Space versus Time Tradeoff

The 2 small example programs here illustrate how space (memory) can be used to gain a significant improvement in running time for a problem. The problem is, given an array of numbers, find out if any numbers occurrs more than once.

```
// occurs1.cpp

#include <iostream.h>
#include <stdlib.h>
#include  <math.h>
#define N 5000

void main()
{
    short a[N];
    int i, j;
    int count=0;

    for(i=0; i<N; ++i)
        a[i] = rand();

    // search for duplicates
    for(i=0; i<N-1; ++i)
        for(j=i+1; j<N; ++j)
            if(a[i] == a[j]) {
                cout << a[i] << "  ";
                ++count;
            }

    cout << "\n\nCount= " << count << endl;
}
```

This search loop here involves $N^2/2$ steps.

So time complexity = $O(N^2)$

Space complexity = $O(N)$

```
// occurs2.cpp

#include <iostream.h>
#include <stdlib.h>
#include  <math.h>
#define N 5000
#define M 32767
void main()
{
    short a[N];
    char b[M+1];  // RAND_MAX is 32767
    int i;
    int count=0;

    // setup an array of nos
    for(i=0; i<N; ++i)
        a[i] = rand();

    // initialise count array
    for(i=0; i<=M; ++i)
        b[i] = 0;

    // search for duplicates
    for(i=0; i<N; ++i) {
        ++b[a[i]];
        if (b[a[i]] > 1) {
            cout << a[i] << "  ";
            ++count;
        }
    }
    cout << "\n\nCount= " << count << endl;
}
```

This search loop here involves N steps.

So time complexity = $O(N)$

Space complexity = $O(M)$ where $M = 2^m - 1$ and m is the maximum number of bits that any number in the array can be. m = 15 so M = 32,767 in this example.