C Programming Basics

SDS 322/329

October 27, 2015

Email any questions to:
rauta@tacc.utexas.edu

TACC

THE UNIVERSITY OF TEXAS AT AUSTIN

Recap

- Control Structures
  - Sequence structure
  - Selection structure
    - if-else
    - Ternary operator
    - if- else if – else
    - switch-case
  - Loop structure
    - for-loop
    - while-loop
    - do-while loop

Overview of the Course

- Writing a Basic C Program
- Understanding Errors
- Comments Keywords, Identifiers, Variables
- Operators
- Standard Input and Output
- Control Structures
- Arrays, Structures
- Functions in C
- Pointers
- Working with Files

All the concepts will be accompanied with examples.

Arrays

- An array is a multivariable that allows you to store many different values of same data type in a single unit and in contiguous memory locations

- You can have arrays of any valid data type in C (**not `void` though**)

- Arrays are declared just like other variables, though the variable name ends with a set of square brackets
  - `char myName[50];` <---- You have seen this before
  - `int myVector[3];` //one-dimensional array
  - `int myMatrix[3][3];` //two-dimensional array

Initializing Arrays

- The content of the array is undetermined till you store any value in it

- Method 1

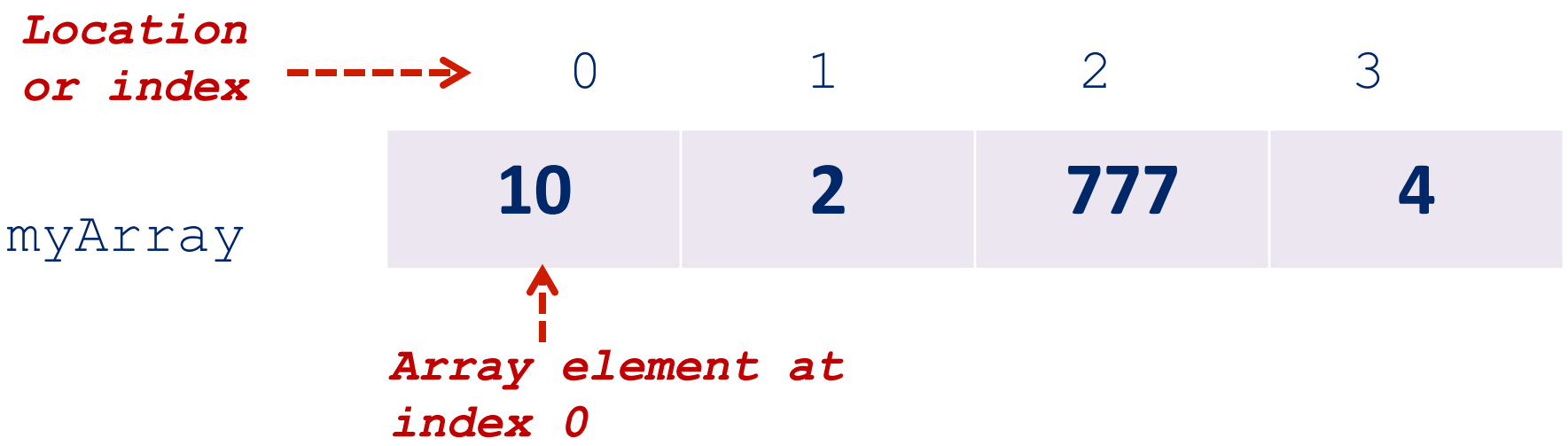
```
int myArray[4] = { 10, 2, 777, 4};
```

- Method 2

```
for (i =0; i<4; i++){
    scanf("%d",&myArray[i]);
}
```

Initializing Arrays

- Upon declaration and initialization an array is created like:




```
myArray[3]
myArray[1]
myArray[2]???
```

Computing With Arrays

- Access the array element

  `yArray[i]` where `i` is the index of the array

- Use it in computation like a regular variable

```
for (i=0; i< 3; i++){
  xArray[i] = yArray[i] + zArray[i];
}
xArray[i] = yArray[i] + zArray[i];
```

Arrays Example: arrayExample.c

```
#include <stdio.h>
int main(){
   int i;
   int age[4];
   age[0]=23;
   age[1]=34;
   age[2]=65;
   age[3]=74;
   for(i=0; i<4; i++){
     printf("age[%d]: %d\n", i, age[i]);
   }
   return 0;
}
```

Note: The number in the square brackets is the position number of a particular array element. Notice that count begins at 0

Output:

```
age[0]: 23
age[1]: 34
age[2]: 65
age[3]: 74
```

Multi-dimensional Arrays

- An array-of-arrays is called a multi-dimensional array
- A 2-dimensional array, `myArray` with 3 rows and 3 columns looks like:

	0		1		2	
0	10	(0,0)	42	(0,1)	3	(0,2)
1	24		53		62	
2	77		84		97	

```
myArray[0][0] = 10;
myArray[1][2] = ???
```

2-D Arravs: arrav2D.c

```
1.  #include <stdio.h>
2.  int main(){
3.    int i, j;
4.    int xArray[2][2] = {{1, 2}, {3, 4}};
5.    int yArray[2][2] = {{1,2},{3,4}};
6.    int zArray[2][2] = {{0,0},{0,0}};
7.    for (i=0; i< 2; i++){
8.      for (j=0; j <2; j++){
9.         zArray[i][j] = xArray[i][j] + yArray[i][j];
10.     }
11.   }
12.   for (i=0; i< 2; i++){
13.     for (j=0; j <2; j++){
14.        printf(" %d ", zArray[i][j]);
15.     }
16.     printf("\n");
17.   }
18.   return 0;
19. }
```

Note the two sub-lists, inside the main list

Nesting of loops: A for-loop inside another for-loop

Structures

- Multiple variables can be combined into a single package called structure
- Members of the structure variable need not be of the same type
- They can be used to do database work in C! Example:

```
struct sample{
   int a;
   char b;
};
struct sample mySample;
```

- `typedef` is the keyword that can be used to simplify the usage of `struct`

```
typedef struct sample newType;
```

Structure Example: structExample.c

```
#include <stdio.h>

typedef struct point{
  double x;
  double y;
}point;

int main(){
  point myPoint;
  myPoint.x = 12.2;
  myPoint.y = 13.3;
  printf("X is %lf and Y is %lf\n",myPoint.x, myPoint.y);
  return 0;
}
```

Declaring a variable of type structure

Notice the "." operator

References

- C Programming Language, Brian Kernighan and Dennis Ritchie
- Let Us C, Yashavant Kanetkar
- C for Dummies, Dan Gookin
- http://cplusplus.com
- http://www.cprogramming.com/tutorial/c/lesson11.html