

(Downloaded from <https://cs.stanford.edu/~knuth/programs.html> and typeset on May 28, 2023)

**1. Intro.** This program produces a list of all optimal moves in tictactoe.

```
#define rank z.I      /* number of moves made */
#define link y.V      /* next vertex of same rank */
#define head x.V      /* first vertex of given rank */
#define winner w.I    /* is this a winning position? */
#define score w.I     /* minimax value of position */
#define bitcode v.I   /* binary representation of this position */

#include "gb_graph.h"
#include "gb_save.h"
int pref[] = {5, 1, 3, 9, 7, 2, 6, 8, 4}; /* preference order for moves */
int y[10], count[10];

main()
{
    register int j, k, l, q, qq, s;
    register Graph*g = restore_graph("/tmp/tictactoe.gb");
    register Vertex*u, *v;
    register Arc*a;
    < Compute and print the optimum moves 2 >;
    for (k = 1; k ≤ 9; k++) printf("can play %d from %d positions\n", k, count[k]);
}
```

**2.** The *score* takes over from the *winner* field in the input graph.

< Compute and print the optimum moves 2 > ≡

```
for (l = 9; l ≥ 0; l--)
    for (v = (g-vertices + l)-head; v; v = v-link) {
        if (v-winner) v-score = -1;
        else if (v-rank < 9) {
            for (s = 99, a = v-arcs; a; a = a-next) {
                u = a-tip;
                if (s > u-score) s = u-score;
            }
            v-score = -s;
            for (q = 0, a = v-arcs; a; a = a-next) {
                u = a-tip;
                if (s ≡ u-score) q |= u-bitcode;
            }
            < Print the results for position v 3 >;
        }
    }
```

This code is used in section 1.

**3.** < Print the results for position v 3 > ≡

```
for (j = 8, k = v-bitcode; j ≥ 0; j--, k ≫= 2, q ≫= 2) {
    if ((k & 3) ≡ (q & 3)) y[pref[j]] = 0;
    else y[pref[j]] = 1, count[pref[j]]++;
}
printf("%05x: %d%d%d%d%d%d%d%d\n", v-bitcode, y[1], y[2], y[3], y[4], y[5], y[6], y[7], y[8], y[9]);
```

This code is used in section 2.

**4. Index.***a*: [1](#).*Arc*: [1](#).*arcs*: [2](#).*bitcode*: [1](#), [2](#), [3](#).*count*: [1](#), [3](#).*g*: [1](#).*Graph*: [1](#).*head*: [1](#), [2](#).*j*: [1](#).*k*: [1](#).*l*: [1](#).*link*: [1](#), [2](#).*main*: [1](#).*next*: [2](#).*pref*: [1](#), [3](#).*printf*: [1](#), [3](#).*q*: [1](#).*qq*: [1](#).*rank*: [1](#), [2](#).*restore\_graph*: [1](#).*s*: [1](#).*score*: [1](#), [2](#).*tip*: [2](#).*u*: [1](#).*v*: [1](#).*Vertex*: [1](#).*vertices*: [2](#).*winner*: [1](#), [2](#).*y*: [1](#).

⟨ Compute and print the optimum moves 2 ⟩ Used in section 1.  
⟨ Print the results for position  $v$  3 ⟩ Used in section 2.

# TICTACTOE4

	Section	Page
Intro .....	<a href="#">1</a>	1
Index .....	<a href="#">4</a>	2