```c
#include <stdio.h>
#include <glib.h>

#define ATOM_STATES 3

typedef struct _AtomList {
    gint8 state;
    gdouble x;
    gdouble y;
} AtomList;

typedef struct _Atoms {
    gint32 number;
    gint32 states[ATOM_STATES];
    gdouble wide;
    AtomList *list;
} Atoms;

Atoms *create_atoms(gint32 number)
{
    Atoms *atoms;

    atoms = (Atoms *) g_malloc(sizeof(Atoms));
    atoms->list = (AtomList *) g_malloc(number * sizeof(AtomList));

    atoms->number = number;
    atoms->states[0] = number;
    atoms->states[1] = 0;
    atoms->states[2] = 0;

    return atoms;
}

void destroy_atoms(Atoms *atoms)
{
    g_free(atoms->list);
    g_free(atoms);
}
```

```
gint main(gint argc, gchar *argv[])
{
    GRand *rand;
    GTimer *timer;
    gdouble halftime, ctime, waittime;
    gint32 number, i;
    Atoms *atoms;

    ctime = waittime = 0;

    if (argc != 3) {
        printf("Usage: %s NUMBER HALFTIME\n", argv[0]);
        exit(1);
    }

    number = (gint32) g_ascii_strtod(argv[1], NULL);
    halftime = g_ascii_strtod(argv[2], NULL);

    atoms = create_atoms(number);

    rand = g_rand_new();
    timer = g_timer_new();

    while (atoms->states[0] > 0) {
        ctime = g_timer_elapsed(timer, NULL);

        for (i = atoms->states[0]; i > 0; i--) {
            if (g_rand_boolean(rand)) {
                atoms->states[0]--;
                atoms->states[1]++;
            }
        }

        waittime = halftime - (g_timer_elapsed(timer, NULL) - ctime);
        if (waittime > 0)
            g_usleep((gulong) (waittime + 0.5) * G_USEC_PER_SEC);
        else
            printf("WARNING: your computer is to SLOOOOW to compute this!\n");
    }

    g_timer_stop(timer);
    g_timer_destroy(timer);

    g_rand_free(rand);

    destroy_atoms(atoms);
    return 0;
}
```