Arrays- III

CS10001: Programming & Data Structures

Sudeshna Sarkar
Dept. of Computer Sc. & Engg.,
Indian Institute of Technology Kharagpur

Array and pointer

- The expression a+e is the address of location a[e]

$$\&a[e] \equiv a+e$$

$$*(a+e) \equiv a[e]$$

- The *i*th location of a 1-D array a[ ] of type int starts from the address a + i∗sizeof(int)

Pointer Arithmetic

```
#include <stdio.h>
int main () {
        char c[5], *cp;
        int i[5], *ip;
        double d[5], *dp;

        printf (“c: %p, i:%p, d:%p”, c, i, d) ;
        printf (“c+1: %p, i+1:%p, d+1:%p”, c+1, i+1, d+1) ;

        cp = c; ip=i; dp=d;
        printf (“cp: %p, ip:%p, dp:%p”, cp, ip, dp) ;
        printf (“cp+1: %p, ip+1:%p, dp+1:%p”, cp+1, ip+1, dp+1) ;
        return 0;
}
```

Passing an array to a function

- Formal parameter int x[] , or int *x
- The formal parameter x receives the address of an int location. It is usually treated as a starting address of an 1-d array. But it is essentially a pointer of type int

```
int main () {
    int arr[50];
    ………
    reverse (arr, 20);
    …..
}
void reverse (int x[], int size)  {
     ……
}
```

Reverse: iterative version

```
void reverse (int x[], int size)  {
    int i, temp;
    for (i=0; i< (size); i++)
        temp = x[size-i-1] ;
        x[size-1-1] = x[i] ;
        x[i] = temp;
}
```

Reverse: iterative version

```
void reverse (int x[], int size)  {
    int i, temp;
    for (i=0; i< (size/2); i++)
        temp = x[size-i-1] ;
        x[size-1-1] = x[i] ;
        x[i] = temp;
}
```

Reverse: recursive version

```
void reverse (int x[], int size)  {
    int temp;
    if (size <= 1)
        return ;
    /* swap x[0] and x[size-1]  */
    temp = x[0];
    x[0] = x[size-1];
    x[size-1] = temp;

    reverse (x+1, size-2) ;
}
```

findmax: iterative

```
int findmax (int x[], int size) {
    int i, max;
    max = x[0];
    for (i=1; i< size; i++)
        if (x[i] > max)
            max = x[i] ;
    return max;
}
```

findmax: recursive

```
int findmax (int x[], int size) {
    int maxofrest;

    if (size == 1)
        return x[0];

    maxofrest = findmax (x, size-1) ;
    return (x[size-1] > maxofrest? x[size-1]:maxofrest) ;
   }
```

Arrays as Output Parameters

```
void VectorSum (int a[], int b[], int vsum[],int length){
    int i;
    for (i=0; i<length; i=i+1)
        vsum[i] = a[i] + b[i] ;
}
int main ( ) {
    int x[3] = {1,2,3}, y[3] = {4,5,6}, z[3];
    VectorSum (x, y, z, 3) ;
    PrintVector (z, 3) ;
}
```

```
void PrintVector (int a[], int length) {
    int i;
    for (i=0; i<length; i++) printf (“%d “, a[i]);
}
```

Strings

- **Strings are 1-dimensional arrays of type char.**
- **By convention, a string in C is terminated by the end-of-string sentinel \0, or null character.**
- **String constant : “abc” is a character array of size 4, with the last element being the null chaaracter \0.**
- **char s[ ] = “abc” ;**

a	b	c	\0