**Codes**

Table of Contents

[Problem 1 Anurag 1-2](#_Toc30821394)

[Problem 2 Balloon 2-6](#_Toc30821395)

[Problem 3 Bicoloring 3-9](#_Toc30821396)

[Problem 4 Cycle Finding 4-12](#_Toc30821397)

[Problem 5 Equation 5-14](#_Toc30821398)

[Problem 6 Flip Bit 6-17](#_Toc30821399)

[Problem 7 Galaxy 2/Wormhole 7-20](#_Toc30821400)

[Problem 8 Marathon 8-23](#_Toc30821401)

[Problem 9 Probability 9-25](#_Toc30821402)

[Problem 10 Research 10-27](#_Toc30821403)

[Problem 11 River 11-30](#_Toc30821404)

[Problem 12 Rock 12-33](#_Toc30821405)

[Problem 13 Spaceship 13-36](#_Toc30821406)

[Problem 14 Travelling Salesman Problem (TSP) 14-39](#_Toc30821407)

# Anurag

enter image description here - 1 enter image description here - 2 enter image description here - 3 enter image description here - 4 enter image description here - 5 enter image description here - 6 enter image description here - 7

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 2  5 6 2 1 3  0 0 5 3 6 0  0 0 2 0 2 0  3 3 1 3 7 0  0 0 0 0 0 0  0 0 0 0 0 0  5 6 2 2 6  3 0 0 0 0 3  2 0 0 0 0 6  1 3 1 1 3 1  2 0 2 0 0 2  0 0 4 3 1 1 | 5  15 |

#include <stdio.h>

#define MAX 1003

typedef struct nodeStruct

{

    int x, y, l;

} Node;

Node Q[MAX \* MAX];

int front, rear;

int T, Case, Ans;

int N, M, X, Y, L;

int grid[MAX][MAX];

int vis[MAX][MAX];

void initQ()

{

    front = rear = 0;

}

int isEmpty()

{

    return (front == rear);

}

void push(int x, int y, int l)

{

    Q[rear].x = x;

    Q[rear].y = y;

    Q[rear++].l = l;

}

Node pop()

{

    return Q[front++];

}

int goUp(int i, int j)

{

    int x = grid[i][j];

    return (1 == x || 2 == x || 4 == x || 7 == x);

}

int goDown(int i, int j)

{

    int x = grid[i][j];

    return (1 == x || 2 == x || 5 == x || 6 == x);

}

int goLeft(int i, int j)

{

    int x = grid[i][j];

    return (1 == x || 3 == x || 6 == x || 7 == x);

}

int goRight(int i, int j)

{

    int x = grid[i][j];

    return (1 == x || 3 == x || 4 == x || 5 == x);

}

void readcase()

{

    int i, j;

    scanf("%d %d %d %d %d", &N, &M, &X, &Y, &L);

    for (i = 0; i < N; i++)

        for (j = 0; j < M; j++)

        {

            scanf("%d", &grid[i][j]);

            vis[i][j] = 0;

        }

}

void solvecase()

{

    Ans = 0;

    initQ();

    if (grid[X][Y])

    {

        vis[X][Y] = 1;

        printf("pipe\n");

        push(X, Y, L);

    }

    while (0 == isEmpty())

    {

        Ans++;

        Node node = pop();

        int x = node.x;

        int y = node.y;

        int l = node.l;

        if (l > 1)

        {

            if (x - 1 >= 0 && goUp(x, y) && goDown(x - 1, y) && 0 == vis[x - 1][y])

            {

                push(x - 1, y, l - 1);

                vis[x - 1][y] = 1;

            }

            if (x + 1 < N && goDown(x, y) && goUp(x + 1, y) && 0 == vis[x + 1][y])

            {

                push(x + 1, y, l - 1);

                vis[x + 1][y] = 1;

            }

            if (y - 1 >= 0 && goLeft(x, y) && goRight(x, y - 1) && 0 == vis[x][y - 1])

            {

                push(x, y - 1, l - 1);

                vis[x][y - 1] = 1;

            }

            if (y + 1 < M && goRight(x, y) && goLeft(x, y + 1) && 0 == vis[x][y + 1])

            {

                push(x, y + 1, l - 1);

                vis[x][y + 1] = 1;

            }

        }

    }

}

void printcase()

{

    printf("%d\n", Ans);

}

int main()

{

    freopen("anurag\_input.txt", "r", stdin);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

Common Problems

* 0 == vis[x][y] is not put in solvecase
* initQ is not initialized in solvecase

# Balloon

|  |  |
| --- | --- |
| Input | Output |
| 10  7  45 3 92 8 14 30 7  10  1 10 2 9 3 8 4 7 5 6  10  2 9 3 8 4 7 5 6 12 11  10  1 10 2 9 3 8 4 7 5 6  10  2 9 3 8 4 7 5 6 12 11  10  1 10 2 9 3 8 4 7 5 6  10  2 9 3 8 4 7 5 6 12 11  10  1 10 2 9 3 8 4 7 5 6  10  2 9 3 8 4 7 5 6 12 11  10  1 10 2 9 3 8 4 7 5 6  10  2 9 3 8 4 7 5 6 12 11 | #1 9898  #2 498  #3 608  #4 498  #5 608  #6 498  #7 608  #8 498  #9 608  #10 498 |

Common Problems

* forget to unvisit in solve function
* solve function is void type

/\*

    6.01m

    ->ok

    ->confusing solve function

\*/

#include <stdio.h>

#define MAX 103

int T, Case, max;

int N, a[MAX], vis[MAX];

void readcase()

{

    int i, j;

    scanf("%d", &N);

    for (i = 0; i < N; i++)

    {

        scanf("%d", &a[i]);

        vis[i] = 0;

    }

}

int Left(int i)

{

    int j;

    for (j = i - 1; j >= 0; j--)

        if (0 == vis[j])

            return a[j];

    return 0;

}

int Right(int i)

{

    int j;

    for (j = i + 1; j < N; j++)

        if (0 == vis[j])

            return a[j];

    return 0;

}

void solve(int score)

{

    int j;

    for (j = 0; j < N; j++)

        if (0 == vis[j])

        {

            vis[j] = 1;

            int left = Left(j);

            int right = Right(j);

            if (left && right)

                solve(score + left \* right);

            else if (left)

                solve(score + left);

            else if (right)

                solve(score + right);

            if (score > max)

                max = score;

            vis[j] = 0;

        }

}

void solvecase()

{

    max = 0;

    solve(0);

}

void printcase()

{

    printf("#%d %d\n", Case, max);

}

int main()

{

    freopen("balloon\_input.txt", "r", stdin);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

# Bicoloring

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 3  3  0 1  1 2  2 0  3  2  0 1  1 2  9  8  0 1  0 2  0 3  0 4  0 5  0 6  0 7  0 8  0 | NOT BICOLORABLE.  BICOLORABLE.  BICOLORABLE. |

#include <stdio.h>

#define SIZEN 202

int N;

int L;

int Color[SIZEN];

int Path[SIZEN][SIZEN];

int Ans;

void initCase()

{

    int x, y;

    for (x = 0; x < N; x++)

    {

        Color[x] = 0;

        for (y = 0; y < N; y++)

        {

            Path[x][y] = 0;

        }

    }

    Ans = 1;

}

void readCase()

{

    int x, y;

    scanf("%d", &L);

    while (L--)

    {

        scanf("%d %d", &x, &y);

        Path[x][y] = Path[y][x] = 1;

    }

}

void solve(int x, int c)

{

    int y;

    if (0 == Color[x])

    {

        Color[x] = c;

        for (y = 0; y < N; y++)

        {

            if (Path[x][y])

            {

                if (1 == c)

                    solve(y, 2);

                else

                    solve(y, 1);

            }

        }

    }

    else if (c != Color[x])

    {

        Ans = 0;

    }

}

void solveCase()

{

    int x;

    for (x = 0; x < N; x++)

        if (0 == Color[x])

        {

            solve(x, 1);

        }

}

void printCase()

{

    if (1 == Ans)

        printf("BICOLORABLE.\n");

    else

        printf("NOT BICOLORABLE.\n");

}

int main()

{

    while (1 == scanf("%d", &N))

    {

        if (0 == N)

            break;

        initCase();

        readCase();

        solveCase();

        printCase();

    }

    return 0;

}

# Cycle Finding

#include<stdio.h>//cycle

int N;

int edge;

int grid[100][100];

int color[100];

void init\_grid()

{

    for (int i = 0; i < 100; i++)

    {

        color[i] = 0;

        for (int j = 0; j < 100; j++)

            grid[i][j] = 0;

    }

}

void input()

{

    int x, y;

    scanf("%d%d", &N, &edge);

    for (int i = 0; i < edge; i++)

    {

        scanf("%d%d", &x, &y);

        grid[x][y] = 1;

        grid[y][x] = 1;

    }

}

int solve(int x,int c)

{

    if (color[x] != 0 && color[x] != c)

        return color[x];

    color[x] = c;

    int r;

    for (int j = 0; j < N; j++)

    {

        if (grid[x][j])

        {

            grid[x][j] = 0;

            grid[j][x] = 0;

            r = solve(j, c + 1);

            if (r != -1)

                return r;

        }

    }

    color[x] = 0;

    return -1;

}

void cycle()

{

    int ans;

    ans = solve(0, 1);

    if (ans!=-1)

    {

        for (int i = 0; i < N; i++)

        {

            if (color[i] >= ans)

                printf("%d ", i);

        }

        printf("\n");

    }

}

int main()

{

    int test;

    scanf("%d", &test);

    while (test--)

    {

        input();

        cycle();

    }

}

# Equation

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 50  54 65 0 339442840066549  15 80 0 45842754826580  78 45 25 836757564541231  88 33 24 527978013399  51 16 58 3105320517669  61 80 83 497079350504  37 96 0 284897313181813  77 8 77 28334864824638  56 8 18 174189973030  100 71 94 40938910354  11 66 0 62753369521491  96 38 90 2762433039070  14 47 0 117910384188861  15 39 60 156214902752286  68 74 40 328229694194  21 5 86 18353978584672  35 28 69 10400966777606  6 78 56 219792837965490  76 13 76 1177553093491  6 49 70 6992814108232  61 62 0 564376970680067  4 10 41 848029399675  91 83 14 983867763892409  92 96 0 631996786986384  85 45 5 6397047433245  97 79 11 1284654780560  53 91 28 75310997439894  98 75 27 3493818184308  81 88 32 281861870723460  70 22 73 85988028498581  59 77 30 967273531145182  95 21 28 830681683510929  47 20 58 2018726961865  63 19 0 165158098688296  21 10 0 42168956750095  92 88 22 9144564727934  4 4 0 10053277920772  88 5 97 101490774178797  40 16 43 102331619083877  84 49 82 22034279719958  36 11 68 70122917029243  85 42 9 113171095896586  85 75 92 6884484774438  53 68 17 3084099836854  86 83 89 65227648308667  77 72 0 188746165488048  97 70 41 286266441166368  91 76 0 406113649891788  68 64 96 60086786066792  31 49 0 11863959055936 | #1 6285978519716  #2 3056183654956  #3 32227  #4 2802  #5 3769  #6 1816  #7 7699927383217  #8 7166  #9 2131  #10 758  #11 5704851774507  #12 3131  #13 8422170299107  #14 13757  #15 2017  #16 5976  #17 5322  #18 15774  #19 2493  #20 4640  #21 9252081486529  #22 2745  #23 41267  #24 6869530293300  #25 10856  #26 4888  #27 13907  #28 5058  #29 20652  #30 10561  #31 31828  #32 30957  #33 3265  #34 2621557122028  #35 2008045559515  #36 7463  #37 2513319480165  #38 10152  #39 13351  #40 6453  #41 10103  #42 23254  #43 4214  #44 5661  #45 9016  #46 2451248902416  #47 19113  #48 4462787361424  #49 8554  #50 382708356602 |

#include <stdio.h>

#define L 2.71828

// X = A\*N + B\*log(N) + C\*N\*N\*N

// Input A B C X

// Output N

// 1 <= A <= 100

// 1 <= B <= 100

// 0 <= C <= 100

// 0 <= X <= 10^15

// N must be integer

// 0 <= N <= 10^13

#define MAXC 10000000000000

#define MINC 100000

int log(long long n)

{

    double d = (double)n;

    int value = 0;

    while (d >= L)

    {

        d /= L;

        value++;

    }

    return value;

}

long long a, b, c, x, n;

int T, Case;

void readcase()

{

    scanf("%lld %lld %lld %lld", &a, &b, &c, &x);

}

void solvecase()

{

    long long low, mid, high, result;

    low = n = 0;

    if (0 == c)

        high = MAXC;

    else

        high = MINC;

    while (low <= high)

    {

        mid = low + (high - low) / 2;

        result = a \* mid + b \* log(mid) + c \* mid \* mid \* mid;

        if (x > result)

            low = mid + 1;

        else if (x < result)

            high = mid - 1;

        else

        {

            n = mid;

            break;

        }

    }

}

void printcase()

{

    printf("#%d %lld\n", Case, n);

}

int main()

{

    freopen("equation\_input.txt", "r", stdin);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

Mistakes:

* Didn’t print in %lld format N

# Flip Bit

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 5  5 3 3  1 1 1  0 0 0  0 0 0  1 1 1  0 0 0  5 4 4  1 0 0 0  1 1 1 0  1 1 1 0  1 0 0 0  1 0 0 0  5 5 3  1 1 1 0 0  1 0 1 1 1  1 1 1 0 0  1 0 1 1 1  1 1 1 0 0  5 2 1  0 1  0 1  0 1  0 1  0 1  5 7 2  0 0 0 1 1 0 1  1 1 1 0 1 1 1  0 0 0 1 1 0 1  1 1 1 0 1 1 1  0 0 0 1 1 0 1 | #1 3  #2 0  #3 2  #4 5  #5 0 |

/\*

    5.02m

    4.54m

\*/

#include <stdio.h>

#define MAX 101

int bit[MAX][21];

int zero[MAX];

int freq[MAX];

int val[MAX];

int T, Case, max;

int n, m, k;

void readcase()

{

    int i, j;

    scanf("%d %d %d", &n, &m, &k);

    for (i = 0; i < n; i++)

        for (j = 0; j < m; j++)

            scanf("%d", &bit[i][j]);

}

void solvecase()

{

    int i, j;

    max = 0;

    for (i = 0; i < n; i++)

    {

        val[i] = zero[i] = 0;

        for (j = 0; j < m; j++)

        {

            val[i] = (val[i] << 1) + bit[i][j];

            if (0 == bit[i][j])

                zero[i]++;

        }

        for (j = i - 1; j >= 0; j--)

            if (val[i] == val[j])

            {

                freq[i] = freq[j] + 1;

                break;

            }

        if (j < 0)

            freq[i] = 1;

        if (zero[i] <= k && (k - zero[i]) % 2 == 0)

            if (freq[i] > max)

                max = freq[i];

    }

}

void printcase()

{

    printf("#%d %d\n", Case, max);

}

int main()

{

    freopen("flip\_input.txt", "r", stdin);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

# Galaxy 2/Wormhole

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 5  1 0 4 1  4  5 5 9 1 3  7 0 4 7 5  3 1 0 2 3  7 4 4 6 5    18 16 13 5  4  4 3 19 5 5  13 3 18 3 7  16 11 6 15 1  0 15 8 13 5    4 9 2 10  4  4 24 1 22 11  28 23 24 0 15  17 12 2 17 4  16 4 29 2 12    13 7 9 8  5  13 19 4 36 15  20 39 38 38 4  12 6 16 18 12  17 21 0 30 20  0 36 19 10 2    28 12 5 39  2  2 18 35 39 3  25 12 18 15 21 | #1 4  #2 16  #3 3  #4 5  #5 50 |

#include <stdio.h>

#define MAX 1001

#define inf (MAX \* MAX)

int T, Case, min;

int x1[5], y1[5], x2[5], y2[5], dist[5], vis[5];

int sx, sy, dx, dy, N;

int Abs(int a, int b)

{

    return a > b ? a - b : b - a;

}

int Dist(int x1, int x2, int y1, int y2)

{

    return Abs(x1, x2) + Abs(y1, y2);

}

void readcase()

{

    int i;

    scanf("%d %d %d %d", &sx, &sy, &dx, &dy);

    scanf("%d", &N);

    for (i = 0; i < N; i++)

    {

        scanf("%d %d %d %d %d", &x1[i], &y1[i], &x2[i], &y2[i], &dist[i]);

        vis[i] = 0;

    }

}

void solve(int x, int y, int cost)

{

    int i, left, right;

    if (cost >= min)

        return;

    int total = Dist(x, dx, y, dy) + cost;

    if (total < min)

        min = total;

    for (i = 0; i < N; i++)

        if (0 == vis[i])

        {

            vis[i] = 1;

            left = Dist(x, x1[i], y, y1[i]);

            right = Dist(x, x2[i], y, y2[i]);

            if (left < right)

                solve(x2[i], y2[i], cost + left + dist[i]);

            else

                solve(x1[i], y1[i], cost + right + dist[i]);

            vis[i] = 0;

        }

}

void solvecase()

{

    min = inf;

    solve(sx, sy, 0);

}

void printcase()

{

    printf("#%d %d\n", Case, min);

}

int main()

{

    freopen("galaxy\_input.txt", "r", stdin);

    freopen("test.txt", "w", stdout);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

# Marathon

1 lok er 1 km jete mm:ss time lage and ei time e se E energy khoros kore. take 5ta time set dea thakbe, tar energy level and take koto km jete hobe seta dea thakbe. akon ber korte hobe j se minimum koto time e D km jete pare.

Cons:

H, D  H is his energy level, D is koto km jete hobe. H > 10 && H <= 6000, D <= 40

I/O

1

130 30 // 130 energy, 30Km distance

5 30 5 // 5 min, 30 seconds, cost 5 energe for 1 KM.

5 2 4

5 20 7

4 7 10

4 30 9

#1 149 56 // #1 case number, 149 min, 56 seconds

#include<stdio.h>

int main()

{

    int cs;

    scanf("%d", &cs);

    for(int ii =1; ii <= cs; ii++)

    {

        int h, d, e[10], ttt, tt, t[10];

        scanf("%d %d", &h, &d);

        for(int i =0; i<5; i++)

        {

            scanf("%d %d %d", &ttt, &tt, &e[i]);

            t[i] = ttt\* 60 + tt;

        }

        int mn = 99999999;

        for(int i = 0; i <= d; i++)

        {

            for(int j =0; j <= d - i; j++)

            {

                for(int k = 0; k <= d - i - j; k++)

                {

                    for(int l = 0; l <= d - i - j - k; l++)

                    {

                        for(int m = 0; m <= d - i - j - k - l; m++)

                        {

                            if(i + j + k + l + m == d)

                            {

                                if(e[0]\*i + e[1]\*j + e[2]\*k + e[3]\*l + e[4]\*m <= h)

                                {

                                    if(t[0]\*i + t[1]\*j + t[2]\*k + t[3]\*l + t[4]\*m < mn)

                                        mn = t[0]\*i + t[1]\*j + t[2]\*k + t[3]\*l + t[4]\*m;

                                }

                            }

                        }

                    }

                }

            }

        }

        printf("#%d %d %d\\n", ii, mn/60, mn%60);

    }

}

# Probability

|  |  |
| --- | --- |
| INPUT | OUTPUT |
|  |  |

#include <stdio.h>

#define sizen 102

int n, m, time, node;

int T, Case;

double Ans;

double pro[sizen][sizen];

double dp[sizen][sizen];

void readcase()

{

    int i, j, x, y;

    double p;

    scanf("%d %d %d", &n, &m, &time);

    for (i = 1; i <= n; i++)

        for (j = 1; j <= n; j++)

            pro[i][j] = 0.0;

    for (i = 0; i < m; i++)

    {

        scanf("%d %d %lf", &x, &y, &p);

        pro[x][y] = p;

    }

}

void solvecase()

{

    int i, j, k;

    Ans = 0;

    dp[0][1] = 1.0;

    for (j = 2; j <= n; j++)

        dp[0][j] = 0.0;

    for (i = 0; i < time / 10; i++)

    {

        for (j = 1; j <= n; j++)

            dp[i + 1][j] = 0.0;

        for (j = 1; j <= n; j++)

            if (dp[i][j] > 0)

                for (k = 1; k <= n; k++)

                    if (pro[j][k])

                        dp[i + 1][k] += dp[i][j] \* pro[j][k];

    }

    for (j = 1; j <= n; j++)

        if (dp[i][j] > Ans)

        {

            Ans = dp[i][j];

            node = j;

        }

}

void printcase()

{

    printf("#%d %d %lf\n", Case, node, Ans);

}

int main()

{

    freopen("probability\_input.txt", "r", stdin);

    freopen("test.txt", "w", stdout);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

# Research

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 4    4 3 4  1 0 1  1 1 1  1 1 0  1 1 0  3 0  2 1  0 2  0 0    4 2 4  1 1  0 1  1 1  1 1  3 0  0 1  2 1  0 0    6 3 4  0 0 1  0 0 0  0 0 1  0 1 1  1 0 1  1 1 0  2 2  3 1  4 2  3 2    5 4 2  1 1 0 0  0 0 1 0  0 1 0 0  0 0 1 1  1 0 1 1  0 0  4 2 | #1 4  #2 2  #3 3  #4 -1 |

#include <stdio.h>

#define MAX 202

#define inf (10001 \* 10001)

int T, Case, Ans;

int a[MAX][MAX];

int x[MAX], y[MAX];

int dp[MAX][MAX][5];

int n, m, c;

int max, min;

void readcase()

{

    int i, j, k;

    scanf("%d %d %d", &n, &m, &c);

    for (i = 0; i < n; i++)

        for (j = 0; j < m; j++)

        {

            scanf("%d", &a[i][j]);

            for (k = 0; k < c; k++)

                dp[i][j][k] = inf;

        }

    for (i = 0; i < c; i++)

        scanf("%d %d", &x[i], &y[i]);

}

void solve(int i, int j, int k, int cost)

{

    if (cost < dp[i][j][k])

        dp[i][j][k] = cost;

    else

        return;

    if (a[i][j])

    {

        if (i - 1 >= 0)

            solve(i - 1, j, k, cost + 1);

        if (i + 1 < n)

            solve(i + 1, j, k, cost + 1);

        if (j - 1 >= 0)

            solve(i, j - 1, k, cost + 1);

        if (j + 1 < m)

            solve(i, j + 1, k, cost + 1);

    }

}

void solvecase()

{

    int i, j, k;

    for (k = 0; k < c; k++)

        solve(x[k], y[k], k, 0);

    Ans = inf;

    for (i = 0; i < n; i++)

        for (j = 0; j < m; j++)

            if (!a[i][j])

            {

                max = dp[i][j][0];

                for (k = 0; k < c; k++)

                    if (max < dp[i][j][k])

                        max = dp[i][j][k];

                if (max < Ans)

                    Ans = max;

            }

}

void printcase()

{

    printf("#%d %d\n", Case, Ans == inf ? -1 : Ans);

}

int main()

{

    freopen("research\_input.txt", "r", stdin);

    freopen("test.txt", "w", stdout);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

# River

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 5  2 1  7 2  16 6  6 10  8 8  16 1  5 8  10 4  12 1  3 3  11 5  16 7  5 9  12 4  18 4 | #1 3  #2 10  #3 4  #4 3  #5 4 |

#include <stdio.h>

#define TOTAL 30

int T, Case, Ans;

int g1, g2, g3, f1, f2, f3;

int Abs(int a, int b)

{

    return a > b ? a - b : b - a;

}

int Max(int a, int b)

{

    return a > b ? a : b;

}

int Min(int a, int b)

{

    return a < b ? a : b;

}

void readcase()

{

    scanf("%d %d %d %d %d %d", &g1, &f1, &g2, &f2, &g3, &f3);

}

void solvecase()

{

    int i, j, k;

    int left, right;

    int dist1, dist2, dist3, max;

    Ans = TOTAL;

    for (i = 1; i + f1 + f2 + f3 - 1 <= TOTAL; i++)

    {

        left = Abs(i, g1);

        right = Abs(g1, i + f1 - 1);

        dist1 = Max(left, right);

        for (j = i + f1; j + f2 + f3 - 1 <= TOTAL; j++)

        {

            left = Abs(j, g2);

            right = Abs(g2, j + f2 - 1);

            dist2 = Max(left, right);

            for (k = j + f2; k + f3 - 1 <= TOTAL; k++)

            {

                left = Abs(k, g3);

                right = Abs(g3, k + f3 - 1);

                dist3 = Max(left, right);

                max = Max(dist1, dist2);

                max = Max(max, dist3);

                Ans = Min(max, Ans);

            }

        }

    }

}

void printcase()

{

    printf("#%d %d\n", Case, Ans);

}

int main()

{

    freopen("river\_input.txt", "r", stdin);

    freopen("test.txt", "w", stdout);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

# Rock

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 4  5  1 0 0 1 0  1 1 1 1 1  1 0 0 1 0  1 3 0 0 0  1 0 0 0 0    6  1 1 1 0 1 3  1 0 1 1 0 1  0 0 1 0 1 0  0 1 0 1 0 0  1 0 0 0 0 0  1 0 1 0 1 0    7  1 0 0 1 1 0 0  1 0 1 1 1 1 0  1 1 1 0 0 0 0  1 0 0 1 0 0 0  0 1 1 1 0 0 1  0 0 1 0 3 0 0  1 1 1 0 0 1 1    8  0 0 1 0 0 1 1 3  1 0 0 0 0 0 0 1  1 1 0 1 0 1 1 1  1 0 0 0 0 1 0 1  1 0 0 1 1 1 1 0  1 1 1 1 1 1 1 1  0 0 0 0 1 0 0 0  1 0 1 0 0 0 1 1 | #1 0  #2 1  #3 1  #4 1 |

#include <stdio.h>

#define SIZE 101

#define inf (SIZE \* SIZE)

int T, Case, Ans;

int N, X, Y;

int grid[SIZE][SIZE], dp[SIZE][SIZE];

void readcase()

{

    int i, j;

    scanf("%d", &N);

    for (i = 0; i < N; i++)

        for (j = 0; j < N; j++)

        {

            scanf("%d", &grid[i][j]);

            dp[i][j] = inf;

            if (3 == grid[i][j])

            {

                X = i;

                Y = j;

            }

        }

}

void solve(int x, int y, int maxzero, int curzero)

{

    if (0 == grid[x][y])

        curzero++;

    else

        curzero = 0;

    if (maxzero < curzero)

        maxzero = curzero;

    if (maxzero < dp[x][y])

        dp[x][y] = maxzero;

    else

        return;

    if (x - 1 >= 0)

        solve(x - 1, y, maxzero, curzero);

    if (x + 1 < N)

        solve(x + 1, y, maxzero, curzero);

    if (y - 1 >= 0 && grid[x][y - 1])

        solve(x, y - 1, maxzero, curzero);

    if (y + 1 < N && grid[x][y + 1])

        solve(x, y + 1, maxzero, curzero);

}

void solvecase()

{

    solve(N - 1, 0, 0, 0);

    if (inf == dp[X][Y])

        dp[X][Y] = -1;

    Ans = dp[X][Y];

}

void printcase()

{

    printf("#%d %d\n", Case, Ans);

}

int main()

{

    freopen("rock\_input.txt", "r", stdin);

    freopen("test2.txt", "w", stdout);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

# Spaceship

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 5  6  1 1 2 1 2  2 0 2 1 0  2 1 1 1 0  1 0 0 1 2  2 1 1 0 0  0 2 2 2 1  7  0 0 2 0 2  1 2 2 1 1  1 2 2 2 1  0 2 2 2 2  1 0 2 1 0  1 0 1 1 1  1 2 0 0 2  8  2 0 2 2 0  1 1 2 2 2  0 2 1 1 0  1 2 2 1 2  0 1 1 1 0  1 1 1 2 0  1 1 2 0 2  1 2 1 2 0  9  0 0 2 2 0  2 2 1 2 0  0 2 2 0 2  2 2 2 1 2  1 2 0 0 0  2 2 0 1 1  0 0 0 0 2  0 2 2 2 2  0 2 2 0 0  9  0 2 0 2 1  2 1 2 1 1  2 0 0 2 1  1 0 1 0 1  2 0 2 2 0  1 0 0 2 2  0 1 0 0 0  1 2 2 2 2  0 2 2 0 2 | #1 5  #2 4  #3 7  #4 3  #5 6 |

/\*

    mistook col+1 with col+5

\*/

#include <stdio.h>

int T, Case, max, N;

int board[101][5];

void readcase()

{

    int i, j;

    scanf("%d", &N);

    for (i = 0; i < N; i++)

        for (j = 0; j < 5; j++)

            scanf("%d", &board[i][j]);

}

void solve(int row, int col, int life, int score)

{

    if (row < 0)

    {

        if (max < score)

            max = score;

        return;

    }

    if (life > 0 && life < 6)

        life--;

    if (1 == board[row][col])

        score++;

    else if (2 == board[row][col])

    {

        if (0 == life)

        {

            if (max < score)

                max = score;

            return;

        }

        else if (6 == life)

            life--;

    }

    if (col - 1 >= 0)

        solve(row - 1, col - 1, life, score);

    solve(row - 1, col, life, score);

    if (col + 1 < 5)

        solve(row - 1, col + 1, life, score);

}

void solvecase()

{

    max = 0;

    board[N][2] = 0;

    solve(N, 2, 6, 0);

}

void printcase()

{

    printf("#%d %d\n", Case, max);

}

int main()

{

    freopen("spaceship\_input.txt", "r", stdin);

    freopen("test.txt", "w", stdout);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}

# Travelling Salesman Problem (TSP)

|  |  |
| --- | --- |
| INPUT | OUTPUT |
| 3  5  0 0 100 100 70 40 30 10 10 5 90 70 50 20  6  88 81 85 80 19 22 31 15 27 29 30 10 20 26 5 14  10  39 9 97 61 35 93 62 64 96 39 36 36 9 59 59 96 61 7 64 43 43 58 1 36 | #1 200  #2 304  #3 366 |

#include <stdio.h>

#define MAX 101

#define inf (MAX\*MAX)

int x[MAX], y[MAX], TSP[MAX], vis[MAX];

int N, T, Case, min;

int diff(int i, int j)

{

    int dist = x[i] > x[j] ? x[i] - x[j] : x[j] - x[i];

    return dist += y[i] > y[j] ? y[i] - y[j] : y[j] - y[i];

}

void readcase()

{

    int i;

    scanf("%d", &N);

    scanf("%d %d %d %d", &x[0], &y[0], &x[N + 1], &y[N + 1]);

    vis[0] = 0;

    for (i = 1; i <= N; i++)

    {

        scanf("%d %d", &x[i], &y[i]);

        vis[i] = 0;

    }

}

void solve(int i, int dist)

{

    int j;

    if (min <= dist)

        return;

    if (i == N + 1)

    {

        dist += diff(i, TSP[i - 1]);

        if (min > dist)

        {

            min = dist;

            return;

        }

    }

    for (j = 1; j <= N; j++)

    {

        if (!vis[j])

        {

            vis[j] = 1;

            TSP[i] = j;

            solve(i + 1, dist + diff(TSP[i], TSP[i - 1]));

            vis[j] = 0;

        }

    }

}

void solvecase()

{

    min = inf;

    TSP[0] = 0;

    solve(1, 0);

}

void printcase()

{

    printf("#%d %d\n", Case, min);

}

int main()

{

    freopen("tsp\_input.txt", "r", stdin);

    scanf("%d", &T);

    for (Case = 1; Case <= T; Case++)

    {

        readcase();

        solvecase();

        printcase();

    }

    return 0;

}