定义两个指针（标志），从前往后，从后往前。
（6）算法题看清题目注意是否要用到递归。如果是单链表，很有可能是递归。链表的题目在解答时，首先写出链表的结构体定义，然后画图分析，谨记，一定要画图。
（7）本质：顺序表就是数组（操作），这里涉及到排序。链表就是指针的变化，还是画图。

3、栈和队列

（1）理解栈和队列的性质及异同点。
（2）进栈出栈的代码要会写。栈的考查主要体现在算法的递归中，因为递归运

用了栈的思想。

（3）进队出队的代码要会写。理解循环队列，溢出，判断空等条件。双端队列，受限的双端队列。队列的应用：树的层次遍历，图的广度优先遍历（掌握）。

4、串

（1）理解串的模式匹配，最简单的算法的思想，非 KMP。

5、数组和广义表

（1）理解多维数组的行存储，列存储。

（2）矩阵的压缩存储、特殊矩阵、稀疏矩阵；广义表不考。

（3）数组这一章的知识点考查在顺序表中体现，特别是一维数组。

6、树与二叉树

（1）树的定义、二叉树的性质、完全二叉树、满二叉树

（2）二叉树的遍历算法，特别是递归代码必须要会写。求二叉树的高度，叶子结点数等都是基于遍历算法模板来实现的。

（3）二叉树的层次遍历算法要会写。运用了队列。

（4）最优二叉树-赫夫曼树的定义，Huffman 树的构造，带权路径长度。

（5）二叉排序树及平衡二叉树的定义，如何构建二叉排序树，平衡二叉树。

（6）在考查二叉树的算法题目时，首先也要写出二叉树的结构体定义，然后画图分析。二叉树的定义本身是递归的，递归思想要会。考查的代码基本是在二叉树的遍历算法代码模板基础上更改而来。

7、图

（1）图的基本概念，有向图无向图中的概念，如连通图，强连通图等。

（2）图的存储结构，邻接矩阵邻接表表示的图的结构体代码必须理解掌握。掌握了基本的结构体代码，才能更好的理解图的遍历算法。

（3）图的遍历算法：深度优先及广度优先代码必须要掌握。

（4）最小生成树 prim 算法要理解，什么情况下的无向图能构造成一颗唯一的最

小生成树。

（5）最短路径 Dijstrala 算法理解，好久没考了，这个估计 19 年很有可能考。

（6）拓扑排序 AOV 网，理解拓扑排序的概念。

8、查找

（1）哈希表的构造，冲突处理方法，哈希因子。好久没考大题了，重点注意。

（2）二叉排序树，平衡二叉树，二分查找（折半查找）。

9、排序

（1）重点掌握快速排序，堆排序，归并排序，基数排序，希尔排序。这几种排序轮着考，要掌握。

（2）理解各个排序算法的时间复杂度，空间复杂。

10、例题及相关代码

（1）假设有两个按元素值递增次序排列的线性表，均以单链表形式存储。请编写算法将这两个单链表归并为一个按元素值递减次序排列的单链表，并要求利用原来两个单链表的结点存放归并后的单链表。

【解答】因为两链表已按元素值递增次序排列，将其合并时，均从第一个结点起进行比较，将小的插入链表中，同时后移链表工作指针。该问题要求结果链表按元素值递减次序排列。故在合并的同时，将链表结点逆置。

```
LinkedList Union(LinkedList la,lb)
{
    pa=la->next; pb=lb->next; //pa，pb 分别是链表 la 和 lb 的工作指针
    la->next=null;              //la 作为结果链表的头指针，先将结果链表初始化为空。
    while(pa!=null && pb!=null) //当两链表均不为空时作
    {
     if(pa->data<=pb->data)
      {
```

```
            r=pa->next;              //将 pa 的后继结点暂存于 r。
             pa->next=la->next;      //将 pa 结点插入结果表中，同时逆置。
            la->next=pa;
             pa=r;                   //恢复 pa 为当前待比较结点。
        }
         else
         {
           r=pb->next;// 将 pb 的后继结点暂存于 r。
           pb->next=la->next; //将 pb 结点插入结果表中，同时逆置。
           la->next=pb;
           pb=r; //恢复 pb 为当前待比较结点。
         }
      }
      while(pa!=null) //将 la 表的剩余部分链入结果表，并逆置。
        {r=pa->next; pa->next=la->next; la->next=pa; pa=r; }
      while(pb!=null)
        {r=pb->next; pb->next=la->next; la->next=pb; pb=r; }
}
```

（2）已知一个单链表中每个结点存放一个整数，并且结点数不少于 2，请设计算法以判断该链表中第二项起的每个元素值是否等于其序号的平方减去其前驱的值，若满足则返回 true，否则返回 false.

【解答】算法思想：要求对单链表结点的元素值进行运算，判断元素值是否等于其序号的平方减去其前驱的值。主要是结点的序号和前驱及后继指针的正确指向。

```
int   Judge（LinkedList   la）
{
  p=la->next->next；//p 是工作指针，初始指向链表的第二项。
  pre=la->next；      //pre 是 p 所指结点的前驱指针。
  i=2；                //i 是 la 链表中结点的序号，初始值为 2。
  while（p!=null）
```

```
    {
       if（p->data==i*i-pre->data)
       {
        i++;
        pre=p;
        p=p->next；
       }
      else
         break；
    }
    if（p!=null）
          return false； //未查到表尾就结束了。
    else
          return true;
}
```

（3）顺序表的题目：基数在前，偶数在后。

（4）求二叉树的深度（递归算法）

```
int Depth(BTree T)
{
    int ldepth, rdepth;
    if (T == NULL){
          return 0;
    }
    else{
          ldepth = Depth(T->lchild);
          rdepth = Depth(T->rchild);
    }
    if(ldepth > rdepth){
          return ldepth+1;
```

```
    }
    else{
        return rdepth+1;
    }
}
```

（4）折半查找

```
int BinSearch(int A[], int n, int key)
{
    int low=0, high=n-1, mid;
    while (low <= high)
    {
        mid = (low+high)/2;
        if (key == A[mid])
            return mid;
        if (key > A[mid])
            low = mid + 1;
        else
            high = mid – 1;
    }
    return -1;
}
```

递归算法

```
int RecurBinSearch(int A[], int low, int high, int key)
{
    int mid;
    if (low > high)
        return -1;
    else {
        mid = (low+high)/2;
```

```
    if (key == A[mid])
            return mid;
    if (key > A[mid])
          return RecurBinSearch(A, mid+1, high, key);
      else
          return RecurBinSearch(A, low, mid-1, key);
  }
}
```

（5）快速排序

```
void QuickSort(int A[], int low, int high)
{
      int pivotloc;
      if (low < high)
      {
            pivotloc = Partition(A, low, high);
            QuickSort(A, low, pivotloc-1);
            QuickSort(A, pivotloc+1, high);
      }
}
int Partition(int A[], int low, int high)
{
      int pivot;
      pivot = A[low];
      // 从线性表的两端交替地向中间扫描
      while (low < high)
      {
            while (low < high && A[high] >= pivot)
                  high--;
            A[low] = A[high];
```

```
        while (low < high && A[low] <= pivot)
            low++;
        A[high] = A[low];
    }
    A[low] = pivot;
    return low;
}
```

（6）二叉树的前序、中序、后序及层次遍历算法

```
void PreOrderTraverse(BinaryTreeNode * pRoot)
{
    if(pRoot == NULL)
        return;
    Visit(pRoot); // 访问根节点
    PreOrderTraverse(pRoot->m_pLeft); // 前序遍历左子树
    PreOrderTraverse(pRoot->m_pRight); // 前序遍历右子树
}

void InOrderTraverse(BinaryTreeNode * pRoot)
{
    if(pRoot == NULL)
        return;
    InOrderTraverse(pRoot->m_pLeft); // 中序遍历左子树
    Visit(pRoot); // 访问根节点
    InOrderTraverse(pRoot->m_pRight); // 中序遍历右子树
}

void PostOrderTraverse(BinaryTreeNode * pRoot)
{
    if(pRoot == NULL)
```

```
        return;
    PostOrderTraverse(pRoot->m_pLeft); // 后序遍历左子树
    PostOrderTraverse(pRoot->m_pRight); // 后序遍历右子树
    Visit(pRoot); // 访问根节点
}

void LevelTraverse(BinaryTreeNode * pRoot)
{
    if(pRoot == NULL)
        return;
    queue<BinaryTreeNode *> q;
    q.push(pRoot);
    while(!q.empty())
    {
        BinaryTreeNode * pNode = q.front();
        q.pop();
        Visit(pNode); // 访问节点
        if(pNode->m_pLeft != NULL)
            q.push(pNode->m_pLeft);
        if(pNode->m_pRight != NULL)
            q.push(pNode->m_pRight);
    }
    return;
}
```

二、操作系统部分

1、绪论

（1）操作系统的功能、特征及层次结构。操作系统的主要目标考过。

（2）用户态核心态的异同点，即双模式。中断和异常。

（3）操作系统的体系结构，考过 Unix 的结构是怎样的。有模块组合结构，层次结构和微内核结构。

（4）这一章主要考点是以判断题形式考查。

2、进程管理

（1）进程与线程的区别。进程的状态转换。

（2）作业调度、进程调度算法，掌握衡量调度算法的指标响应时间，周转时间，平均周转时间，带权平均周转时间的计算。这两年考的比较多。

（3）信号量实现同步，PV 操作，掌握生产者消费者，读者写者等经典问题。必须要会写程序。解决这类题目，要搞清楚进程间的关系，是同步关系还是异步关系，还要搞清楚互斥资源。画图，画图，画图-讲三遍。19 年可能考。

（4）死锁，理解死锁的本质概念，发生死锁时有几个进程，有多少资源才能不会发生死锁，这种问题只有充分理解了银行家算法，才能理解。记住是理解，不是背诵。

3、内存管理

（1）逻辑地址、物理地址的概念

（2）连续分配管理方式，单一分区、固定分区、动态分区等，这个好理解，画图即可理解。产生内碎片还是外碎片要知道。

（3）离散分配管理方式掌握页式存储管理，虚拟页式存储管理。这里涉及到重要的知识点是逻辑地址和物理地址的表示及转换，理解对应关系，比如页框对应页面。理解逻辑地址的够成。还是画图理解。

页：将进程划分的块，对应的大小就叫页面大小。

页框：将内存划分的块，也称为物理块。

页和页框二者一一对应，一个页放入一个页框，（理论上）页的大小和页框的大小相等。

页表：就是一个页和页框一一对应的关系表。页表是存放在内存中的。页表只是起到一个索引的作用，说白了就是能根据关系表能查到某一个页面和哪一个页框所对应。

（4）求 EAT，可以看出两部分，第一部分是计算找到物理地址的时间，第二部分是根据物理地址找到数据的时间（最后一次访问内存的时间）。最难的是第一部分如何找到物理地址，涉及到快表，页面不在内存中的情况，页面中断的时间，更新快表的时间。注意：如果是请求页式存储系统，找地址的顺序：先快表，再内存，内存不在则调页，调页完后更新到快表后需要再访问一次快表，此时得到了物理地址，进而访问内存，最终求出平均有效访问时间。好久没单独出题考了，19 年极有可能考。

4、文件管理

（1）大数据很火，这一章考查的概率及比重越来越大，要好好理解掌握。

（2）基本概念： 文件、逻辑结构、物理结构、目录等的定义概念。

（3）文件物理结构，连续、链接、索引。这个必须要掌握。掌握求索引文件的大小，掌握三种方式的优缺点。可以联系数据结构中的顺序表，链表来理解。索引类似于哈希。

（4）文件目录，目录的作用。19 年可能考。

（5）如何设计一个文件系统。目录的实现，文件的实现及文件的物理结构方面考虑。

（6）掌握磁盘调度算法，磁盘时间计算。

5、设备管理

（1）这一章考的很少，掌握缓冲 spooling 假脱机技术。

6、例题及知识点

1、双模式操作

硬件支持至少两种运行模式:

（1） 用户模式 （又叫用户态、目态） - 执行普通用户的应用程序

（2） 监控模式（Monitor mode，也称内核模式、系统模式、监控态、管态、特权模式) - 执行操作系统核心代码

（3）当中断或故障发生（自陷），硬件自动从用户态切换到系统态。

（4）当用户程序需要操作系统的服务（通过系统调用），它必须由目态切换到管态。

双重模式操作可以确保系统和用户程序不受错误的用户程序的影响。

（1）实现方法：将能引起损害的机器指令作为特权指令。 特权指令只能在系统态下运行。如果用户模式下试图 执行特权指令，那么硬件认为该指令非法，并以陷阱 的形式通知操作系统。

（2）由管态转换到目态是特权指令，其他的特权指令有 I/O 控制、定时器管理和中断管理。

2、操作系统中常用的调度算法总结

1）进程(作业)调度算法

（1）先来先服务调度算法（FCFS）:每次调度是从就绪队列中，选择一个最先进入就绪队列的进程，把处理器分配给该进程，使之得到执行。该进程一旦占有了处理器，它就一直运行下去，直到该进程完成或因发生事件而阻塞，才退出处理器。特点：利于长进程，而不利于短进程。

（2）短进程（作业）优先调度算法(SPF):它是从就绪队列中选择一个估计运行时间最短的进程，将处理器分配给该进程，使之占有处理器并执行，直到该进程完成或因发生事件而阻塞，然后退出处理器，再重新调度。

（3）时间片轮转调度算法:系统将所有的就绪进程按进入就绪队列的先后次序排列。每次调度时把 CPU 分配给队首进程，让其执行一个时间片，当时间片用完，由计时器发出时钟中断，调度程序则暂停该进程的执行，使其退出处理器，并将它送到就绪队列的末尾，等待下一轮调度执行。

（4）优先级调度算法:它是从就绪队列中选择一个优先权最高的进程，让其获得处理器并执行。

（5）响应比高者优先调度算法：它是从就绪队列中选择一个响应比最高的进程，

让其获得处理器执行，直到该进程完成或因等待事件而退出处理器为止。特点：既照顾了短进程，又考虑了进程到达的先后次序，也不会使长进程长期得不到服务，因此是一个比较全面考虑的算法，但每次进行调度时，都需要对各个进程计算响应比。所以系统开销很大，比较复杂。

（6）多级队列调度算法

（7）衡量算法的指标

作业周转时间（Ti）＝完成时间(Tei)－提交时间(Tsi)

作业平均周转时间(T)＝周转时间/作业个数

作业带权周转时间（Wi）＝周转时间/运行时间

响应比＝（等待时间＋运行时间）/运行时间

2）内存连续分配方式中动态分区分配算法

（1）首次适应分配算法（FF）：对空闲分区表记录的要求是按地址递增的顺序排列的，每次分配时，总是从第 1 条记录开始顺序查找空闲分区表，找到第一个能满足作业长度要求的空闲区，分割这个空闲区，一部分分配给作业，另一部分仍为空闲区。

（2）循环首次适应算法：每次分配均从上次分配的位置之后开始查找。

（3）最佳适应分配算法(BF)：是按作业要求从所有的空闲分区中挑选一个能满足作业要求的最小空闲区，这样可保证不去分割一个更大的区域，使装入大作业时比较容易得到满足。为实现这种算法，把空闲区按长度递增次序登记在空闲区表中，分配时，顺序查找。

（4）最坏适应算法(worst fit algorithm)：要求空闲区按其大小递减的顺序组成空闲区可用表或自由链。当用户作业或进程申请一个空闲区时，先检查空闲区可用表或自由链的第一个空闲可用区的大小是否大于或等于所要求的内存长度，若可用表或自由链的第一个项所示空闲区长度小于所要求的，则分配失败，否则从空闲区可用表或自由链中分配相应的存储空间给用户，然后修改和调整空闲区可用表或自由链。

3）请求分页存储系统中的页面置换算法

（1）最佳置换算法（OPT）：选择以后永不使用或在最长时间内不再被访问的内存页面予以淘汰。

（2）先进先出置换算法（FIFO）：选择最先进入内存的页面予以淘汰。

（3）最近最久未使用算法（LRU）：选择在最近一段时间内最久没有使用过的页，把它淘汰。

（4）最少使用算法（LFU）：选择到当前时间为止被访问次数最少的页转换。

4）磁盘调度算法

（1）先来先服务（FCFS）：是按请求访问者的先后次序启动磁盘驱动器，而不考虑它们要访问的物理位置

（2）最短寻道时间优先（SSTF）：让离当前磁道最近的请求访问者启动磁盘驱动器，即是让查找时间最短的那个作业先执行，而不考虑请求访问者到来的先后次序，这样就克服了先来先服务调度算法中磁臂移动过大的问题

（3）扫描算法（SCAN）或电梯调度算法：总是从磁臂当前位置开始，沿磁臂的移动方向去选择离当前磁臂最近的那个柱面的访问者。如果沿磁臂的方向无请求访问时，就改变磁臂的移动方向。在这种调度方法下磁臂的移动类似于电梯的调度，所以它也称为电梯调度算法。

（4）循环扫描算法（CSCAN）：循环扫描调度算法是在扫描算法的基础上改进的。磁臂改为单项移动，由外向里。当前位置开始沿磁臂的移动方向去选择离当前磁臂最近的哪个柱面的访问者。如果沿磁臂的方向无请求访问时，再回到最外，访问柱面号最小的作业请求。

一、 数据结构部分

注意：算法可用类 C、C++、类 JAVA 或类 PYTHON 等语言编写，请写出类型说明，关键语句请添加注释。

1、各种排序算法总结。

【解答】

各种排序算法的使用范围总结：（1）当数据规模较小的时候，可以用简单的排序算法如直接插入排序或直接选择排序。（2）当文件的初态已经基本有序时，可以用直接插入排序或冒泡排序。（3）当数据规模比较大时，应用速度快的排序算法。可以考虑用快速排序。当记录随机分布的时候，快排的平均时间最短，但可能出现最坏的情况，这时候的时间复杂度是 O(n^2)，且递归深度为 n,所需的栈空间问 O(n)。(4)堆排序不会出现快排那样的最坏情况，且堆排序所需的辅助空间比快排要少。但这两种算法都不是稳定的，若要求排序时稳定的，可以考虑用归并排序。（5）归并排序可以用于内排序，也可以用于外排序。在外排序时，通常采用多路归并，并且通过解决长顺串的合并，产生长的初始串，提高主机与外设并行能力等措施，以减少访问外存额次数，提高外排序的效率。

各类排序算法的特点及比较：

冒泡排序算法思想：将待排序的元素看作是竖着排列的“气泡”，较小的元素比较轻，从而要往上浮。在冒泡排序算法中我们要对这个“气泡”序列处理若干遍。所谓一遍处理，就是自底向上检查一遍这个序列，并时刻注意两个相邻的元素的顺序是否正确。如果发现两个相邻元素的顺序不对，即“轻”的元素在下面，就交换它们的位置。

选择排序算法思想：选择排序的基本思想是对待排序的记录序列进行 n-1 遍的处理，第 i 遍处理是将 L[i..n]中最小者与 L[i]交换位置。这样，经过 i 遍处理之后，前 i 个记录的位置已经是正确的了。

插入排序算法思想：经过 i-1 遍处理后,L[1..i-1]已排好序。第 i 遍处理仅将 L[i]插入 L[1..i-1]的适当位置，使得 L[1..i]又是排好序的序列。

快速排序算法思想：快速排序的基本思想是基于分治策略的。对于输入的子序列 L[p..r]，如果规模足够小则直接进行排序，否则分三步处理：1.分解(Divide)：将输入的序列 L[p..r]划分成两个非空子序列 L[p..q]和 L[q+1..r]，使 L[p..q]中任一元素的值不大于 L[q+1..r]中任一元素的值。2.递归求解(Conquer)：通过递归调用快速排序算法分别对 L[p..q]和 L[q+1..r]进行排序。3.合并(Merge)：由于对分解出的两个子序列的排序是就地进行的，所以在 L[p..q]和 L[q+1..r]都排好序后不需要执行任何计算 L[p..r]就已排好序。

归并排序算法思想：分而治之(divide - conquer)。每个递归过程涉及三个步骤：1.分

解，把待排序的 n 个元素的序列分解成两个子序列，每个子序列包括 n/2 个元素。2.治理，对每个子序列分别调用归并排序 MergeSort，进行递归操作。3.合并，合并两个排好序的子序列,生成排序结果。

Shell 排序算法思想：算法先将要排序的一组数按某个增量 d 分成若干组，每组中记录的下标相差 d.对每组中全部元素进行排序，然后再用一个较小的增量对它进行，在每组中再进行排序。当增量减到 1 时，整个要排序的数被分成一组，排序完成。

堆排序算法思想：用大根堆排序的基本思想：1.先将初始文件 R[1..n]建成一个大根堆，此堆为初始的无序区。2.再将关键字最大的记录 R[1](即堆顶)和无序区的最后一个记录 R[n]交换，由此得到新的无序区 R[1..n-1]和有序区 R[n]，且满足 R[1..n-1].keys≤R[n].key。3.由于交换后新的根 R[1]可能违反堆性质，故应将当前无序区 R[1..n-1]调整为堆。

【点评】 重点掌握快速排序、希尔排序、堆排序。

2、

(1)什么是最小生成树？

(2)简述常用的构造最小生成树 Prim 算法的思想。

(3)给定一个无向图，要会画出最小生成树

【解答】

（1）如果无向连通图是一个网，那么，它的所有生成树中必有一棵边的权值总和最小的生成树，我们称这棵生成树为最小生成树，简称为最小生成树。

（2）假设 G＝（V，E）为连通图，其中 V 为网图中所有顶点的集合，E 为网图中所有带权边的集合。设置两个新的集合 U 和 T，其中集合 U（顶点集） 用于存放 G 的最小生成树中的顶点，集合 T（边集合）存放 G 的最小生成树中的边。T ，U 的初始状态：令集合 U 的初值为 U＝{u1}（假设构造最小生成树时，从顶点 u1 出发），集合 T 的初值为 T＝{}。

Prim 算法的思想是：从所有 u∈U，v∈V－U 的边中，选取具有最小权值的边（u，v）∈E，将顶点 v 加入集合 U 中，将边（u，v）加入集合 T 中，如此不断重复，直到 U＝V 时，最小生成树构造完毕，这时集合 T 中包含了最小生成树的所有边。

Prim 算法可用下述过程描述，其中用 wuv 表示顶点 u 与顶点 v 边上的权值。

a、U＝{u1},T={};

b、while (U≠V)do

(u，v)＝min{wuv；u∈U，v∈V－U }

T＝T＋{(u，v)}

U＝U＋{v}

c、结束。

3、图的邻接矩阵表示法会考，要掌握，掌握最短路径算法迪杰斯特拉（Dijkstra）算法的思想。

4、设哈希表的填装因子为 2/3，现要把数据：1,13,12,34,38,33,27,22 插入散列表中。

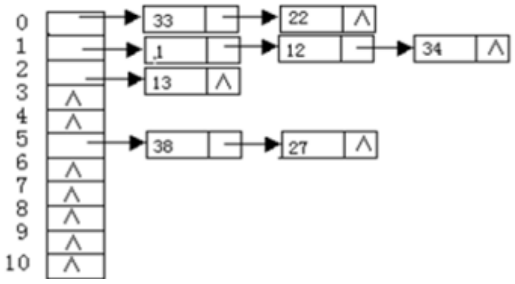
（1）使用除留余法构造散列函数，并使用链地址方法来构造散列表。

（2）等概率情况下查找成功时和不成功时的平均查找长度为多少？

（3）若查找关键字 34，则需要依次与哪些关键字比较。

【解答】

（1）由题意知，可求出散列表的表长为 12，取不大于表长的最大素数 11，则构造的散列函数为 hashf(x)=x mod 11，采用链地址法构造散列表，先算出关键字的哈希值，将关键字结点插入到哈希地址所在的链表中。有哈希函数克制，散列地址空间是 0—10，链表地址构造的表如下：



（2）在链地址表中查找成功时，查找关键字为 33 的记录需要进程 1 次探测，关键字 22 需要进行 2 次探测，一次类推。故成功时的平均查找长度为：

$ASL_{成功}$ = （1*4 +2*3+3）/8 = 13/8

查找失败时，假设对空结点的查找长度为 0，则对于地址 0，查找失败的探测次数为 2；对于地址 1，查找失败的探测次数为 3，则平均探测查找长度为:

$ASL_{失败}$ = （2+3+1+0+2+0+0+0+0+0+0）/11 = 8/11

（3）查找关键字 34，需要依次与关键字 1,12,34 进行比较。

【点评】 掌握哈希表相关内容。

5、（1）子串是主串中任意个连续字符组成的序列。（√）

（2）双端队列是一种特殊的线性表，对它所有的插入和删除都限制在表的两端进行。（√）

（3）哈夫曼树中没有度为 1 的结点，所以必为满二叉树。（×）

（4）邻接矩阵适用于稀疏图的表示，邻接表适用于稠密图的表示。（×）

（5）折半搜索所对应的判定树，既是一棵二叉搜索树，又是一棵理想平衡二叉树。（√）

（6）在采用线性探测法处理冲突的散列表中，所有同义词在表中一定相邻。（×）

6、试编写尽可能高效的算法找出不带头结点的单链表中倒数第 k 个结点，已知单链表的头指针。

【解答】分析：首先容易想到的是，可以先获取到链表的长度，然后输出第 length-k 个元素的值，就是倒数第 k 个元素的值了，这样得遍历两次链表。能否在遍历一遍链表的情况下找到所求结点，可以采取判断链表循环的思路。

算法思想：采用两个指针 p，q，第一个指针 p 提前 k-1 步向下走，第二个指针 q 再随着第一个指针 p 一直走；直到第一个指针 p 指向末尾，第二个指针 q 的元素的值，便是所求的值。具体算法如下：

```
typedef struct LNode{
    int data;
    struct LNode * next;
}LinkListNode;
int getReverseKNum(LinkListNode *head,int k){
    LinkListNode *p,*q;
    int i;
    if(head->next == NULL || k == 0)
        return -1;
```

```
    p = head->next;
    for(i=1;i<k;i++){
        if(p->next == NULL)
            return -1;
        p = p->next;
    }
    q = head->next;
    while(p->next!=NULL){
        p = p->next;
        q = q->next;
    }
    return q-> data;
}
```

【总结】掌握用两个指针遍历的思想，顺序表单链表中都可以使用。

7、一个台阶总共有 n 级，如果一次可以跳 1 级，也可以跳 2 级。求总共有多少总跳法，并分析算法的时间复杂度。

【解答】分析：如果只有 1 级台阶，那显然只有一种跳法。如果有 2 级台阶，那就有两种跳的方法了：一种是分两次跳，每次跳 1 级；另外一种就是一次跳 2 级。

我们把 n 级台阶时的跳法看成是 n 的函数，记为 f(n)。当 n>2 时，第一次跳的时候就有两种不同的选择：一是第一次只跳 1 级，此时跳法数目等于后面剩下的 n-1 级台阶的跳法数目，即为 f(n-1)；另外一种选择是第一次跳 2 级，此时跳法数目等于后面剩下的 n-2 级台阶的跳法数目，即为 f(n-2)。因此 n 级台阶时的不同跳法的总数 f(n)=f(n-1)+(f-2)。

我们把上面的分析用一个公式总结如下：

```
        /   1                       n=1
f(n)=   —   2                       n=2
        \   f(n-1)+(f-2)            n>2
```

由递推式可知是 Fibonacci 序列。下面给出算法时间复杂度为 O（n）的算法 1。

```
int   Fibonacci (int n)
{
    int result[2] = {1, 2};
    if(n>0 && n < 2)
        return result[n];

    int   fibNMinusOne = 1;
    int   fibNMinusTwo = 0;
    int   fibN = 0;
    for(int int i = 2; i <= n; ++ i)
    {
        fibN = fibNMinusOne + fibNMinusTwo;
        fibNMinusTwo = fibNMinusOne;
        fibNMinusOne = fibN;
    }
```

```
    return fibN;
}
```

递归算法 2：递归方法计算的时间复杂度是以 n 的指数的方式递增.

```
int Fibonacci2(int int n)
{
    int result[2] = {1, 2};
    if(n>0 && n < 2){
        return result[n];
    }
    return Fibonacci2(n - 1) + Fibonacci2(n - 2);
}
```

【点评】这道题的意义在于大家掌握递归算法，好好琢磨，递归 19 年必考，极大可能体现在单链表中。

8、编写递归算法实现判断二叉树是否是平衡二叉树。

【解答】

算法思想：如果二叉树为空，返回真；如果二叉树不为空，如果左子树和右子树都是 AVL 树并且左子树和右子树高度相差不大于 1，返回真，其他返回假

具体算法：

```
bool IsAVL(BinaryTreeNode * pRoot, int & height)
{
    if(pRoot == NULL) // 空树，返回真
    {
        height = 0;
        return true;
    }
    int heightLeft;
    bool resultLeft = IsAVL(pRoot->m_pLeft, heightLeft);
    int heightRight;
    bool resultRight = IsAVL(pRoot->m_pRight, heightRight);
    if(resultLeft && resultRight && abs(heightLeft - heightRight) <= 1) // 左子树和右子树都是AVL，并且高度相差不大于 1，返回真
    {
        height = max(heightLeft, heightRight) + 1;
        return true;
    }
    else
    {
        height = max(heightLeft, heightRight) + 1;
        return false;
    }
}
```

【举一反三】判断二叉树是否为完全二叉树，是否为严格二叉树。

二、操作系统部分

6、判断下列论述是否正确，若果有错，请指出错误之处。

（1）文件系统中分配存储空间的基本单位不是记录。（√）
（2）具有多道功能的操作系统一定是多用户操作系统。（×）
（3）存储器是由操作系统提供的一个假想的特大存储器，它并不是实际的内存，其大小可比内存空间大得多。（√）
（4）批处理系统的主要优点是系统的吞吐量大、资源利用率高、系统的开销较小。（√）
（5）文件系统中源程序是有结构的记录式文件。（×）
（6）若没有进程处于运行状态，则就绪队列和等待队列均为空。（×）
（7）顺序文件适合建立在顺序存储设备上，而不适合建立在磁盘上。（×）
（8）SPOOLing 系统实现设备管理的虚拟技术，即：将独占设备改造为共享设备。它由专门负责 I/O 的常驻内存进程以及输入、输出井组成。（√）
（9）系统调用是操作系统与外界程序之间的接口，它属于核心程序。在层次结构设计中，它最靠近硬件。（×）
（10）若系统中存在一个循环等待的进程集合，则必定会死锁。（×）
（11）分页式存储管理中，（页的大小）是可以不相等的。（×）
（12）对磁盘进行移臂调度优化的目的是为了缩短启动时间。（√）
（13）虚拟存储器的最大容量是由磁盘空间决定的。（×）
（14）单级文件目录可以解决文件的重名问题。（×）
（15）进程控制一般都由操作系统内核来实现。（×）

7、为什么要引入线程，解释一下线程和进程之间的相互关系。

【解答】因为虽然进程可以提供 CPU 的利用率，但是进程之间的切换是非常耗费资源和时间的，为了能更进一步的提供操作系统的并发，进而引入了线程，这样，进程是分配资源的基本单位，而线程则是系统调度的基本单位。一个进程内部的线程可以共享改进程的所分配到的资源。线程的创建与撤销，线程之间的切换所占用的资源比进程要少很多。总的来说就是为了更进一步提高系统的并发性，提高 CPU 的利用率。线程是进程的基础，进程包含多个线程，是线程的载体。

【点评】理解进程和线程，并发性及并行性。

8、假定在单 CPU 条件下有下列要执行的作业

作业	运行时间	优先级
1	10	2
2	4	3
3	3	0

作业到来的时间是按作业编号顺序进行的（即后面作业依次比前一个作业迟到一个时间单位，优先级数越小，优先级越高）。

（1）用一个执行时间图描述在采用非抢占式优先级算法时执行这些作业的情况。
（2）对于上述算法，各个作业的周转时间是多少?平均周转时间是多少?
（3）对于上述算法，各个作业的带权周转时间是多少?平均带权周转时间是多少?

【解答】：如下：

作业	到达时间	运行时间	完成时间	周转时间	带权周转时间

1	0	10	10	10	1.0
2	1	4	17	16	4.0
3	2	3	13	11	3.7
平均周转时间		12.3			
平均带权周转时间		2.9			

【点评】掌握进程的调度及衡量调度算法的指标计算。

9、某虚拟存储器的用户空间共有 32 个页面，每页 1KB，主存 16KB。试问：

（1）逻辑地址的有效位是多少？

（2）物理地址需要多少位？

（3）假定某时刻系统用户的第 0,1,2,3 页分别分配的物理块号为 5,10,4,7，试将虚拟地址 0A5C 和 093C 变为物理地址。

【解答】

（1）程序的空间大小为 32KB，则逻辑地址的有效位数为 15 位。

（2）内存空间的大小为 16KB，因此物理地址至少需要 14 位。

（3）当页面为 1KB 时，则页内位移的位数为 10 位，可以算出页号的位数为 5 位，虚拟地址 0A5C 表示页号为 00010，页内地址是 1001011100。该页在内存的第 4 块，即块号为 0100，因此 0A5C 的物理地址为：01 0010 0101 1100，即 125CH。同理 093C 的物理地址为：113CH。

【点评】掌握虚拟存储的逻辑地址与物理地址间的转换。

10、请求分页管理系统中，假设某进程的页表内容如下表所示。页面大小为 4KB，一次内存的访问时间是 100ns，一次快表（TLB）的访问时间是 10ns，处理一次缺页的平均时间是 108ns（已含更新 TLB 和页表的时间），进程的驻留集的大小固定为 2，采用最近最少使用置换算法（LRU）和局部淘汰策略。假设①TLB 初始为空；②地址转换时先访问 TLB，若 TLB 未命中，再访问页表（忽略访问页表之后的 TLB 更新时间）；③有效位为 0 表示页面不在内存，产生缺页中断，缺页中断处理后，返回到产生缺页中断的指令处重新执行。页表如下：

页号	页框（Page Frame）号	有效位（存在位）
0	101H	1
1	—	0
2	254H	1

设有虚地址访问序列 2362H、1565H、25A5H，请问：

（1）依次访问上述三个虚地址，各需多少时间？给出计算过程。

（2）基于上述访问序列，虚地址 1565H 的物理地址是多少？请说明理由。

【解答】

（1）根据页式管理工作原理，应先考虑页面大小，以便将页号和页内位移分解出来。页面大小为 4KB，即 212，则得到页内位移占虚地址 12 位，页号占剩余高位。可得三个虚地址的页号 P 如下：

2362H：P=2，访问块表 10ns，因初始为空，访问页表 100ns 得到页框号，合成物理地址后访问主存 100ns，共计 10ns+100ns+100ns=210ns。

1565H：P=1，访问块表 10ns，落空，访问页表 100ns，落空，进行缺页中断处理 108ns，合成物理地址后访问主存 100ns，共计 10ns+100ns+108ns+100ns=318n。

25A5H：P=2，访问块表，因第一次访问已将该页号放入块表，因此花费 10ns 便可合成物理地址，访主存 100ns，共计 10ns+100ns=110ns。

（2）当访问虚地址 1565H 时，产生缺页中断，合法驻留集为 2，必须从页表中淘汰一个页面，根据题目的置换算法，应淘汰 0 号页面，因此 1565H 的对应页框号为 101H。由此可得 1565H 物理地址为 101565H。

【点评】掌握求 EAT。

11、文件 Y 由 80 条记录组成，记录从 1 开始编号。用户打开文件后，欲将内存中的一条记录插入到文件 Y 中，作为其第 30 条记录。请回答下列问题，并说明理由。

（1）若文件系统采用连续分配方式，每个磁盘块存放一条记录，文件 Y 存储区域前后均有足够的空闲磁盘空间，则完成上述插入操作最少需要访问多少次磁盘块？Y 的文件控制块内容会发生哪些改变？

（2）若文件系统采用链接分配方式，每个磁盘块存放一条记录和一个链接指针，则完成上述插入操作需要访问多少次磁盘块？

（3）若每个存储块大小为 1KB，其中 4 个字节存放连接指针，则该文件系统支持的文件最大长度是多少？

【解答】（1）系统采用连续分配时，插入记录需要移动其他记录块（类似数据结构中数组的插入），这个文件有 80 条记录，插入新记录为第 30 条，而存储前后均有足够的磁盘空间，且要求最少的访问存储块数，则把文件的前 29 条记录向前移动，若移动一条记录时读出和存回磁盘各算一次访问磁盘，则 29 条记录共访问磁盘 58 次，存回第 30 条记录访问 1 次，则总共访问磁盘 59 次。

（2）文件系统采用链接分配方式时，插入记录需找到相应的记录，并修改指针即可。首先找到文件系统的第 29 块，需要访问磁盘 29 次，然后把第 29 块的指针（下块地址）指向新块（即第 30 块），把新块存回内存访问磁盘 1 次，然后修改内存中第 29 块的下块地址字段，再存回磁盘需要访问磁盘 1 次，总共 31 次。

（3）4 个字节共 32 位，可以寻址 2^{32} = 4G 块存储块，每块的大小为 1KB，即 1024B，其中下块地址部分占 4B，那么该系统的文件最大长度是 4G×1024B = 4080GB

【点评】掌握连续、链接、索引分配访问磁盘次数（插入、删除操作）

数据结构：

① 解算法题时尽量用画图的方式帮助分析题意，得出解题思路。解题步骤：先写算法思想再写出单链表、二叉树的结构体定义，再写具体算法代码，最后有必要在重要的代码块加注释。② 算法题写代码思路：尽可能套用模板代码，如二叉树的遍历，都是在此基础上演变而来的。注意运用两个指针（快慢，前后）的思想。③图的算法可能考

操作系统：

①操作系统所涉及的计算题：逻辑地址变换为物理地址；虚拟地址变换为物理地址；页面置换（淘汰）算法的缺页率；磁盘调度算法磁道移动次数；多级索引的文件大小；有效访问时间。② 解进程同步 PV 题时尽量用画图的方式帮助分析题意，理清各个进程间的同步 or 互斥关系，定义用于表示互斥和同步的信号量。③ 判断题一般是错的多，对的情况 1-2 道。