1. Intro. This program creates the McGregor graph of order n, given n on the command line.

Martin Gardner's famous April Fool column of 1975 included six "major discoveries of 1974 that for one reason or another were inadequately reported to both the scientific community and to the public at large."

One of these supposed breakthroughs was the planar graph found by William McGregor, which (Martin said) cannot be colored with four colors. (See his *Time Travel and Other Mathematical Bewilderments* (1988), page 127.)

This graph can obviously be generalized to other sizes, so I've decided to call it the McGregor graph of order 10. In general, the McGregor graph of order n has $n(n+1)$ vertices and $3n(n+1)-6$ edges; its edges are implicit in the diagram above and made formally explicit in the algorithm below.

(Note added 27 Nov 2012: I found McGregor's original correspondence among Gardner's papers in the Stanford archives. He explicitly considered the published graph to be number 10 in an infinite series beginning with $n = 3$.)

```
#include "gb_graph.h"    /* we use the GB_GRAPH data structures */
#include "gb_save.h"    /* and we save our results in ASCII format */
  int n;    /* the order */
  char buf[16];

  int main(int argc, char *argv[])
  {
    register int i, j, k;
    register Graph *g;

    ⟨Read the command line to determine n 2⟩;
    ⟨Create an empty graph of n(n + 1) vertices, and name them 3⟩;
    for (j = 0; j ≤ n; j++)
      for (k = 0; k < n; k++) ⟨Generate edges that involve vertex (j, k) 4⟩;
    sprintf(g→id, "mcgregor(%d)", n);
    sprintf(buf, "mcgregor%d.gb", n);
    save_graph(g, buf);
  }
```

2. ⟨Read the command line to determine n 2⟩ ≡

```
  if (argc ≠ 2 ∨ sscanf(argv[1], "%d", &n) ≠ 1) {
    fprintf(stderr, "Usage:␣%s␣n\n", argv[0]);
    exit(−1);
  }
```

This code is used in section 1.

3.

```
#define vert(j, k)  (g→vertices + (n * (j) + (k)))
#define edge(j, k, jj, kk)  gb_new_edge(vert(j, k), vert(jj, kk), 1)
```

⟨Create an empty graph of $n(n+1)$ vertices, and name them 3⟩ ≡

```
  g = gb_new_graph(n * (n + 1));
  if (¬g) {
    fprintf(stderr, "Can't␣create␣an␣empty␣graph␣of␣%d␣vertices!\n", n * (n + 1));
    exit(−2);
  }
  for (j = 0; j ≤ n; j++)
    for (k = 0; k < n; k++) {
      sprintf(buf, "%d.%d", j, k);
      vert(j, k)→name = gb_save_string(buf);
    }
```

This code is used in section 1.

4. I suspect that we'd get a graph with fewer 4-colorings if the threshold $\lfloor n/2 \rfloor$ were changed to $\lfloor (n-1)/2 \rfloor$ in the code below for $(j,k) = (0,0)$ and $(j,k) = (1,0)$. (Then McGregor's original graph would have had $(1,0)$ adjacent to $(10,4)$, which is labeled 'a4' in the diagram, but $(0,0)$ wouldn't have been adjacent to $(10,5)$.) I hope to test this conjecture soon. It matters only if n is even.

⟨ Generate edges that involve vertex (j,k) 4 ⟩ ≡

```
{
  if (j ≡ 0 ∧ k ≡ 0) {
    edge(j, k, 1, 0);
    for (i = 1; i ≤ n ≫ 1; i++) edge(j, k, n, i);
  }
  if (j ≡ 1 ∧ k ≡ 0) {
    for (i = n ≫ 1; i < n; i++) edge(j, k, n, i);
    continue;    /* no other edges from (1, 0) */
  }
  if (j < n ∧ k < n − 1) edge(j, k, j + 1, k + 1);
  if (j ≡ 0) edge(j, k, n, n − 1);
  if (j ≠ k ∧ j < n) edge(j, k, j + 1, k);
  if (j ≠ k + 1 ∧ k < n − 1) edge(j, k, j, k + 1);
  if (j ≡ k ∧ k < n − 1) edge(j, k, n − j, 0);
  if (j ≡ k ∧ j > 0) edge(j, k, n + 1 − j, 0);
  if (k ≡ n − 1 ∧ j > 0 ∧ j < k) edge(j, k, n − j, n − j − 1);
  if (k ≡ n − 1 ∧ j > 0 ∧ j < n) edge(j, k, n + 1 − j, n − j);
}
```

This code is used in section 1.

5. Index.

$\langle$ Create an empty graph of $n(n+1)$ vertices, and name them 3 $\rangle$ Used in section 1.
$\langle$ Generate edges that involve vertex (j,k) 4 $\rangle$ Used in section 1.
$\langle$ Read the command line to determine n 2 $\rangle$ Used in section 1.

MCGREGOR-GRAPH