

(Downloaded from <https://cs.stanford.edu/~knuth/programs.html> and typeset on September 17, 2017)

1. Intro. This program simply generates data for GDANCE to solve the problem of placing MacMahon's 24 triangles into a regular hexagon, as discussed in Martin Gardner in Chapter 16 of *Mathematical Magic Show*. But instead of making the outside edge a solid color, I'm trying for a solution with 180° symmetry when we map the colors $a \leftrightarrow d$, $b \leftrightarrow c$.

And it should tile the plane too.

```
char piece[24][4] = {{ 'a', 'a', 'a', 0}, { 'd', 'd', 'd', 0}, { 'b', 'b', 'b', 0}, { 'c', 'c', 'c', 0}, { 'a',
    'a', 'd', 0}, { 'd', 'd', 'a', 0}, { 'a', 'a', 'b', 0}, { 'd', 'd', 'c', 0}, { 'a', 'a', 'c', 0}, { 'd', 'd',
    'b', 0}, { 'b', 'b', 'a', 0}, { 'c', 'c', 'd', 0}, { 'b', 'b', 'c', 0}, { 'c', 'c', 'b', 0}, { 'b', 'b', 'd',
    0}, { 'c', 'c', 'a', 0}, { 'a', 'b', 'c', 0}, { 'b', 'd', 'c', 0}, { 'a', 'b', 'd', 0}, { 'a', 'd', 'c', 0},
    { 'a', 'c', 'd', 0}, { 'a', 'd', 'b', 0}, { 'a', 'c', 'b', 0}, { 'b', 'c', 'd', 0}, };
char pos[36][4] = {{ 2, 3, 8, 0}, { 3, 0, 4, 0}, { 4, 5, 9, 0}, { 5, 1, 6, 0}, { 6, 7, 10, 0}, { 11, 12, 19, 0}, { 12, 8, 13, 0},
    { 13, 14, 20, 0}, { 14, 9, 15, 0}, { 15, 16, 21, 0}, { 16, 10, 17, 0}, { 17, 18, 22, 0}, };
char map[23] = { 0, 0, 11, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 7, 19, 20, 20, 19 };
main()
{
    register int j, k;
    for (k = 12; k < 36; k++)
        pos[k][0] = pos[k - 12][1], pos[k][1] = pos[k - 12][2], pos[k][2] = pos[k - 12][0];
    < Output the first line of dance data 3 >;
    for (j = 0; j < 24; j++)
        for (k = 0; k < (j >= 4 ? 36 : 12); k++) < Generate rows for piece j in position k 2 >;
}
```

```
2. < Generate rows for piece j in position k 2 > ≡
{
    printf("%sP%02d", piece[j], k % 12);
    if (map[pos[k][0]] == pos[k][0]) printf("□%02d:%c", pos[k][0], piece[j][0]);
    else printf("□%02d:%c", map[pos[k][0]], 'a' + 'd' - piece[j][0]);
    if (map[pos[k][1]] == pos[k][1]) printf("□%02d:%c", pos[k][1], piece[j][1]);
    else printf("□%02d:%c", map[pos[k][1]], 'a' + 'd' - piece[j][1]);
    if (map[pos[k][2]] == pos[k][2]) printf("□%02d:%c", pos[k][2], piece[j][2]);
    else printf("□%02d:%c", map[pos[k][2]], 'a' + 'd' - piece[j][2]);
    printf("□%s\n", piece[j ⊕ 1]);
}
```

This code is used in section 1.

```
3. < Output the first line of dance data 3 > ≡
for (j = 0; j < 24; j++) printf("%s□", piece[j]);
for (k = 0; k < 12; k++) printf("P%02d□", k);
printf("|");
for (k = 0; k < 21; k++)
    if (k ≠ 1 ∧ k ≠ 2 ∧ k ≠ 18) printf("□%02d", k);
printf("\n");
```

This code is used in section 1.

4. Index.*j*: [1](#).*k*: [1](#).*main*: [1](#).*map*: [1](#), [2](#).*piece*: [1](#), [2](#), [3](#).*pos*: [1](#), [2](#).*printf*: [2](#), [3](#).

⟨ Generate rows for piece j in position k 2 ⟩ Used in section 1.
⟨ Output the first line of dance data 3 ⟩ Used in section 1.

MACMAHON-TRIANGLES-SYM-TILE

	Section	Page
Intro	1	1
Index	4	2