7-1 找鞍点

```c
#include <stdio.h>
int main()
{
    int n;
    int a[10][10] = {0};
    int max_line[10] = {0}, min_row[10] = {0};
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            scanf("%d", &a[i][j]);
    // 行最大值和列最小值可以同时查找
    for (int i = 0; i < n; i++)
    {
        max_line[i] = a[i][0];
        min_row[i] = a[0][i];
        for (int j = 0; j < n; j++)
        {
            if (max_line[i] < a[i][j])
                max_line[i] = a[i][j];
            if (min_row[i] > a[j][i])
                min_row[i] = a[j][i];
        }
    }
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            if (min_row[j] == a[i][j] && max_line[i] == a[i][j])
            {
                printf("%d %d\n", i, j);
                return 0;
            }
    printf("NONE");
    return 0;
}
```

7-2 螺旋方阵

每次大循环填充一层数字共 $4(n-1-2i)$ 个，总共循环 $\frac{n}{2}$ 次

$$\sum_{i=0}^{\frac{n}{2}-1} 4(n-1-2i) = n^2$$

```c
#include <stdio.h>
int main()
{
    int i, j, k, n, a[15][15];
    scanf("%d", &n);
    k = 1;
    for (i = 0; i < n / 2; i++) //按螺旋方阵的层数循环， n 为单数时 最后一层单独输出
    {
        for (j = i; j < n - 1 - i; j++)
            a[i][j] = k++;
        for (j = i; j < n - 1 - i; j++)
            a[j][n - 1 - i] = k++;
        for (j = n - i - 1; j > i; j--)
            a[n - i - 1][j] = k++;
        for (j = n - i - 1; j > i; j--)
            a[j][i] = k++;
    }
    if (n % 2 == 1)
        a[n / 2][n / 2] = k;
    for (i = 0; i < n; i++)
    {
        for (j = 0; j < n; j++)
            printf("%3d", a[i][j]);
        cout << endl;
    }
    return 0;
}
```

7-3 IP地址转换

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    string s;
    cin >> s;
    for (int i = 0; i < 4; i++)
    {
        int sum = 0;
        for (int j = 0; j < 8; j++)
            sum += (s[i * 8 + j] - 48) << (7 - j);
        printf("%d%c",sum, ".\n"[i==3]);
    }
    return 0;
}
```

7-4 二分法求多项式单根

```
#include <stdio.h>
#include <math.h>
float a0, a1, a2, a3;
float f(float x) { return a3 * x * x * x + a2 * x * x + a1 * x + a0; }
int main()
{
    float a, b;
    scanf("%f %f %f %f", &a3, &a2, &a1, &a0);
    scanf("%f %f", &a, &b);
    float mid;
    while (b - a > 0.0001)
    {
        if (f(a) == 0)
        {
            printf("%.2f", a);
            return 0;
        }
        if (f(b) == 0)
        {
            printf("%.2f", b);
            return 0;
        }
        mid = (a + b) / 2;
        if (f(mid) * f(a) > 0)
        {
            a = mid;
        }
        else
        {
            b = mid;
        }
    }
    printf("%.2f", mid);
    return 0;
}
```

7-5 猴子选大王

```
#include <stdio.h>
int main()
{
    int N, i, count = 0, k = 0, flag = 0;
    scanf("%d", &N);
    int n[N];
    for (i = 0; i < N; i++)
        n[i] = 1;
    while (k != N - 1)
        for (i = 0; i < N; i++)
            if (n[i] == 1)
            {
                flag = i;
                count++;
                if (count == 3)
                {
                    n[i] = 0;
                    k++;
                    count = 0;
                }
            }
    printf("%d", flag + 1);
    return 0;
}
```

7-6 N个数求和

```
#include <stdio.h>
#include <math.h>
using namespace std;
int gcd(int a, int b) { return b ? gcd(b, a % b) : a; }
int main()
{
    int n;
    scanf("%d", &n);
    int a, b;
    int p = 0, q = 1; // 总和为 p/q
    for (int i = 0; i < n; i++)
    {
        scanf("%d/%d", &a, &b);
        p = p * b + q * a;
        q = q * b;
        int r = gcd(abs(p), abs(q));
        p /= r;
        q /= r;
    }
    if (q == 1)
        printf("%d", p);
    else if (p > q)
        printf("%d %d/%d", p / q, p % q, q);
    else
        printf("%d/%d", p, q);
    return 0;
}
```

7-7 整数分解为若干项之和

```
#include <iostream>
#include <vector>
using namespace std;
int k = 0, n;
void dfs(int x, vector<int> ans, int st)
{
    if (x == 0)
    {
        cout << n << "=";
        for (int i = 0, l = ans.size(); i < l; i++)
            cout << ans[i] << (string[2]){"+", ""}[i == l - 1];
        cout << ";\n"[++k % 4 == 0 || ans[0] == n];
        return;
    }
    for (int i = st; i <= x; i++)
    {
        ans.push_back(i);
        dfs(x - i, ans, i);
        ans.pop_back();
    }
}
int main()
{
    cin >> n;
    dfs(n, {}, 1);
    return 0;
}
```

7-8 输出全排列

```
#include<iostream>
#include<vector>
using namespace std;
int n;
vector<int> num(10);
```

```
vector<int> arr(10);
void dfs(int step) {
    int i;
    if (step == n + 1) {
        for (int i = 1; i <= n; i++) {
            cout<<num[i];
        }
        cout<<endl;
        return;
    }
    for (i = 1; i <= n; i++) {
        if (arr[i] == 0) {
            num[step] = i;
            arr[i] = 1;
            dfs(step+1);
            arr[i] = 0;
        }
    }
}
int main() {
    cin>>n;
    dfs(1);
}
```

使用c++ STL函数`std::next_permutation`

```
#include <iostream>
#include <algorithm>
using namespace std;
int main()
{
    int a[10], n;
    cin >> n;
    for (int i = 0; i < n; i++)
        a[i] = i + 1;
    do
    {
        for (int i = 0; i < n; i++)
            cout << a[i];
        cout << endl;
    } while (next_permutation(a, a + n));
    return 0;
}
```

7-9 出栈序列的合法性

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int m, n, k;
    cin >> m >> n >> k;
    while (k--)
    {
        int a[n];
        stack<int> s;
        int x = 1;
        bool is_right = true;
        for (int j = 0; j < n; j++)
            cin >> a[j];
        for (int j = 0; j < n;)
        {
            while (x <= a[j])
            {
                s.push(x++);
                if (s.size() > m)
                    is_right = false;
            }
            if (s.top() != a[j])
            {
                is_right = false;
                break;
            }
            else
            {
                s.pop();
                j++;
            }
        }
        cout << (is_right ? "YES" : "NO") << endl;
    }
    return 0;
}
```

7-10 包装机

```
#include <iostream>
#include <stack>
#include <queue>
using namespace std;
const int MAX = 100 + 5;
int main()
{
    stack<char> s;
    queue<char> q[MAX];
    string t;
    int n, m, s_max, op;
    cin >> n >> m >> s_max;
    cin.get();
    for (int i = 1; i <= n; i++)
    {
        getline(cin, t);
        for (char c : t)
            q[i].push(c);
    }
    while (cin >> op && op != -1)
    {
        if (op == 0)
        {
            if (!s.empty())
            {
                cout << s.top();
                s.pop();
            }
        }
        else if (!q[op].empty())
        {
            if (s.size() >= s_max)
            {
                cout << s.top();
                s.pop();
            }
            s.push(q[op].front());
            q[op].pop();
        }
    }
    return 0;
}
```