C under Linux

Dr. Naeem Odat

Department of Computer and Communications Engineering

C - Arrays

C - Arrays

C - Arrays

Arrays

- Represent homogeneous data.
- Allocated memory is contiguous.
- Number of elements must be positive.
- Number of elements need not be specified, if the array is initialized.
- The number of elements in the array is given by a constant-expression.
- The first element in the array is at index zero, and the last element is at *(n-1)* index, where *n* is the size of the array.

C - Arrays

Declaring an array

```
<type> <name> [<constant-expression>];
```

Example

```
int nArray[10];
```

C - Arrays

Initializing arrays

```
int a[5] = {1, 2, 3, 4, 5};
int a[] = {1, 2, 3, 4, 5};/*Here size of the array is same
                          as the number of elements specified.*/
int a[5] = {1, 2, 3};/*The remaining elements
                      are initialized to 0.*/
```

C - Arrays

Accessing elements in arrays

```
#include<stdio.h>
int main(){
    int nArr[5] = {1, 2, 3, 4, 5};
    int nCtr;
    for(nCtr = 0; nCtr < 5; nCtr++)
        printf("%d\n", nArr[nCtr]);
    return(0);
}
```

C - Arrays

What is wrong?

```
#include<stdio.h>
int main(){
    int nArr[5] = {1, 2, 3, 4, 5};
    int nCtr;
    for (nCtr = 1; nCtr <= 5; nCtr++)
        printf("%d\n", nArr[nCtr]);
    return(0);
}
```

C - Arrays

What is the output?

```
#include<stdio.h>
int main(){
    int nSize = 5;
    int nArr[nSize] = {1, 2, 3, 4, 5};
    for (nSize -= 1; nSize >=0; nSize--)
        printf("%d\n", nArr[nSize]);
    return(0);
}
```

C - Arrays

Multidimensional arrays

- Elements are represented in (Row, Column) format.
- C stores the elements as first row elements, followed by second row elements, and so on.
- When initializing, all subscripts except the first must be specified, example:

```
int nArray[][3] = {1, 2, 3, 4, 5, 6};
/* if we are not specifying the second column, it will
result in compilation error*/
```

C - Arrays

Initializing a multidimensional array

```
int a[2][3] = {1, 2, 3, 4, 5, 6};/*Elements are assigned from
                                   a[0][0], a[0][1], a[0][2],
                                   a[1][0], and so on.*/
int a[2][3] = {{1, 2, 3}, {4, 5, 6}};
int a[][3] = {1, 2, 3, 4, 5, 6};
int a[][3] =  { {1, 2, 3}, {4, 5, 6}};/*All the subscripts excep
                                       for the first one are
                                       necessary when assigning
                                       with values.*/
int a[2][2][2] = {{{1, 2}, {3, 4}}, {{5, 6}, {7, 8}}};
```

C - Arrays

Accessing elements

```
int  nMyArray[2][5] = {{1, 2, 3, 4, 5},{6, 7, 8, 9, 10}};
//How to access an individual element

nMyArray[0][4]  //5
nMyArray[1][2]  //8
```

C - Arrays

Copying arrays

```
char a[10] = {1, 2, 3, ..., 10};
char b[10];

b = a; /* Not legal*/
```

Copying arrays

- Copy elements one by one.
- Use a loop.