1. Intro. Given a graph g with m edges, make data from which DLX2 should tell us all ways to label the vertices, using distinct labels in $\{0, 1, \ldots, m\}$, so that the edges have distinct difference. (Those differences will be $\{1, \ldots, m\}$.

Selected vertex labels may be prespecified on the command line, as in BACK-GRACEFUL.

```
#define encode(x) ((x) < 10 ? (x) + '0' : (x) < 36 ? (x) - 10 + 'a' : (x) < 62 ? (x) - 36 + 'A' : (x) + 99)
#define maxm 156 /* based on that encoding, but I could go higher in a pinch! */
#define maxn 100
#include <stdio.h>
#include <stdlib.h>
#include "gb_graph.h"
#include "gb_save.h"
  int c;
  int label; /* a label value read from argv[k] */
  int prespec[maxn]; /* prespecified labels */
  int verttoprespec[maxn]; /* has this vertex been prespecified? */
  int prespecptr; /* how many are prespecified? */
  main(int argc, char *argv[])
  {
    register int i, j, k, m, n, p, x, bad;
    register Arc *a;
    register Graph *g;
    register Vertex *v, *w;
    ⟨ Process the command line, and set prespec to the prespecified labelings 2 ⟩;
    ⟨ Output the item-name line 3 ⟩;
    for (k = 1; k ≤ m; k++) ⟨ Output the options for edge k 4 ⟩;
  }
```

2. ⟨ Process the command line, and set *prespec* to the prespecified labelings 2 ⟩ ≡

```
if (argc < 2) {
  fprintf(stderr, "Usage: %s foo.gb [VERTEX=label...]\n", argv[0]);
  exit(-1);
}
g = restore_graph(argv[1]);
if (¬g) {
  fprintf(stderr, "I couldn't reconstruct graph %s!\n", argv[1]);
  exit(-2);
}
m = g→m/2, n = g→n;
if (m > maxm) {
  fprintf(stderr, "Sorry, at present I require m<=%d!\n", maxm);
  exit(-3);
}
if (n > maxn) {
  fprintf(stderr, "Sorry, at present I require n<=%d!\n", maxn);
  exit(-4);
}
for (k = 2; argv[k]; k++) {
  for (i = 1; argv[k][i]; i++)
    if (argv[k][i] ≡ '=') break;
  if (¬argv[k][i] ∨ sscanf(&argv[k][i + 1], "%d", &label) ≠ 1 ∨ label < 0 ∨ label > m) {
    fprintf(stderr, "spec `%s' doesn't have the form `VERTEX=label'!\n", argv[k]);
    exit(-3);
  }
  argv[k][i] = 0;
  for (j = 0; j < n; j++)
    if (strcmp((g→vertices + j)→name, argv[k]) ≡ 0) break;
  if (j ≡ n) {
    fprintf(stderr, "There's no vertex named `%s'!\n", argv[k]);
    exit(-5);
  }
  if (verttoprespec[j]) {
    fprintf(stderr, "Vertex %s was already specified!\n", (g→vertices + j)→name);
    exit(-6);
  }
  argv[k][i] = '=';
  verttoprespec[j] = 1;
  prespec[prespecptr++] = (j ≪ 8) + label;
}
fprintf(stderr, "OK, I've got a graph with %d vertices, %d edges, %d prespec%s.\n", n, m,
    prespecptr, prespecptr ≡ 1 ? "" : "s");
printf("|");
for (k = 0; argv[k]; k++) printf(" %s", argv[k]);
printf("\n");
```

This code is used in section 1.

3. There's a primary item k for each edge label, and a primary item uv for each edge. This enforces a permutation between edges and labels.

There's a secondary item `.`v for each vertex; its color will be its label.

There's a secondary item `+`k for each vertex label; its color will be the vertex so labeled.

```
⟨ Output the item-name line 3 ⟩ ≡
  for (k = 1; k ≤ m; k++) printf("%c␣", encode(k));
  for (v = g→vertices; v < g→vertices + n; v++)
    for (a = v→arcs; a; a = a→next)
      if (a→tip > v) printf("%s-%s␣", v→name, a→tip→name);
  printf("|");
  for (v = g→vertices; v < g→vertices + n; v++) printf("␣.%s", v→name);
  for (k = 0; k ≤ m; k++) printf("␣+%c", encode(k));
  printf("\n");
```

This code is used in section 1.

4. `#define` $vrt(v)$ $((\textbf{int})((v) - g\text{→}vertices))$

```
⟨ Output the options for edge k 4 ⟩ ≡
  {
    for (i = 0, j = k; j ≤ m; i++, j++) {
      for (v = g→vertices; v < g→vertices + n; v++)
        for (a = v→arcs; a; a = a→next)
          if (a→tip > v) {
            for (bad = p = 0; p < prespecptr; p++) {
              w = g→vertices + (prespec[p] ≫ 8), x = prespec[p] & #ff;
              if (v ≡ w) {
                if (i ≠ x) bad |= 1;
                if (j ≠ x) bad |= 2;
              } else if (a→tip ≡ w) {
                if (j ≠ x) bad |= 1;
                if (i ≠ x) bad |= 2;
              }
              if (i ≡ x) {
                if (v ≠ w) bad |= 1;
                if (a→tip ≠ w) bad |= 2;
              } else if (j ≡ x) {
                if (v ≠ w) bad |= 2;
                if (a→tip ≠ w) bad |= 1;
              }
            }
            if ((bad & 1) ≡ 0) printf("%c␣%s-%s␣.%s:%c␣.%s:%c␣+%c:%c␣+%c:%c\n", encode(k),
                v→name, a→tip→name, v→name, encode(i), a→tip→name, encode(j), encode(i),
                encode(vrt(v)), encode(j), encode(vrt(a→tip)));
            if ((bad & 2) ≡ 0) printf("%c␣%s-%s␣.%s:%c␣.%s:%c␣+%c:%c␣+%c:%c\n", encode(k),
                v→name, a→tip→name, v→name, encode(j), a→tip→name, encode(i), encode(j),
                encode(vrt(v)), encode(i), encode(vrt(a→tip)));
          }
    }
  }
```

This code is used in section 1.

5. Index.

⟨ Output the item-name line 3 ⟩ Used in section 1.
⟨ Output the options for edge k 4 ⟩ Used in section 1.
⟨ Process the command line, and set *prespec* to the prespecified labelings 2 ⟩ Used in section 1.

GRACEFUL-DLX-PRESETS