Arrays Bigger than 32K

The following example illustrates a C programming technique that is not necessarily Macintosh-specific.

```
/*
    Arrays Bigger than 32K
    This is a very simple program to demonstrate using dynamically allocated arrays
    of larger than 32K in size.

    This example will allocate an integer array larger than 32k, then fill it with
    some sqares of numbers,  then print out the array.

*/

#include <stdio.h>
#include <stdlib.h>

#define DIMENSION_1 10
#define DIMENSION_2 300


int **myarray;


main()
{
    long i,j;

    /*  first allocate all 400 bytes for first dimension of array */
    /*  and of course check for allocation getting done w/o error */
    if ((myarray = (int **)calloc(DIMENSION_1,sizeof(int *))) == NULL)
    {
        printf ("Unable to allocate memory\n");
        exit(1);
    }

    /*  Now allocate each array from withing first dimension */
    for (i= 0; i < DIMENSION_1; i++)
    {
        myarray[i] = (int *)calloc(DIMENSION_2,sizeof(int));
        if (myarray[i] == NULL)
        {
            printf ("Unable to allocate memory for second dimension\n");
            exit(1);
        }
    }

    /* now to see how to use array, fill array with all squares of i */
    for (i = 0; i < DIMENSION_1; i++)
    for (j = 0; j<DIMENSION_2;j++)
        myarray[i][j] = i*i;


    /* now print out first info to see that it worked*/
```

```
    for (i = 0; i < DIMENSION_1; i++)
            for (j=0; j<DIMENSION_2;j++)
                    printf ("%ld,%ld  %ld\n", i,j,myarray[i][j]);

}
```