QSort Examples

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

```
/* QSort Examples
 * This example program demonstrates the use of qsort to sort an array of integers
 */

#include <stdlib.h>
#include <stdio.h>
int compare(const void *item1, const void *item2);

/* create the data to be sorted */

int x[20] =
     {20,19,18,18,545,4534,456,456,23,432,435,7,3,4,32,34,2,30,78,5};

/* declare your compare function */
int compare(const void *item1, const void *item2)
{
     return (*(int *)item1-*(int *)item2);
}

main()
{
     int i;

     printf ("Unsorted Array\n");
     for (i=0;i < 20 ; i++) printf("%d\n",x[i]);
     qsort(x,20,sizeof(int),compare);
     printf("\n");
     printf ("Sorted Array\n");
     for (i=0;i < 20 ; i++) printf("%d\n",x[i]);
}

/*
 * This is an example program that makes use of qsort to sort a dynamically
 * allocated array of character pointers.
*/

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

#define NUM_ARRAY_ELEMS        5        /* number of elements in array */
#define STRING_LENGTH          3        /* maximum length of any string in array
                                         */

int compare(const void *s1, const void *s2);

int compare (const void *s1, const void *s2)
{

     /* note since it is an array of pointers, what you actually have is a pointer
     * to a pointer, i.e. the array subscript and the pointer at that location.
     */

    /* write your compare function to compare 2 strings */
    return (strcmp (*(char **)s1, *(char **)s2));
}

main()
{
    char    **myarray;   /* array to allocate */
    int     i;           /* some index */

    /* allocate enough room to hold 5 pointers in array */
    myarray = (char **)malloc(NUM_ARRAY_ELEMS * sizeof(char *));

    /* handle errors here */

    /* now for each location, allocate char pointer */
    for (i = 0; i < 5; i++)
    {
            myarray[i] = (char *)malloc(STRING_LENGTH);
    /* handle errors here */
    }

    /* now initialize array contents */
    strcpy (myarray[0], "ze");
    strcpy (myarray[1], "y ");
    strcpy (myarray[2], "zz");
    strcpy (myarray[3], "ba");
    strcpy (myarray[4], "a ");
    printf ("array prior to sorting:\n");

    for (i = 0; i < 5; i++)
            printf ("myarray [%d] = %s\n",i,myarray[i]);

    /* now call qsort to sort the pointers, note that you are actually swapping
     * pointer thus the size is sizeof(char *)
     */

    qsort (myarray, NUM_ARRAY_ELEMS, sizeof(char *), compare);

    printf ("array after sorting:\n");

    for (i = 0; i < 5; i++)
            printf ("myarray [%d] = %s\n", i,myarray[i]);

}
```