

# 实验报告

## Strassen矩阵乘法核心算法实现

### 核心思想

Strassen算法优化矩阵乘法利用了分治的思想，将2个 $2^n$ 阶方阵分成8个 $2^{n-1}$ 阶矩阵进行运算

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \quad \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

之后通过8个子矩阵加、减、乘等运算计算出7个中间矩阵

$$\begin{aligned} M_1 &= A_{11} \times (B_{12} - B_{22}) \\ M_2 &= (A_{11} + A_{12}) \times B_{22} \\ M_3 &= (A_{21} + A_{22}) \times B_{11} \\ M_4 &= A_{22} \times (B_{21} - B_{11}) \\ M_5 &= (A_{11} + A_{22}) \times (B_{11} + B_{22}) \\ M_6 &= (A_{12} - A_{22}) \times (B_{21} + B_{22}) \\ M_7 &= (A_{11} - A_{21}) \times (B_{11} + B_{12}) \end{aligned}$$

之后通过7个中间矩阵计算出结果矩阵的4个子矩阵

$$\begin{aligned} C_{11} &= M_5 + M_4 - M_2 + M_6 \\ C_{12} &= M_1 + M_2 \\ C_{21} &= M_3 + M_4 \\ C_{22} &= M_1 + M_5 + M_3 + M_7 \end{aligned}$$

总共算法时间复杂度递推式为

$$T(n) = 7T\left(\frac{n}{2}\right) + O(n^2)$$

根据主定理可得算法总时间复杂度为 $O(n^{\log_2 7})$ 。算法主要思想是将分块乘法的需要8次乘法，通过加减法的组合减少到7次。所付出的代价是算法运行时间的常数较大，在娇小的数据规模中表现得不如朴素的矩阵乘法

### 代码实现

```
auto A11 = makeUp(A, size, 0, 0);
auto A12 = makeUp(A, size, 0, mid);
auto A21 = makeUp(A, size, mid, 0);
auto A22 = makeUp(A, size, mid, mid);
auto B11 = makeUp(B, size, 0, 0);
auto B12 = makeUp(B, size, 0, mid);
auto B21 = makeUp(B, size, mid, 0);
auto B22 = makeUp(B, size, mid, mid);

strassenMul(M1, A11, sub(B12, B22), depth + 1);
strassenMul(M2, add(A11, A12), B22, depth + 1);
```

```

strassenMul(M3, add(A21, A22), B11, depth + 1);
strassenMul(M4, A22, sub(B21, B11), depth + 1);
strassenMul(M5, add(A11, A22), add(B11, B22), depth + 1);
strassenMul(M6, sub(A12, A22), add(B21, B22), depth + 1);
strassenMul(M7, sub(A11, A21), add(B11, B12), depth + 1);

auto C11 = add(add(M5, M4), sub(M6, M2));
auto C12 = add(M1, M2);
auto C21 = add(M3, M4);
auto C22 = sub(add(M5, M1), add(M3, M7));

merge(C, C11, 0, 0);
merge(C, C12, 0, mid);
merge(C, C21, mid, 0);
merge(C, C22, mid, mid);

```

## 优化思路

### 优化1

#### 优化描述

Strassen算法优化矩阵乘法只适用于阶数为 $2^n$ 的方阵，局限性较大。在计算一般矩阵乘法时，我们需要找到不小于矩阵边长最大值的2的指数，将两个矩阵缺失的地方进行补零。

#### 代码实现

```

int len = std::max(n, std::max(m, p));
for (int cur = 1; ; cur *= 2)
    if (len <= cur) {
        size = cur;
        break;
    }

```

### 优化2

#### 优化描述

我们注意到Strassen算法常数较大，在矩阵阶数较小时我们选择朴素矩阵乘法直接计算而非递归继续计算，我们设置矩阵阶数为32而非2作为递归基。

#### 代码实现

```

if (size <= MIN) {
    C = mul(A, B, size, size, size);
    return;
}

```

## 优化3

### 优化描述

我们注意到当矩阵阶数较大时，矩阵中补零的数量较多，导致大量无效计算，我们对每一个矩阵打一个标记，当tag为true的时候代表矩阵为全零矩阵，当矩阵乘法中有一个为全零矩阵时，直接返回全零矩阵。

### 代码实现

```
if (A.selfCheck() | B.selfCheck()) {  
    C = Matrix(size);  
    return;  
}
```

## 优化4

### 优化描述

我们发现Strassen算法需要多次动态分配内存并释放，我们将二维数组压缩为一维数组并重载()运算符简化数组索引，以减少程序向堆中申请内存与释放的次数。

### 代码实现

```
struct Matrix {  
    int *matrix = nullptr;  
    int size;  
  
    inline explicit Matrix(int _size) : size(_size) {  
        matrix = new int[size * size];  
        std::fill(matrix, matrix + size * size, 0);  
    }  
  
    inline ~Matrix() {  
        delete[] matrix;  
    }  
  
    inline Matrix &operator=(const Matrix &M) {  
        size = M.size;  
        matrix = new int[size * size];  
        memcpy(matrix, M.matrix, size * size * sizeof(int));  
        return *this;  
    }  
  
    inline int &operator()(int x, int y) {  
        return matrix[x * size + y];  
    }  
  
    inline const int &operator()(int x, int y) const {  
        return matrix[x * size + y];  
    }  
}
```

```
[[nodiscard]] inline bool selfCheck() const {
    bool flag = true;
    for (int i = 0; i < size * size; i++)
        flag &= !matrix[i];
    return flag;
}
};
```

## 优化5

### 优化描述

我们与朴素的矩阵乘法对比，发现Strassen算法也可以利用并发编程进行优化。同时因为Strassen为递归实现，导致子线程的数量过多，我们限制子线程生成的个数，即限制递归深度来选择是否生成子线程。

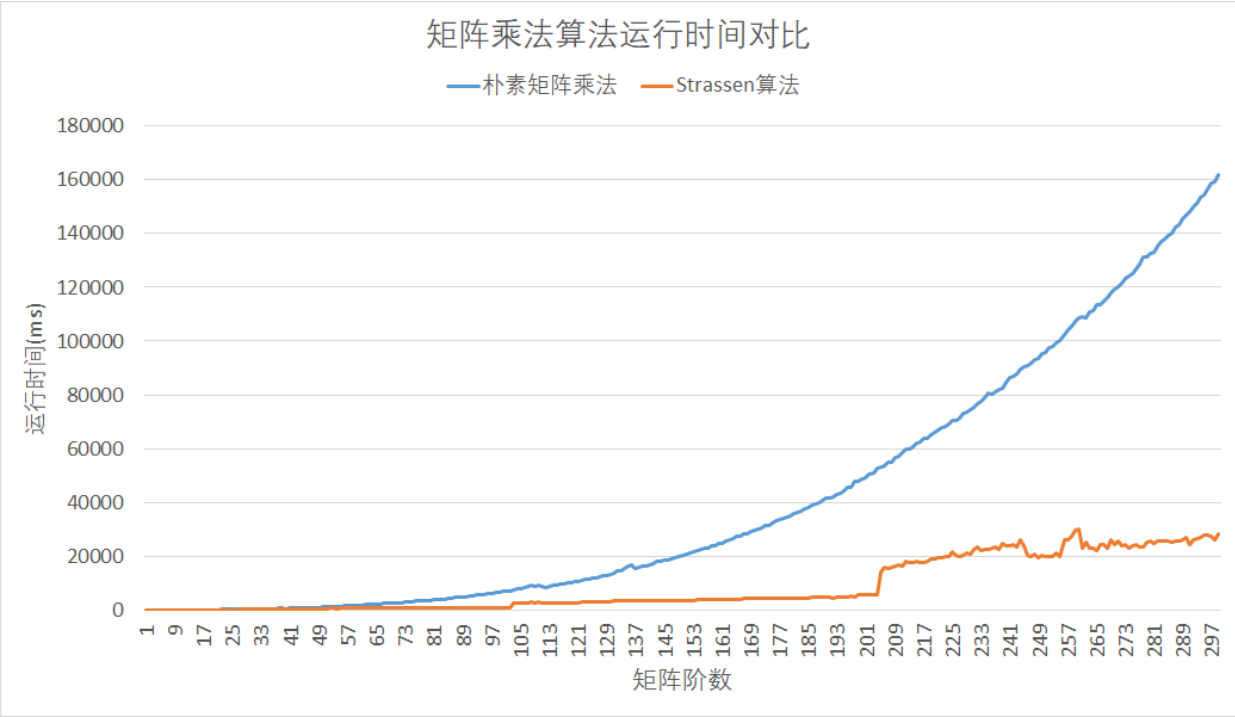
### 代码实现

```
if (depth <= 2) {
    std::thread t1(&) { strassenMul(M1, A11, sub(B12, B22), depth + 1); };
    std::thread t2(&) { strassenMul(M2, add(A11, A12), B22, depth + 1); };
    std::thread t3(&) { strassenMul(M3, add(A21, A22), B11, depth + 1); };
    std::thread t4(&) { strassenMul(M4, A22, sub(B21, B11), depth + 1); };
    std::thread t5(&) { strassenMul(M5, add(A11, A22), add(B11, B22), depth + 1);
};
    std::thread t6(&) { strassenMul(M6, sub(A12, A22), add(B21, B22), depth + 1);
};
    std::thread t7(&) { strassenMul(M7, sub(A11, A21), add(B11, B12), depth + 1);
};

    t1.join();
    t2.join();
    t3.join();
    t4.join();
    t5.join();
    t6.join();
    t7.join();
}
```

## 运行时间比较

我们通过测试不同算法下不同阶数下矩阵乘法运行时间，绘制以下折线图：发现在矩阵阶数较小时，因为Strassen算法的常数较大，与朴素矩阵乘法的差距较小；当矩阵阶数提升时，二者差距显著。



## YOJ通过截图

YOJ2.0

🏠 首页

📖 题库

📝 考试

📊 评测

📈 排名

📖 课程

完成题目数: 194 排名: 35 胡博文

编号	题目名称	状态	分数	总时间	内存	代码	提交者	提交时间
#674942	#786 矩阵乘法	✓ Accepted	100	159 ms	1360 KIB	cpp17 / 5393 B	胡博文(2021201780)	2023/3/1 15:19

下边这个缩略输入输出，老师同意可以拉开了看^\_^ (但考试时候不给看)

测试点 #0	✓ 2	得分: 100	用时: 9 ms	内存: 364 KIB
测试点 #1	✓ 2	得分: 100	用时: 9 ms	内存: 380 KIB
测试点 #2	✓ 2	得分: 100	用时: 8 ms	内存: 504 KIB
测试点 #3	✓ 2	得分: 100	用时: 8 ms	内存: 384 KIB
测试点 #4	✓ 2	得分: 100	用时: 11 ms	内存: 748 KIB
测试点 #5	✓ 2	得分: 100	用时: 8 ms	内存: 612 KIB
测试点 #6	✓ 2	得分: 100	用时: 14 ms	内存: 488 KIB
测试点 #7	✓ 2	得分: 100	用时: 19 ms	内存: 1360 KIB
测试点 #8	✓ 2	得分: 100	用时: 22 ms	内存: 1352 KIB
测试点 #9	✓ 2	得分: 100	用时: 25 ms	内存: 1336 KIB
测试点 #10	✓ 2	得分: 100	用时: 26 ms	内存: 1188 KIB

C++GCC 8.2.0

C++11GCC 8.2.0

C++17GCC 8.2.0

```
1 #include <iostream>
2 #include <random>
3 #include <ctime>
4 #include <cstring>
5 // #include <thread>
6
7 struct Matrix {
8     int *matrix = nullptr;
9     int size;
```

```
using namespace std;
const int N = 1000;
int n;
int a[N][N];
int b[N][N];
int c[N][N];
int d[N][N];
int e[N][N];
int f[N][N];
int g[N][N];
int h[N][N];
int i[N][N];
int j[N][N];
int k[N][N];
int l[N][N];
int m[N][N];
int n[N][N];
int o[N][N];
int p[N][N];
int q[N][N];
int r[N][N];
int s[N][N];
int t[N][N];
int u[N][N];
int v[N][N];
int w[N][N];
int x[N][N];
int y[N][N];
int z[N][N];
int aa[N][N];
int ab[N][N];
int ac[N][N];
int ad[N][N];
int ae[N][N];
int af[N][N];
int ag[N][N];
int ah[N][N];
int ai[N][N];
int aj[N][N];
int ak[N][N];
int al[N][N];
int am[N][N];
int an[N][N];
int ao[N][N];
int ap[N][N];
int aq[N][N];
int ar[N][N];
int as[N][N];
int at[N][N];
int au[N][N];
int av[N][N];
int aw[N][N];
int ax[N][N];
int ay[N][N];
int az[N][N];
int ba[N][N];
int bb[N][N];
int bc[N][N];
int bd[N][N];
int be[N][N];
int bf[N][N];
int bg[N][N];
int bh[N][N];
int bi[N][N];
int bj[N][N];
int bk[N][N];
int bl[N][N];
int bm[N][N];
int bn[N][N];
int bo[N][N];
int bp[N][N];
int bq[N][N];
int br[N][N];
int bs[N][N];
int bt[N][N];
int bu[N][N];
int bv[N][N];
int bw[N][N];
int bx[N][N];
int by[N][N];
int bz[N][N];
int ca[N][N];
int cb[N][N];
int cc[N][N];
int cd[N][N];
int ce[N][N];
int cf[N][N];
int cg[N][N];
int ch[N][N];
int ci[N][N];
int cj[N][N];
int ck[N][N];
int cl[N][N];
int cm[N][N];
int cn[N][N];
int co[N][N];
int cp[N][N];
int cq[N][N];
int cr[N][N];
int cs[N][N];
int ct[N][N];
int cu[N][N];
int cv[N][N];
int cw[N][N];
int cx[N][N];
int cy[N][N];
int cz[N][N];
int da[N][N];
int db[N][N];
int dc[N][N];
int dd[N][N];
int de[N][N];
int df[N][N];
int dg[N][N];
int dh[N][N];
int di[N][N];
int dj[N][N];
int dk[N][N];
int dl[N][N];
int dm[N][N];
int dn[N][N];
int do[N][N];
int dp[N][N];
int dq[N][N];
int dr[N][N];
int ds[N][N];
int dt[N][N];
int du[N][N];
int dv[N][N];
int dw[N][N];
int dx[N][N];
int dy[N][N];
int dz[N][N];
int ea[N][N];
int eb[N][N];
int ec[N][N];
int ed[N][N];
int ee[N][N];
int ef[N][N];
int eg[N][N];
int eh[N][N];
int ei[N][N];
int ej[N][N];
int ek[N][N];
int el[N][N];
int em[N][N];
int en[N][N];
int eo[N][N];
int ep[N][N];
int eq[N][N];
int er[N][N];
int es[N][N];
int et[N][N];
int eu[N][N];
int ev[N][N];
int ew[N][N];
int ex[N][N];
int ey[N][N];
int ez[N][N];
int fa[N][N];
int fb[N][N];
int fc[N][N];
int fd[N][N];
int fe[N][N];
int ff[N][N];
int fg[N][N];
int fh[N][N];
int fi[N][N];
int fj[N][N];
int fk[N][N];
int fl[N][N];
int fm[N][N];
int fn[N][N];
int fo[N][N];
int fp[N][N];
int fq[N][N];
int fr[N][N];
int fs[N][N];
int ft[N][N];
int fu[N][N];
int fv[N][N];
int fw[N][N];
int fx[N][N];
int fy[N][N];
int fz[N][N];
int ga[N][N];
int gb[N][N];
int gc[N][N];
int gd[N][N];
int ge[N][N];
int gf[N][N];
int gg[N][N];
int gh[N][N];
int gi[N][N];
int gj[N][N];
int gk[N][N];
int gl[N][N];
int gm[N][N];
int gn[N][N];
int go[N][N];
int gp[N][N];
int gq[N][N];
int gr[N][N];
int gs[N][N];
int gt[N][N];
int gu[N][N];
int gv[N][N];
int gw[N][N];
int gx[N][N];
int gy[N][N];
int gz[N][N];
int ha[N][N];
int hb[N][N];
int hc[N][N];
int hd[N][N];
int he[N][N];
int hf[N][N];
int hg[N][N];
int hh[N][N];
int hi[N][N];
int hj[N][N];
int hk[N][N];
int hl[N][N];
int hm[N][N];
int hn[N][N];
int ho[N][N];
int hp[N][N];
int hq[N][N];
int hr[N][N];
int hs[N][N];
int ht[N][N];
int hu[N][N];
int hv[N][N];
int hw[N][N];
int hx[N][N];
int hy[N][N];
int hz[N][N];
int ia[N][N];
int ib[N][N];
int ic[N][N];
int id[N][N];
int ie[N][N];
int if[N][N];
int ig[N][N];
int ih[N][N];
int ii[N][N];
int ij[N][N];
int ik[N][N];
int il[N][N];
int im[N][N];
int in[N][N];
int io[N][N];
int ip[N][N];
int iq[N][N];
int ir[N][N];
int is[N][N];
int it[N][N];
int iu[N][N];
int iv[N][N];
int iw[N][N];
int ix[N][N];
int iy[N][N];
int iz[N][N];
int ja[N][N];
int jb[N][N];
int jc[N][N];
int jd[N][N];
int je[N][N];
int jf[N][N];
int jg[N][N];
int jh[N][N];
int ji[N][N];
int jj[N][N];
int jk[N][N];
int jl[N][N];
int jm[N][N];
int jn[N][N];
int jo[N][N];
int jp[N][N];
int jq[N][N];
int jr[N][N];
int js[N][N];
int jt[N][N];
int ju[N][N];
int jv[N][N];
int jw[N][N];
int jx[N][N];
int jy[N][N];
int jz[N][N];
int ka[N][N];
int kb[N][N];
int kc[N][N];
int kd[N][N];
int ke[N][N];
int kf[N][N];
int kg[N][N];
int kh[N][N];
int ki[N][N];
int kj[N][N];
int kk[N][N];
int kl[N][N];
int km[N][N];
int kn[N][N];
int ko[N][N];
int kp[N][N];
int kq[N][N];
int kr[N][N];
int ks[N][N];
int kt[N][N];
int ku[N][N];
int kv[N][N];
int kw[N][N];
int kx[N][N];
int ky[N][N];
int kz[N][N];
int la[N][N];
int lb[N][N];
int lc[N][N];
int ld[N][N];
int le[N][N];
int lf[N][N];
int lg[N][N];
int lh[N][N];
int li[N][N];
int lj[N][N];
int lk[N][N];
int ll[N][N];
int lm[N][N];
int ln[N][N];
int lo[N][N];
int lp[N][N];
int lq[N][N];
int lr[N][N];
int ls[N][N];
int lt[N][N];
int lu[N][N];
int lv[N][N];
int lw[N][N];
int lx[N][N];
int ly[N][N];
int lz[N][N];
int ma[N][N];
int mb[N][N];
int mc[N][N];
int md[N][N];
int me[N][N];
int mf[N][N];
int mg[N][N];
int mh[N][N];
int mi[N][N];
int mj[N][N];
int mk[N][N];
int ml[N][N];
int mm[N][N];
int mn[N][N];
int mo[N][N];
int mp[N][N];
int mq[N][N];
int mr[N][N];
int ms[N][N];
int mt[N][N];
int mu[N][N];
int mv[N][N];
int mw[N][N];
int mx[N][N];
int my[N][N];
int mz[N][N];
int na[N][N];
int nb[N][N];
int nc[N][N];
int nd[N][N];
int ne[N][N];
int nf[N][N];
int ng[N][N];
int nh[N][N];
int ni[N][N];
int nj[N][N];
int nk[N][N];
int nl[N][N];
int nm[N][N];
int nn[N][N];
int no[N][N];
int np[N][N];
int nq[N][N];
int nr[N][N];
int ns[N][N];
int nt[N][N];
int nu[N][N];
int nv[N][N];
int nw[N][N];
int nx[N][N];
int ny[N][N];
int nz[N][N];
int oa[N][N];
int ob[N][N];
int oc[N][N];
int od[N][N];
int oe[N][N];
int of[N][N];
int og[N][N];
int oh[N][N];
int oi[N][N];
int oj[N][N];
int ok[N][N];
int ol[N][N];
int om[N][N];
int on[N][N];
int oo[N][N];
int op[N][N];
int oq[N][N];
int or[N][N];
int os[N][N];
int ot[N][N];
int ou[N][N];
int ov[N][N];
int ow[N][N];
int ox[N][N];
int oy[N][N];
int oz[N][N];
int pa[N][N];
int pb[N][N];
int pc[N][N];
int pd[N][N];
int pe[N][N];
int pf[N][N];
int pg[N][N];
int ph[N][N];
int pi[N][N];
int pj[N][N];
int pk[N][N];
int pl[N][N];
int pm[N][N];
int pn[N][N];
int po[N][N];
int pp[N][N];
int pq[N][N];
int pr[N][N];
int ps[N][N];
int pt[N][N];
int pu[N][N];
int pv[N][N];
int pw[N][N];
int px[N][N];
int py[N][N];
int pz[N][N];
int qa[N][N];
int qb[N][N];
int qc[N][N];
int qd[N][N];
int qe[N][N];
int qf[N][N];
int qg[N][N];
int qh[N][N];
int qi[N][N];
int qj[N][N];
int qk[N][N];
int ql[N][N];
int qm[N][N];
int qn[N][N];
int qo[N][N];
int qp[N][N];
int qq[N][N];
int qr[N][N];
int qs[N][N];
int qt[N][N];
int qu[N][N];
int qv[N][N];
int qw[N][N];
int qx[N][N];
int qy[N][N];
int qz[N][N];
int ra[N][N];
int rb[N][N];
int rc[N][N];
int rd[N][N];
int re[N][N];
int rf[N][N];
int rg[N][N];
int rh[N][N];
int ri[N][N];
int rj[N][N];
int rk[N][N];
int rl[N][N];
int rm[N][N];
int rn[N][N];
int ro[N][N];
int rp[N][N];
int rq[N][N];
int rr[N][N];
int rs[N][N];
int rt[N][N];
int ru[N][N];
int rv[N][N];
int rw[N][N];
int rx[N][N];
int ry[N][N];
int rz[N][N];
int sa[N][N];
int sb[N][N];
int sc[N][N];
int sd[N][N];
int se[N][N];
int sf[N][N];
int sg[N][N];
int sh[N][N];
int si[N][N];
int sj[N][N];
int sk[N][N];
int sl[N][N];
int sm[N][N];
int sn[N][N];
int so[N][N];
int sp[N][N];
int sq[N][N];
int sr[N][N];
int ss[N][N];
int st[N][N];
int su[N][N];
int sv[N][N];
int sw[N][N];
int sx[N][N];
int sy[N][N];
int sz[N][N];
int ta[N][N];
int tb[N][N];
int tc[N][N];
int td[N][N];
int te[N][N];
int tf[N][N];
int tg[N][N];
int th[N][N];
int ti[N][N];
int tj[N][N];
int tk[N][N];
int tl[N][N];
int tm[N][N];
int tn[N][N];
int to[N][N];
int tp[N][N];
int tq[N][N];
int tr[N][N];
int ts[N][N];
int tt[N][N];
int tu[N][N];
int tv[N][N];
int tw[N][N];
int tx[N][N];
int ty[N][N];
int tz[N][N];
int ua[N][N];
int ub[N][N];
int uc[N][N];
int ud[N][N];
int ue[N][N];
int uf[N][N];
int ug[N][N];
int uh[N][N];
int ui[N][N];
int uj[N][N];
int uk[N][N];
int ul[N][N];
int um[N][N];
int un[N][N];
int uo[N][N];
int up[N][N];
int uq[N][N];
int ur[N][N];
int us[N][N];
int ut[N][N];
int uu[N][N];
int uv[N][N];
int uw[N][N];
int ux[N][N];
int uy[N][N];
int uz[N][N];
int va[N][N];
int vb[N][N];
int vc[N][N];
int vd[N][N];
int ve[N][N];
int vf[N][N];
int vg[N][N];
int vh[N][N];
int vi[N][N];
int vj[N][N];
int vk[N][N];
int vl[N][N];
int vm[N][N];
int vn[N][N];
int vo[N][N];
int vp[N][N];
int vq[N][N];
int vr[N][N];
int vs[N][N];
int vt[N][N];
int vu[N][N];
int vv[N][N];
int vw[N][N];
int vx[N][N];
int vy[N][N];
int vz[N][N];
int wa[N][N];
int wb[N][N];
int wc[N][N];
int wd[N][N];
int we[N][N];
int wf[N][N];
int wg[N][N];
int wh[N][N];
int wi[N][N];
int wj[N][N];
int wk[N][N];
int wl[N][N];
int wm[N][N];
int wn[N][N];
int wo[N][N];
int wp[N][N];
int wq[N][N];
int wr[N][N];
int ws[N][N];
int wt[N][N];
int wu[N][N];
int wv[N][N];
int ww[N][N];
int wx[N][N];
int wy[N][N];
int wz[N][N];
int xa[N][N];
int xb[N][N];
int xc[N][N];
int xd[N][N];
int xe[N][N];
int xf[N][N];
int xg[N][N];
int xh[N][N];
int xi[N][N];
int xj[N][N];
int xk[N][N];
int xl[N][N];
int xm[N][N];
int xn[N][N];
int xo[N][N];
int xp[N][N];
int xq[N][N];
int xr[N][N];
int xs[N][N];
int xt[N][N];
int xu[N][N];
int xv[N][N];
int xw[N][N];
int xx[N][N];
int xy[N][N];
int xz[N][N];
int ya[N][N];
int yb[N][N];
int yc[N][N];
int yd[N][N];
int ye[N][N];
int yf[N][N];
int yg[N][N];
int yh[N][N];
int yi[N][N];
int yj[N][N];
int yk[N][N];
int yl[N][N];
int ym[N][N];
int yn[N][N];
int yo[N][N];
int yp[N][N];
int yq[N][N];
int yr[N][N];
int ys[N][N];
int yt[N][N];
int yu[N][N];
int yv[N][N];
int yw[N][N];
int yx[N][N];
int yy[N][N];
int yz[N][N];
int za[N][N];
int zb[N][N];
int zc[N][N];
int zd[N][N];
int ze[N][N];
int zf[N][N];
int zg[N][N];
int zh[N][N];
int zi[N][N];
int zj[N][N];
int zk[N][N];
int zl[N][N];
int zm[N][N];
int zn[N][N];
int zo[N][N];
int zp[N][N];
int zq[N][N];
int zr[N][N];
int zs[N][N];
int zt[N][N];
int zu[N][N];
int zv[N][N];
int zw[N][N];
int zx[N][N];
int zy[N][N];
int zz[N][N];
```