C Functions

COEN 10

C - Lecture 7

Functions

★A function is a self-contained unit of program code designed to accomplish a particular task

★In other languages

- ◎functions
- ◎subroutines
- ◎procedures

Functions

★Functions

- ◎Produce actions and/or
- ◎Provide values

Functions

★Why do we use functions?

- ◎To avoid repetitions
- ◎To enable reuse
- ◎To make the program more modular, easier to read, change, and fix

Functions

★The black-box concept

◎Once a function is created and tested, it can be trusted to do what it is supposed to

Functions

★Stack behavior

◎Program starts at the main function

◎Main functions may call other functions, which may call other functions

◎When a function is done, it return the control flow to the function that initiated the call

◎Eventually, the control flow returns to main, which ends the program

Creating and using a function

★Function prototype

◎Defines the type returned by the function and the types of the arguments it receives

◎Format

```
<type> <name> (<type>, <type>, …);
```

◎Example

```
int max (int, int);
```

Creating and using a function

★Function call

◎In the code, causes the function to be executed

◎Format

```
<ret> = <name> (<arg>, <arg>, …);
```

◎Example

```
x = max (a, b);
```

Creating and using a function

★Function definition

◎Specifies exactly what the function does

◎Format

```
<type>
<name> (<type> <arg>, <type> <arg>, …)
{
        declarations
        statements
        return <value of specified type>
}
```

Creating and using a function

★Function definition

◎Example

```
int
max (int x, int y)
{
        if (x > y)
            return x;
        else
            return y;
}
```

Creating and using a function

★void functions

◎Functions that do not return any values

★void argument

◎No arguments

Creating and using a function

★void -- format

◎Function Prototype

```
void <name> (void);
```

◎Function Call

```
<name> ( );
```

Creating and using a function

★void -- Format

◎Function Definition

```
void
<name> (void)
{
    declarations
    statements
    return;
}
```

Creating and using a function

★void

◎Example

✧Prototype

```
void print_alphabet (void);
```

✧Call

```
print_alphabet ( );
```

Creating and using a function

★void

◎Example -- definition

```
void
print_alphabet (void)
{
    char    c = 'A';
    int     i;
    for (i = 0; i < 26; i++)
        printf ("%c", c++);
    return;
}
```

Variables

★Scope

◎Local

✧Declared inside functions

✧Their name is only known inside the function

◎Global

✧Declared outside functions

✧Their name is known to all the functions in the same file

Variables

- ★Lifetime
  - ◎Local variables
    - ✧Only exist when the declaring function is executing
  - ◎Global variables
    - ✧Exist during the total execution of the program

Program Layout

- ★includes
- ★defines
- ★function prototypes
- ★global variables
- ★main function
- ★other functions