Programming Fundamentals
Lecture #12
Arrays

Junaid Hassan

Lecturer CS & IT Department UOS MBDIN

junaidte14@gmail.com

Passing Arrays to Functions

To pass an array argument to a function, specify the name of the array without any brackets. For example, if array hourlyTemperatures has been defined as :

int hourlyTemperatures[24];

the function call:

modifyArray(hourlyTemperatures, 24) ;

passes array hourlyTemperatures and its size to function modifyArray.

- C automatically passes arrays to functions by reference—the called functions can modify the element values in the callers' original arrays
- Array name is really the address of the first element of an array by printing array, &array[0] and &array using the %p conversion specifier— a special conversion specifier for printing addresses

```c
/* Fig. 6.12: fig06_12.c
   The name of an array is the same as &array[ 0 ] */
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
   char array[ 5 ]; /* define an array of size 5 */

   printf( "    array = %p\n&array[0] = %p\n   &array = %p\n",
      array, &array[ 0 ], &array );
   return 0; /* indicates successful termination */
} /* end main */
```

```
    array = 0012FF78
&array[0] = 0012FF78
   &array = 0012FF78
```

For a function to receive an array through a function call, the function's parameter list must specify that an array will be received. For example, the function header for function modifyArray (that we called earlier in this section) might be written as:

void modifyArray(int b[], int size);

Sorting data (i.e., placing the data into a particular order such as ascending or descending) is one of the most important computing applications

A bank sorts all checks by account number so that it can prepare individual bank statements at the end of each month

To sort array elements different sorting algorithms are available like:

- Bubble Sort
- Quick Sort

- Merge Sort
- Heap Sort

etc….

- As an example we will use Bubble Sort technique/algorithm to sort arrays in ascending order
- This technique we use is called the bubble sort or the sinking sort because the smaller values gradually “bubble” their way upward to the top of the array like air bubbles rising in water, while the larger values sink to the bottom of the array

- On each pass, successive pairs of elements are compared.
- If a pair is in increasing order (or if the values are identical), we leave the values as they are
- If a pair is in decreasing order, their values are swapped in the array

```
/* bubble sort */
/* loop to control number of passes */
for ( pass = 1; pass < SIZE; pass++ ) {

   /* loop to control number of comparisons per pass */
   for ( i = 0; i < SIZE - 1; i++ ) {

      /* compare adjacent elements and swap them if first
      element is greater than second element */
      if ( a[ i ] > a[ i + 1 ] ) {
         hold = a[ i ];
         a[ i ] = a[ i + 1 ];
         a[ i + 1 ] = hold;
      } /* end if */
   } /* end inner for */
} /* end outer for */
```