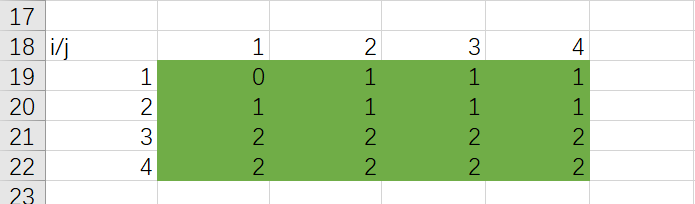
四、应用题（1个小题，共10分）  
1从A 地到D 地要铺设一条煤气管道,其中需经过两级中间站，两点之间的连线上的数字表示距离，如下图所示。问应该选择什么路线，使总距离最短？写出动态规划算法的向前递推（从最小规模开始）过程，并计算最优值及构造最优解。  
  
令f[i]表示到达点i的最短距离,并初始化除f[A]以外的所有点为正无穷。  
f[A] = 0;  
f[B1] = f[A] + 2 = 2;  
f[B2] = f[A] + 4 = 4;  
f[C1] = min(f[B1] + 3, f[B2] + 2) = min(2 + 3, 4 + 2) = 5;  
f[C2] = min(f[B1] + 3, f[B2] + 3) = min(2 + 3, 4 + 3) = 5;  
f[C3] = min(f[B2] + 1, f[B1] + 1) = min(2 + 1, 4 + 1) = 3;  
f[D] = min(f[C1] + 1 f[C2] + 3, f[C3] + 4) = min(5 + 1, 5 + 3, 3 + 4) = 6;  
  
  
  
2.使用回溯法解0/1背包问题：n=3，C=9，V={6,10,3}，W={3,4,4},其解空间有长度为3的0-1向量组成，要求用一棵完全二叉树表示其解空间（从根出发，左1右0），并画出其解空间树，计算其最优值及最优解。

3.用动态规划策略求解最长公共子序列问题：  
（1）给出计算最优值的递归方程。（4分）  
f[i][j] = max (f[i - 1][j], f[i][j - 1]);   if  A[i] != B[j])  
f[i][j] = max (f[i - 1][j - 1]  + 1, f[i][j]);  if A[i] == B[j]  
（2）给定两个序列X={B,A,B,A}，Y={A,B,C,B}，请采用动态规划策略求出其最长公共子序列，要求给出过程（画出求解表格）。（4分）



4.某国为了防御敌国的导弹袭击，发展出一种导弹拦截系统。 但是这种导弹拦截系统 有一个缺陷：虽然它的第一发炮弹能够到达任意的高度，但是以后每一发炮弹都不能高 于前一发的高度。 某天，雷达捕捉到敌国的导弹来袭。由于该系统还在试用阶段，所以只有一套系统，因此有可能不能拦截所有的导弹。现输入导弹依次飞来的高度，计算这套系统最多能拦截多少导弹及拦截的各导弹高度是多少，并写出推导过程。输入：389  154  155  302  299  170  158  65  320  296  111  222  555  68  189  
#include <iostream>  
#include <cstring>  
using namespace std;  
  
const int N = 1e5 + 10;  
  
int n;  
int a[N];  
int f[N];  
int main () {  
    int b;  
    while (cin >> b) {  
        a[n ++] = b;  
    }  
      
    int len = 0;  
    f[0] = -2e9;  
    for (int i = 0; i < n; i++)   
    {  
       int l = 0, r = len;  
        while (l < r) {  
            int mid = l + r + 1 >> 1;  
            if (f[mid] >= a[i]) l = mid;  
            else r = mid - 1;  
        }  
        len = max (len, r + 1);  
        f[r + 1] = a[i];  
    }  
      
    cout << len << endl;  
     len = 0;  
    memset (f, 0, sizeof f);  
    for (int i = 0; i < n; i ++) {  
        int l = 0, r = len;  
        while (l < r) {  
            int mid = l + r + 1 >> 1;  
            if (f[mid] < a[i]) l = mid;  
            else r = mid - 1;  
        }  
        len = max (len, r + 1);  
        f[r + 1] = a[i];  
    }  
    cout << len << endl;  
    return 0;  
}  
  
  
五、算法实现题(4个小题，每小题10分，共40分）  
注：在对算法实现时，编程语言不限，各算法需要用子函数实现，并在main函数中测试。  
1.把一个字符串（数组）进行逆序操作，如“abc"逆序为“cba” ，要求空间复杂度为O(1)。函数原型为：void  reverseString(char\* str);   
   
  
 void reverseString(vector<char>& s) {  
        int n = s.size();  
        for(int i = 0, j = n-1;i<=j; i++,j--){  
            char ch = s[i];  
            s[i] = s[j];  
            s[j] = ch;  
        }  
  
      
    }  
  
2.已知某数列符合规则：f(n)=f(n-1)+f(n-2)+f(n-3),且n为自然数，f(1)=1,f(2)=1，f(3)=1,那么请求第n个数是多少。要求时间复杂度为O(n)，空间复杂度为O(1).   
#include <iostream>  
#include <cstdio>  
#include <cstring>  
using namespace std;  
const int N = 1e5 + 10;  
int f (int n) {  
    if (n == 1 || n == 2 || n == 3) return 1;  
    return f (n - 1) + f (n - 2) + f(n - 3);  
}  
int main () {  
    int n;  
    cin >>n;  
    cout << f(n) <<endl;  
    return 0;  
}  
  
3.最小子段和问题：给出一个序列，其子段为该序列的一个连续的区间。例如：(1 -1 2 2 -3 -3 2 -4) ，( -3 -3 2 -4)为他的一个子段。给出一个序列，求出数字求和最小的一个子段。例如上面的序列， (-3 -3 2 -4)就是一个最小的子段，数字的总和为-8。  
#include <iostream>  
#include <cstdio>  
#include <cstring>  
using namespace std;  
const int N = 1e5 + 10;  
int n;  
int f[N];  
int minv = 0x3f3f3f3f, l = 1, r = n;  
int main () {  
    cin >> n;  
    for (int i = 1; i <= n; i ++) cin >> f[i];  
    for (int i = 1;  i<= n; i ++) {  
        int sum = 0;  
        for (int j = i; j <= n; j ++) {  
            sum += f[j];  
            if (sum < minv) {  
                minv = sum;  
                l = i, r = j;  
            }  
        }  
    }  
    for (int i = l; i <= r;  i ++) cout << f[i] << " ";  
    return 0;  
}  
  
4.键盘输入n个正整数（互不相等），找出其中和等于m的所有组合。输出顺序不限  
【输入样例】输入：5（m的值） 1 2 3 4 5（n个正整数）   
【输出样例】：1  4   
            2  3   
             5   
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e5 + 10;  
int n, m;  
int f[N];  
vector<int> ans;  
void dfs (int u, int sum) {  
    if (u > n) {  
        if (sum == m) {  
        for (auto t : ans) cout << t << " ";  
        puts("");}  
        return ;  
    }  
    dfs(u + 1, sum);  
    if (sum + f[u] <= m) {  
        ans.push\_back(f[u]);  
        dfs (u + 1, sum + f[u]);  
        ans.pop\_back();  
    }  
}  
int main () {  
    cin>> n >> m;  
    for (int i = 1; i <= n; i ++) cin >> f[i];  
  
    dfs(1, 0);  
    return 0;  
}  
5.使用递归算法对一个整数进行逆序转换，如1234逆序为4321。顶层函数原型为：  
   int reverseInteger(int a); 在具体实现时可增加子函数。   
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e5 + 10;  
int reverseInteger (int a) {  
    int ans = 0;  
    while(a) {  
        ans = ans \* 10 + a % 10;  
        a /= 10;  
    }  
    return ans;  
int main () {  
    int n;  
    cin >> n;  
    cout << reverseInteger(n) <<endl;  
    return 0;  
}  
6.已知某数列符合规则：f(n)=2*\*f(n-1)+3\**f(n-2)+f(n-3),且n为自然数，f(1)=1,f(2)=1，f(3)=1,那么请求第n个数是多少。要求时间复杂度为O(n)，空间复杂度为O(1).   
  
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e5 + 10;  
int f (int n) {  
    if (n == 1 || n == 2 || n == 3) return 1;  
    return f(n - 1) \* 2 +3 \* f(n - 2) + f(n - 3);  
}  
int main () {  
    int n;  
    cin >> n;  
    cout << f(n) <<endl;  
    return 0;  
}  
7.最大子段和问题：给出一个序列，其子段为该序列的一个连续的区间。例如：(1 -1 2 2 3 -3 4 -4) ，(2 2 3 -3 4)为他的一个子段。给出一个序列，求出数字求和最大的一个子段。例如上面的序列， (2 2 3 -3 4)就是一个最大的子段，数字的总和为8。  
  
  
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e5 + 10;  
int n;  
int f[N];  
int maxv = -0x3f3f3f3f, l, r;  
int main () {  
    cin >> n;  
    for (int i = 1; i <= n; i ++) cin >> f[i];  
    for (int i = 1; i <= n; i ++) {  
        int sum = 0;  
        for (int j = i; j <= n; j ++) {  
            sum += f[j];  
            if (sum > maxv) {  
                    maxv = sum;  
                    l = i, r = j;  
            }  
        }  
    }  
    for (int i = l; i <= r; i ++) cout << f[i] << " ";  
  
    return 0;  
}  
  
8.键盘输入n个正整数（互不相等），找出其中和大于m的所有组合。输出顺序不限  
【输入样例】输入：5（m的值） 1 2 3 4 5（n个正整数）   
【部分输出样例】：1  5   
                2  4   
                    3  5  
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e5 + 10  
int n, m;  
int f[N];  
vector<int> ans;  
void dfs (int u, int sum) {  
    if (u == n + 1) {  
        if (sum > m) {  
        for (int i = 0; i < ans.size(); i ++) cout << ans[i] << " ";  
        puts("");}  
        return ;  
    }  
    dfs (u + 1, sum);  
    ans.push\_back(f[u]);  
    dfs(u + 1, sum + f[u]);  
    ans.pop\_back();  
  
  
}  
int main () {  
    cin >> n >> m;  
    for (int i = 1; i <= n; i ++) cin >>f[i];  
  
    dfs (1, 0);  
    return 0;  
}  
9.超级楼梯问题：有一楼梯共M级，刚开始时你在第一级，若每次只能跨上一级或二级h或三级，要走上第M级，共有多少种走法？要求时间复杂度为O(n)，空间复杂度为O(1)。  
#include<iostream>  
using namespace std;  
int f (int n) {  
    if (n == 1) return 1;  
    if (n == 2) return 2;  
    if (n == 3) return 4;  
    return f (n - 1) + f(n - 2) + f(n - 3);  
}  
int main () {  
    int n;  
    cin >> n;  
    cout << f(n) <<endl;  
    return 0;  
}  
10请用递归算法判断一个整数数组是否是对称的，比如数组为{1,2,2,1}就是对称的。要求递归函数返回bool类型。  
  
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e5 + 10;  
int n;  
int a[N];  
bool check (int l, int r) {  
    if (l > r) return 1;  
    if (a[l] == a[r]) return check(l + 1, r - 1);  
    else return 0;  
}  
int main () {  
    cin >> n;  
    for (int i = 1; i <= n; i ++) cin >> a[i];  
    cout << check(1, n) << endl;  
    return 0;  
}  
12将正整数n表示成一系列正整数之和：n=n1+n2+…+nk，其中n1≥n2≥…≥nk≥1，k≥1。正整数n的这种表示称为正整数n的划分。输出正整数n的不同划分个数及具体的划分形式。  
例如正整数6有如下11种不同的划分：     
6；     
5+1；      
4+2，4+1+1；     
3+3，3+2+1，3+1+1+1；  
2+2+2，2+2+1+1，2+1+1+1+1；     
1+1+1+1+1+1  
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e5 + 10;  
int n;  
vector<int> ans;  
void dfs (int u, int sum) {  
    if (sum == n) {  
        cout << ans[0];  
        for (int i = 1; i < ans.size(); i ++) cout << "+" << ans[i];  
        puts("");  
        return ;  
    }  
    for (int i = u; i <= n; i ++) {  
        if (sum + i <= n) {  
            ans.push\_back(i);  
            dfs (i, sum + i);  
            ans.pop\_back();  
        }  
    }  
}  
int main () {  
    cin >> n;  
    dfs(1, 0);  
    return 0;  
}  
  
13请用分支限界法实现：  
八皇后问题：在8×8的国际象棋盘上，放置八个皇后，使任何一个皇后都不能吃掉另一个  
国际象棋规则中，皇后可以吃到任何一个与他在同一行、同一列或者同一斜线上的敌方棋子，所以八皇后问题的所有解满足：  
8个皇后都不在同一行、同一列，或者同一斜线上；或者任意行、列或者斜线上 有且仅有一个皇后。  
听说不考，这是最难的题  
14给定一个高度为 n 的“数字三角形”，其中第 i 行（1<=i<=n）有 i 个数。（例子如下图所示）   
1   
2  3  
4  5  6  
7  8  9  10  
初始时，你站在“数字三角形”的顶部，即第一行的唯一一个数上。每次移动，你可以选择移动到当前位置正下方或者当前位置右下方的位置上。即如果你在 (i,j)（表示你在第i行从左往右数第j个数上，下同），你可以选择移动到 (i+1,j) 或 (i+1,j+1)。  
你想让你经过的所有位置（包括起点和终点）的数字总和最大。求这个最大值。  
 #include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e3 + 10;  
int f[N][N];  
int a[N][N];  
int n;  
int main () {  
    cin >> n;  
    for (int i = 1; i <= n; i ++) {  
        for (int j = 1; j <= i; j ++)  
            cin >> a[i][j];  
    }  
    for (int i = 1; i <= n; i ++) {  
        for (int j= 1 ; j <= n; j ++) {  
            f[i][j]= max (f[i - 1][j], f[i - 1][j - 1]) +a[i][j];  
        }  
    }  
    int ans = 0;  
    for (int i = 1; i <= n ;i++)  
        ans = max (ans, f[n][i]);  
    cout << ans <<endl;  
    return 0;  
}  
15用回溯法解决：给定n个正整数wi和一个正整数m，在这n个正整数中找出一个子集，使得子集中的正整数之和等于m。  
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e3 + 10;  
int n, m;  
int w[N];  
vector<int> ans;  
bool dfs (int u, int sum) {  
    if (u > n) return 0;  
    if (sum == m) {  
        for (auto t : ans) cout << t << " ";  
        puts("");  
        return 1;  
    }  
  
    if (dfs(u + 1, sum)) return 1;  
    ans.push\_back(w[u]);  
    if(dfs(u + 1, sum + w[u])) return 1;  
    ans.pop\_back();  
    return 0;  
}  
int main () {  
    cin >> n >> m;  
    for (int i = 1; i <= n; i ++) cin >>w[i];  
    dfs (1, 0);  
    return 0;  
}  
16给定一个存放整数的数组，重新排列数组使得数组左边为偶数，右边为奇数。  
如对数组：1,2,3,4,5,6,7,8,9处理后，该数组变为：2,4,6,8,1,3,5,7,9。   
  
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
const int N = 1e3 + 10;  
int n;  
int f[N];  
int main () {  
    cin >>n;  
    for (int i = 1; i<= n; i ++) cin >> f[i];  
  
    int l = -1, r = n +1;  
    while (l < r) {  
        do l ++; while (f[l] %2 == 0);  
        do r --; while (f[r] & 1);  
        if (l < r) {  
            swap (f[l], f[r]);  
        }  
    }  
    for (int i = 1; i <= n; i ++) cout <<f[i] << " ";  
    return 0;  
}  
17编写函数double getRestul(int n)并测试，它的功能是：根据以下公式计算结果S。  
![算法题17](images/算法题17.png)  
#include <iostream>  
#include <cstdio>  
#include <cstring>  
#include <vector>  
using namespace std;  
double getResult (int n) {  
    double ans = 0;  
    bool f = 1;  
    for (double i = 1; i <= n \* 2 - 1; i ++) {  
        ans += 1 / i \* f;  
        f \*= -1;  
    }  
    return ans;  
}  
int main () {  
    double n;  
    cin >> n;  
    cout <<getResult(n) <<endl;  
    return 0;  
}