一、（总分 10 分）

（1）判断 $3n\lfloor \log_2 n \rfloor = O(n^2)$ 是否正确，请证明；（3 分），

（2）如下 C（n）表示算法复杂度函数（输入规模为 n），用 O(.)渐进函数表示每个算法的复杂度（5 分），并写出下列算法的复杂度的大小排序；（2 分）

A．C（n）=C（n － 1）+k，k 是常量，C（1）=1；

B．C（n）= $2n^2 +5n+1/n$；

C．C（n）= $10^{100}\log_2 n +3n\log_2 n$；

D．C（n）=C（n/2）+100，C（1）=1；

参考答案：

（1）正确，证明：对于任意的正整数 n，

$|3n\lfloor \log_2 n \rfloor| \leqslant |3n\ \log_2 n| \leqslant 3|n^2|$

取 $n_0=1$，C=3，根据定义知命题成立。

（2）A：C(n)=O(n);（2 分）

B：$C(n)=O(n^2)$; （2 分）

C：$C(n)=O(n\log_2 n)$; （2 分）

D：$C(n) = O(\log_2 n)$； （2 分）

因为： $2n^2 > n\log_2 n > n > \log_2 n$；

所以：B>C>A>D （2 分）

二、 （总分 10 分）对以下程序段进行时间复杂度分析。

（1）写出下面程序的时间复杂度的递推式（递归方程）；并分析算法

的空间复杂度，并用大“O(·)”记号表示。（5 分）

```
MERGESORT(low，high)
  if low<high;
   then mid←(low，high)/2;
     MERGESORT(low，mid);
     MERGESORT(mid+1，high);
     MERGE(low，mid，high);
  endif
 end MERGESORT
```

答：

递归方程：（2 分）

$$T(n)=\begin{cases} a & n=1 \\ 2T(n/2)+cn & n>1 \end{cases}$$

设 n=2^k

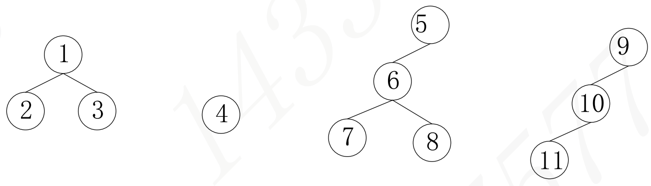
解递归方程：

$$\begin{aligned} T(n) &= 2(2T(n/4)+cn/2)+cn \\ &= 4T(n/4)+2cn \\ &\cdots\cdots \\ &= 2^k T(1)+kcn \\ &= an+cn\log n \end{aligned}$$

算法复杂度：O(nlogn)，（3 分）

三、（总分 10 分）现有 A，B，C，D，4 棵数如下，假设其根节点的

初始秩相等。



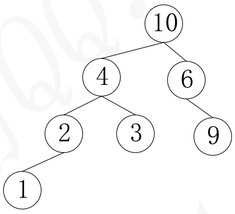
(A)　　(B)　　(C)　　(D)

（1）UNION（4,3），UNION（7,10），UNION（4,9），FIND（8），FIND（6），UNION 操作中的 FIND 不考虑路径压缩，请画出每一步操作完成后的树表示;（要求：对于相同秩的两棵树进行 UNION 操作，默认第二个参数的父节点的秩加 1)；

参考答案：

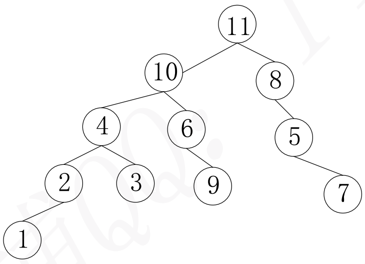
（1）

(1)



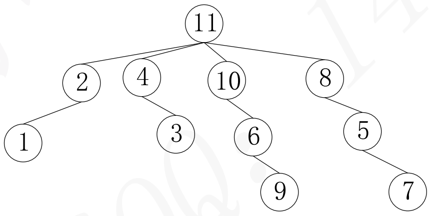
UNION（1,9）；

(2)



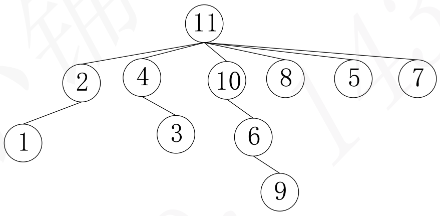
UNION（3,7）；

(3)



FIND（2）；

(4)



FIND（7）；

四、（总分 15 分）给你一个装有 16 个硬币的袋子。16 个硬币中有一个是伪造的，并且那个伪造的硬币比真的硬币要轻一些。你的任务是找出这个伪造的硬币。为了帮助你完成这一任务，将提供一台可用来比较两组硬币重量的仪器，利用这台仪器，可以知道两组硬币的重量是否相同。

（1）写出分治算法的主要思路；（3 分）

（2）如果问题规模为 n，假设硬币的重量用数组 a[i]表示，请用递归调用方式写出算法伪代码；（4 分）试分析算法的时间复杂度；（3

分）

（3）试分析 n=9 和 10，即 n 分别为奇数和偶数，两种情形下的分治过程。（5 分）

参考答案：

（1）算法思路：（3 分）

① 硬币个数≤2，则直接比较，找出伪造币。否则，转②。

②若 n%2=0，则将其分为个数相等的两部分，选择轻的部分保留，转①；否则转③。

③将 a[0…n-2]分为相等的两部分：若两部分重量相等，则 a[n-1]为伪造币，终止；若不等，则保留轻的部分，转①。

（2）伪代码：（4 分）

步骤:略

算法复杂度：以比较操作为基本运算，最好情况比较 1 次，最坏比较 logn 次，

（3）① 分成两部分：a[0…4]、a[5…9]，假定后者轻，保留 a[5…9] ② 分成三部分：a[5…6]、a[7…8]、a[9]，若前两者一样重，故劣质球为 a[9]。

五、（总分 15 分）N 个人过河，船每次只能坐两个人，船载每个人过河的所需时间不同 t[i]，每次过河的时间为船上的人的较慢的那个，问最快的过河时间。(船划过去要有一个人划回来)

（1）请写出两种贪心策略；（5 分）

（2）假设四人所需要的时间 t[i]分别是 1、2、5、8 分钟，说明两种贪心算法过河的步骤以及需要的总时间是多少？（5 分）

（3）写出较优贪心算法的主要思路（伪代码）。（5 分）

解法：

（1）贪心策略：

先将所有人过河所需的时间按照升序排序，我们考虑把单独过河所需要时间最多的两个旅行者送到对岸去，有两种方式：

1.最快的和次快的过河，然后最快的将船划回来；次慢的和最慢的过河，然后次快的将船划回来，所需时间为：t[0]+2*t[1]+t[n-1]；

2.最快的和最慢的过河，然后最快的将船划回来，最快的和次慢的过河，然后最快的将船划回来，所需时间为：2*t[0]+t[n-2]+t[n-1]。

（2）第一种办法：先让甲乙过去（2 分钟），甲回来（1 分钟），甲丙过去（5 分钟），甲回来（1 分钟），甲丁再过去（8 分钟），总共需要 17 分钟就可以让四个人都过去。

第二种办法：先让甲乙过去（2 分钟），甲回来（1 分钟），丙丁过去（8 分钟），乙回来（2 分钟），甲乙再过去（2 分钟），总共需要 15 分钟就可以让四个人都过去。

（5 分）

（3）贪心算法的主要思路

```
#include<iostream>
#include<algorithm>
using namespace std;

```

```
int main()
{
    int a[1000],t,n,sum;
    scanf("%d",&t);
    while(t--)
    {
        scanf("%d",&n);
        sum=0;
        for(int i=0;i<n;i++) scanf("%d",&a[i]);
        while(n>3)
        {
            sum=min(sum+a[1]+a[0]+a[n-1]+a[1],sum+a[n-1]+a[0]+a[n-2]+a[0]);
            n-=2;
        }
        if(n==3) sum+=a[0]+a[1]+a[2];
        else if(n==2) sum+=a[1];
        else sum+=a[0];
        printf("%d\n",sum);
    }
}
```

六、（总分 15 分）试用动态规划算法实现下列问题求解：设 A 和 B 是两个字符串。我们要用最少的字符操作，将字符串 A 转换为字符串 B，这里所说的字符操作包括：删除一个字符（delete）、插入一个字符（insert）、将一个字符改为另一个字符（replase）。对于原字符串 A[1,...,i]，目标字符串 B[1,...,j]，将字符串 A 变换为字符串 B 所用的最少字符操作数称为字符串 A 到 B 的**编辑距离**，则编辑距离定义为 C[i,j]。

例如将 kitten 一字转成 sitting：第一步：sitten （k 改为 s）；第二步：sittin （e 改为 i）；第三步：sitting （插入 g）；则其编辑距离为 3；

（1）请写出求解编辑距离的动态规划思路，并写出该算法的递归方

程；（8 分）

（2）计算字符串 A=fail 转换为字符串 B= sai 的编辑距离，写出动态规划计算编辑距离的矩阵表示。（7 分）

参考答案：

（1）动态规划思路：

首先定义这样一个函数——c(i, j)，它表示第一个字符串的长度为 i 的子串到第二个字符串的长度为 j 的子串的编辑距离。

当 A 的第 i 个字符 x 与 B 的第 j 个字符 y 进行比较，如果：

if x == y, then c[i, j] == c[i-1,j-1]

if x != y, and we insert y for A, then c[i,j] = c[i,j-1] + 1

if x != y, and we delete x for A, then c[i,j] = c[i-1,j] + 1

if x != y, and we replace x with y for A, then c[i,j] = c[i-1,j-1] + 1

显然可以有如下动态规划公式：

- if i == 0 且 j == 0，c(i, j) = 0
- if i == 0 且 j > 0，c(i, j) = j
- if i > 0 且 j == 0，c(i, j) = i
- if i ≥ 1 且 j ≥ 1 ，c(i, j) == min{ c(i-1, j) + 1, c(i, j-1) + 1, c(i-1, j-1) + f(i, j) }，当第一个字符串的第 i 个字符不等于第二个字符串的第 j 个字符时，f(i, j) = 1；否则，f(i, j) = 0。

（2）动态规划算法代码：

```
int dist( )
{
  int m=A.size( );
  int n=B.size( );
  vector<int>c(n+1,0);
  for(int i=1;i<=n;i++) c[i]=i;
  for(i=1;i<=m;i++){
```

```
    int y=i-1;
    for(int j=1;j<=n;j++){
      int x=y;
      y=c[j];
      int z=j>1?c[j-1]:i;
      int del=A[i-1]= B[j-1]?0:1;
      c[j]=min(x+del,y+1,z+1);
    }
  }
  return c[n];
}
```

（3）编辑距离矩阵

	0	f	a	i	l
0	0	1	2	3	4
s	1	1	2	3	4
a	2	2	1	2	3
i	3	3	2	1	2

七、（总分 15 分）羽毛球队有男女运动员各 n 人。给定两个 n×n 的矩阵 P 和 Q。P[i][j]是男运动员 i 和女运动员 j 配合组成混合双打的竞赛优势，Q[i][j]是女运动员 i 和男运动员 j 配合的竞赛优势。由于技术配合或心理状况等各种因素的影响，P[i][j]并不一定等于 Q[j][i]。男运动员 i 女运动员 j 配合组成混合双打的男女双方竞赛优势乘积为 P[i][j]×Q[j][i]。

请采用回溯法设计一个算法，计算男女运动员最佳搭配的配对法，使

得各组男女双方竞赛优势乘积的总和达到最大。

（1）写出回溯法的算法思路，画出状态空间树。（6 分）

（2）考虑算法的剪枝方法，并说明；（4 分）

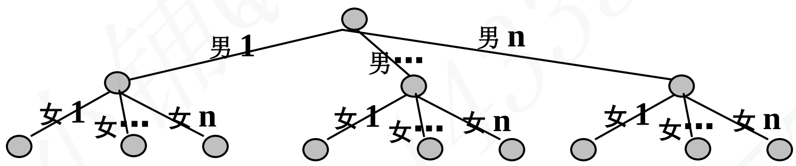
（3）如下面 P 和 Q 数组的数据：

P=	10	2	3	Q=	2	2	2
	2	3	4		3	5	3
	3	4	5		4	5	1

计算最大的男女双方竞赛优势总和，并写出最佳搭配；（5 分）

参考答案：

（1）状态空间树



在这个状态空间树中采用回溯方法，将男女队员的竞赛优势乘积计算出来，然后将各组男女的优势乘积进行相加。找出最大值。

（2）剪枝方法：

由于一个男队员只能和一个女队员搭档，反之也同理，因此，对于搜索的第一步选定某男和某女，那么第二个男队员就不能和第一个男队员的女搭档组合，因此，剪去改女队员的分枝。

（3）最大的男女双方竞赛优势总和为：

10*2 + 4*5 + 4*3 = 52

最佳搭配为：（女 1，男 1）（女 2，男 3）（女 3，男 2）

八、（总分 10 分）什么是 P 问题，NP 问题；（5 分）试分析图的 3 着色问题是哪一类问题，并说明原因；（5 分）

参考答案：

（1）P 问题解释：如果一个问题可以找到一个能在多项式的时间里解决它的算法，那么这个问题就属于 P 问题。（2 分）

（2）NP 问题解释：存在一个确定性算法 A，该算法在对问题Π的一个实例展示一个断言解时，它能在多项式时间内验证解的正确性。即如果断言解导致答案是 yes，就存在一种方法可以在多项式时间内验证这个解。

（3）3 着色问题是 NPC 问题。同时它也属于 NP 类问题。用定义证明即可。