answers08

1.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main()
{
    int N, T = 10000;
    int array[100][100];
    int offset[4][2] = { {0, -1}, {1, 0}, {0, 1}, {-1, 0} };
    int succCount = 0;

    srand((unsigned int)time(NULL));
    scanf("%d", &N);

    for (int t = 0; t < T; t++) {
        /* initialize x, y, and the array for each experiment */
        int x = N/2, y = N/2;
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < N; j++)
                array[i][j] = 0;
        }
        bool success = true;
        while (x > 0 && x < N-1 && y > 0 && y < N-1 ) {
            int direction = -1;
            int nAvailableDirs = 0;
            array[x][y] = 1;
            for (int d = 0; d < 4; d++) {
                int newx = x + offset[d][0];
                int newy = y + offset[d][1];
                if (newx>=0&&newx<N&&newy>=0&&newy<N&&array[newx][newy]==0) {
                    nAvailableDirs++;
                    if (rand() % nAvailableDirs == 0)
                        direction = d;
                }
            }
            if (direction == -1) {
                success = false;
                break;
            }
            x += offset[direction][0];
            y += offset[direction][1];
        }
        if (success) succCount++;
    }
    printf("%lf\n", (double)succCount/T);
}
```

2.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main()
{
    int N, T = 10000;
    int array[100][100][100];
    int offset[6][3] = {{0, -1, 0}, {1, 0, 0}, {0, 1, 0},
                           {-1, 0, 0},{0, 0, 1},{0, 0, -1} };
    int succCount = 0;
    srand((unsigned int)time(NULL));
    scanf("%d", &N);

    for (int t = 0; t < T; t++) {
        /* initialize x, y, z, and the array for each experiment */
        int x = N/2, y = N/2, z = N/2;
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < N; j++)
                for (int k = 0; k < N; k++)
                    array[i][j][k] = 0;
        }
        bool success = true;
        while (x > 0 && x < N-1 && y > 0 && y < N-1 && z > 0 && z < N-1) {
            int direction = -1;
            int nAvailableDirs = 0;
            array[x][y][z] = 1;
            for (int d = 0; d < 6; d++) {
                int newx = x + offset[d][0];
                int newy = y + offset[d][1];
                int newz = z + offset[d][2];

                if (newx>=0 && newx<N && newy>=0 && newy<N && newz>=0
                            && newz < N && array[newx][newy][newz] == 0) {
                    nAvailableDirs++;
                    if (rand() % nAvailableDirs == 0)
                        direction = d;
                }
            }
            if (direction == -1) {
                success = false;
                break;
            }
            x += offset[direction][0];
            y += offset[direction][1];
            z += offset[direction][2];
        }
        if (success) succCount++;
    }
    printf("%lf\n", (double)succCount/T);
}
```

3.

```
#include <stdio.h>
int main()
{
    int n, k;
    int table[101][101] = { 0 };
    scanf("%d %d", &n, &k);

    for (int i = 0; i <= n; i++) {
        for (int j = 0; j <= i; j++ ) {
            if ( j == 0 || i == j )
                table[i][j] = 1;
            else
                table[i][j] = table[i-1][j-1] + table[i-1][j];
        }
    }
    printf("%d\n", table[n][k]);
}
```

4.

```
#include <stdio.h>
#include <math.h>
int main()
{
    int N;
    int mat[100][100];
    int colSum[100] = { 0 }, rowSum[100] = { 0 };
    double colVar[100] = { 0.0 }, rowVar[100] = { 0.0 };

    FILE *fp = fopen("input4.txt", "r");
    fscanf(fp, "%d", &N);
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++ ) {
            fscanf(fp, "%d", &mat[i][j]);
        }
    }
    fclose(fp);

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++ ) {
            colSum[j] += mat[i][j];
            rowSum[i] += mat[i][j];
        }
    }

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++ ) {
            double colDiff = mat[i][j] - (double)colSum[j]/N;
            double rowDiff = mat[i][j] - (double)rowSum[i]/N;
            colVar[j] += (colDiff*colDiff);
            rowVar[i] += (rowDiff*rowDiff);
        }
    }

    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++ ) {
            printf("%d ", mat[i][j]);
        }
        printf("%lf %lf\n", (double)rowSum[i]/N, sqrt(rowVar[i]/N));
    }
    for (int j = 0; j < N; j++ )
        printf("%lf ", (double)colSum[j]/N);
    printf("\n");
    for (int j = 0; j < N; j++ )
        printf("%lf ", sqrt(colVar[j]/N));
    printf("\n");
}
```

5.

```
#include <stdio.h>
int main()
{
    int N;
    int mat[100][100];

    FILE *fp = fopen("input5.txt", "r");
    fscanf(fp, "%d", &N);
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++ ) {
            fscanf(fp, "%d", &mat[i][j]);
        }
    }
    fclose(fp);

    int maxSum = mat[0][0];
    for (int sx = 0; sx < N; sx++) {
        for (int ex = sx; ex < N; ex++) {
            for (int sy = 0; sy < N; sy++) {
                for (int ey = sy; ey < N; ey++) {
                    int sum = 0;
                    for (int i = sx; i <= ex; i++) {
                        for (int j = sy; j <= ey; j++)
                            sum += mat[i][j];
                    }
                    if (sum > maxSum) maxSum = sum;
                }
            }
        }
    }
    printf("%d\n", maxSum);
}

6.
#include <stdio.h>
int main()
{
    int N;
    int mat[100][100];

    FILE *fp = fopen("input6.txt", "r");
    fscanf(fp, "%d", &N);
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++ ) {
            fscanf(fp, "%d", &mat[i][j]);
        }
    }
    fclose(fp);

    for (int i = 0; i < N; i++) {
        int j = ( i%2 == 0 ? 0 : N-1);
        while (j >= 0 && j < N ) {
            printf("%d ", mat[i][j]);
            if ( i%2 == 0 ) j++;
            else j--;
        }
    }
}
```