Computer Programming Using C
COP 3275 - Summer 2017

Lecture 13: Functions

By Ahmed E. Khaled
Recap to previous lecture!

• 1D and 2D Arrays
• Develop a program that find the second largest element in an array

```
max1 ← arr[0];
max2 ← arr[0];
For i ← 1 to N do
  If (arr[i] > max1) then
    max2 ← max1;
    max1 ← arr[i];
  Else if (arr[i] > max2) then
    max2 ← arr[i];
```
Multidimensional Arrays

• *Nested for loops* are ideal for processing multidimensional arrays.

• Consider the problem of initializing an array for use as an identity matrix. A pair of nested for loops is perfect:

```c
#define N 10

double ident[N][N];
int row, col;

for (row = 0; row < N; row++)
    for (col = 0; col < N; col++)
        if (row == col)
            ident[row][col] = 1.0;
        else
            ident[row][col] = 0.0;
```
Functions *(or methods)* in C
• A function is a *series of statements* grouped together with a scope and given a name.

• A function *avoids duplicating code*, as it is used to execute certain functionality more than once (reusable).

• Think of the function as a *small program* within your program, easy to understand and modified.
Function Definition!

#include libraries

Function1 // function definition
Function2 // function definition

main(){
  Function1 // function call
  Function2 // function call
}

Function Definition!

```plaintext
return-type function-name ( parameters )
{
    declarations
    statements
}
```
• The *return-type* of a function is the type of value that the function *returns*.

• Example, function that multiplies two integers should return an *integer value* (the result).

• Some functions may not return values, *void* is used to indicate such case.
• After the function name comes a list of parameters as input to the function, separated by commas.

• Each parameter is defined by the type followed by the name.
• Develop a function that adds two integer values and return the result value.

```c
int add(int a, int b)    //define function
{
    int c;
    c = a + b;
    return c;
}
main()  {
    int x = 3;   int y = 5;   int z;
    z = add(x,y);          //call function
    printf("%d",z);
}
```
Function Calls

• int z = add (3, 5);

• if (add(3, 5) > 4)
   
   printf("result is higher than 4\n");

else

   printf("result is equal or lower than 4\n");
• `printf(“Summation: %d\n”, add(x, y));`

• The value returned by a non-`void` function can always be discarded *if it’s not needed*:

```
add(x, y);  // discards return value
```
Develop a function that multiplies the input value by three and return the result value.

```c
int Mul_Three(int a)    //define function
{
    int c = a * 3;
    return c;
}

main() {
    int z;
    z = Mul_Three(5);      //call function
    printf("%d",z);
}
```
• Develop a function that returns the average of two double values and return the result value.

```c
double average(double val1, double val2) {
    double avg = (val1 + val2) / 2;
    return avg;
}
main() {
    double x = average(12, 6);
    printf("%lf", x);
    double y = average(18, 4);
    printf("%lf", y);
    double z = average(12.7, 3.5);
    printf("%lf", z);
}
```
Develop a function that prints welcome message.

```c
void print_welcome()
{
    printf("welcome !! \n");
}

main()
{
    print_welcome();
    print_welcome();
}
```
Develop a function that prints welcome message N times, where N is an input from the user.

```c
void display_welcome(int n) {
    int i=0;
    for(i=0;i<n;i++){
        printf("welcome !! \n");
    }
}

main() {
    display_welcome(4);
}
```
• Develop a function that print *countdown* from the integer input to zero.
• Develop a function that *prints* the average and summation of three double inputs.
• Develop a function that returns the maximum of three integer inputs.
• Develop a function that returns *true*, if the summation of the two integer inputs is larger than 500.
• Develop a function that returns the value of X to the power of Y, where X and Y are integer inputs.
• Develop a function that print *all even* numbers between two integer inputs.