算法设计考试范围

1.填空题(共10题，每题1分)

2.选择题(共5题，每题2分)

3.判断题(共5题，每题1分)

4.地推关系式(共1题，共10分)

5.解答题(共5题，每题10分)

其中包括:

(1)插入排序

(2)快速排序

这两个主要看程序考算法

(3)单元点最短路径

这题会根据一个实际问题考最短路径的算法

(4)多段图最小成本问题

(5)资源分配问题

6.背包问题(共1题，共15分)

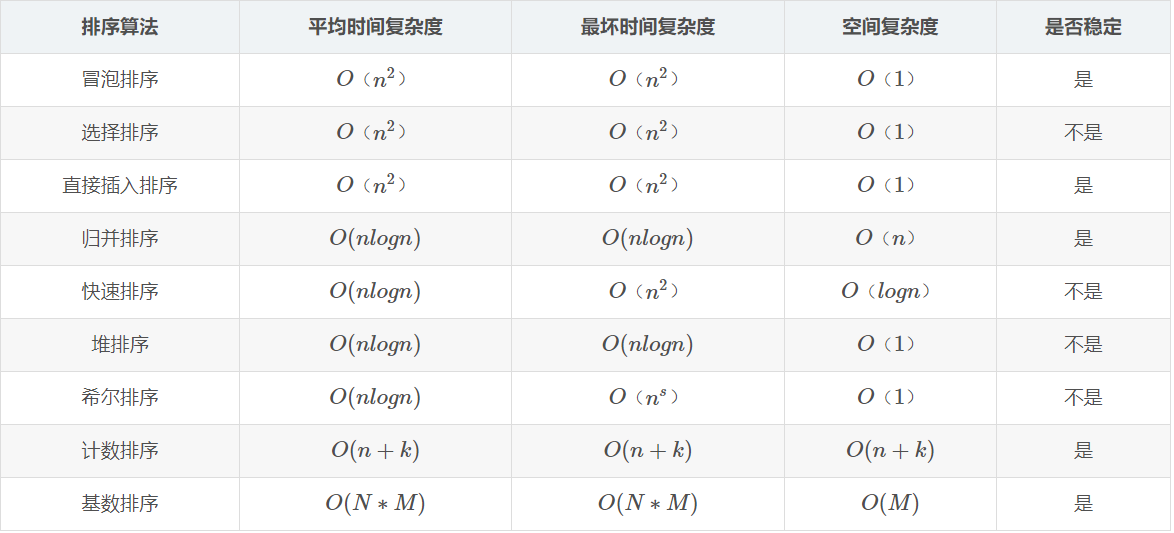
注:

1.第1大题主要考一些理论方面的问题，例如:分治算法的步骤，贪心算法的思想之类的。

2.会考一个快速排序的不稳定问题，快速排序在时候效率不如地柜算法的插入排序，这个不在书本介绍范围内，略有超纲，特别说明。

3.递归会有那个经典的两个柱子上落盘子的问题，参考递归的方法说明。

常用排序比较：



时间复杂度记忆-

冒泡、选择、直接 排序需要两个for循环，每次只关注一个元素，平均时间复杂度为O（n2）O（n2）（一遍找元素O(n)O(n)，一遍找位置O(n)O(n)）

快速、归并、希尔、堆基于二分思想，log以2为底，平均时间复杂度为O(nlogn)O(nlogn)（一遍找元素O(n)O(n)，一遍找位置O(logn)O(logn)）

稳定性记忆-“快希选堆”稳定。

“算法的稳定性”指的是：排序前后相同元素的相对位置不变，则称排序算法是稳定的；否则排序算法是不稳定的。

**插入排序：**

工作原理是通过构建有序序列，对于未排序数据，在已排序序列中从后向前扫描，找到相应位置并插入。插入排序在实现上，在从后向前扫描过程中，需要反复把已排序元素逐步向后挪位，为最新元素提供插入空间。

核心算法：

//array[]为待排序数组，n为数组长度

void insertSort(int array[], int n)

{

int i,j,temp;

for( i=1;i<n;i++)

{

if(array[i]<array[i-1])

{

temp=array[i];

for( j=i;array[j-1]>temp;j--)

{

array[j]=array[j-1];

}

array[j]=temp;

}

}

}

**快速排序：**

步骤为：

1、从数列中挑出一个元素，称为"基准"（pivot），

2、重新排序数列，所有元素比基准值小的摆放在基准前面，所有元素比基准值大的摆在基准的后面（相同的数可以到任一边）。在这个分区结束之后，该基准就处于数列的中间位置。这个称为分区（partition）操作。

3、递归地（recursive）把小于基准值元素的子数列和大于基准值元素的子数列排序。

核心算法：

void adjust\_quicksort(int k[],int n)

{

quicksort(k,0,n-1);

}

void quicksort(int a[], int left, int right)

{

int i,j,t,temp;

if(left>right) //（递归过程先写结束条件）

return;

temp=a[left]; //temp中存的就是基准数

i=left;

j=right;

while(i!=j)

{

//顺序很重要，要先从右边开始找（最后交换基准时 //换过去的数要保证比基准小，因为基准

//选取数组第一个数，在小数堆中）

while(a[j]>=temp && i<j)

j--;

//再找右边的

while(a[i]<=temp && i<j)

i++;

//交换两个数在数组中的位置

if(i<j)

{

t=a[i];

a[i]=a[j];

a[j]=t;

}

}

//最终将基准数归位 （之前已经temp=a[left]过了，交换只需要再进行两步）

a[left]=a[i];

a[i]=temp;

quicksort(left,i-1);//继续处理左边的，这里是一个递归的过程

quicksort(i+1,right);//继续处理右边的 ，这里是一个递归的过程

}

