动态规划

数位dp

```
#include<bits/stdc++.h>
#define int long long
using namespace std;
constexpr int MAXN = 24 + 10;
int a[MAXN], mod, f[MAXN][MAXN * 10][MAXN * 10];

int dfs(int pos, int sum, int cur, bool lead0, bool lim) {
    if (!pos)return !lead0 && sum == mod && cur == 0;
    int& now = f[pos][cur][sum];
    if (!lead0 && !lim && ~now)return now;
    int up = lim ? a[pos] : 9, res = 0;
    for (int i = 0;i <= up;++i)
        res += dfs(pos - 1, sum + i, (cur * 10 + i) % mod, lead0 && !i, lim && i == up);
    if (!lead0 && !lim)now = res;
    return res;
}

signed main() {
    ios::sync_with_stdio(false);
    cin.tie(0), cout.tie(0);
    int n;cin >> n;
    int len = 0;
    while (n)a[++len] = n % 10, n /= 10;
    int res = 0;
    for (int i = 1;i <= len * 9;++i) {
        mod = i;memset(f, -1, sizeof f);
        res += dfs(len, 0, 0, 1, 1);
    }
    cout << res;
    return 0;
}
```

状压dp

最短Hamilton路径

```
using namespace std;

const int N = 20,M = 1 << N;

int n;
int w[N][N];
int f[M][N];//第一维表示是否访问到该点的压缩状态，第二维是走到点j
            //f[i][j]表示状态为i并且到j的最短路径

int main(){
    cin>>n;
    for (int i = 0; i < n; i ++ )
        for (int j = 0; j < n; j ++ )//读入i到j的距离
            cin>>w[i][j];
    memset(f, 0x3f, sizeof f);
    f[1][0]=0;
    for (int i = 0; i < 1 << n; i ++ )//枚举压缩的状态
        for (int j = 0; j < n; j ++ )//枚举到0~j的点
            if(i >> j & 1)//该状态存在j点
                for (int k = 0; k < n; k ++ )//枚举从j倒数第二个点k
                    if(i >> k & 1)//倒数点k存在
                        f[i][j]=min(f[i][j],f[i-(1<<j)][k]+w[k][j]);//状态转移方程，在f[i][j]和状
    cout<<f[(1<<n)-1][n-1]<<endl;//输出状态全满也就是所有点都经过且到最后一个点的最短距离
    return 0;
}
```

状态转移方程:

```
f[i][j]=min(f[i][j],f[i-(1<<j)][k]+w[k][j]);
```

SOSdp 高维前缀和

子集向超集转移

```
for(int j = 0; j < n; j++)
    for(int i = 0; i < 1 << n; i++)
        if(i >> j & 1) f[i] += f[i ^ (1 << j)];
```

超集向子集转移

```
for(int j = 0; j < n; j++)
    for(int i = (1 << n) - 1; i >= 0 ; i--)
        if(!(i >> j & 1)) f[i] += f[i ^ (1 << j)]
```

汉明权重

```
for (int i = 0; (1<<i)-1 <= n; i++) {
    for (int x = (1<<i)-1, t; x <= n; t = x+(x&-x), x = x ? (t|((((t&-t)/(x&-x))>>1)-1)) : (n+1
        // todo
    }
}
```