Intro to C Programming Language

- It was developed in Bell Labs in the early 1970s to write OSs.
- Ability to work with individual bits and bytes in memory (control the memory)
- It has No classes, No Objects, No methods.
- The main function must be called main.
- To compile the program, use:

gcc hello.c	or	gcc -o FileName hello.c
./a.out		./FileName

- Data types:

Type	Format Specifier	Storage size	Value range
char	%c	1 byte	-128 to 127 or 0 to 255
unsigned char	%c	1 byte	0 to 255
signed char	%c	1 byte	-128 to 127
int	%d	2 or 4 bytes	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
unsigned int	%i	2 or 4 bytes	0 to 65,535 or 0 to 2^{32}-1
short	%hi	2 bytes	-32,768 to 32,767
unsigned short	%hu	2 bytes	0 to 65,535
long	%l	4 or 8 bytes	-9223372036854775808 to 9223372036854775807
unsigned long	%lu	8 bytes	0 to 18446744073709551615
long long	%lld	8 bytes	
float	%f	32	1.2E-38 to 3.4E+38
double	%lf	64	2.3E-308 to 1.7E+308
Long double	%Lf	10 bytes	3.4E-4932 to 1.1E+4932

- There are no booleans (int: 0=F, any other value=T).
- You can assign Hex values (use %x as format specifier for printing),for example:

```
int x=0x43a2c
printf("%x",x);
```

- No string data type, but you can define String, as array of characters:

```
char str[100];
fgets(str, sizeof(str), stdin);
```

- There two types of comments in c: // or /* */
- Main c instructions:
    - Arithmetics: -, *, /, %, =, ++,–, +=,-+
    - Comparison: > < >= <= == !=
    - Logical: ! && ||
    - Bitwise operators: | & >> << ~
    - printf() \n, \\, \', \t
    - scanf("%d %d %d", &first, &second, & third)
    - c = getchar();
    - for, while, do{}while();
    - If, if else, switch
- Global variable:
    - Are defined outside any function.
    - Can be accessed inside any function.
- Local variables:
    - Are declared inside a function or a block.
    - Are used only by the statements that are inside that block of code.
- Arrays:
    - Not Objects but just a sequence of values in memory, for example:

```
int a[5];        // is five 4-bytes integer values.
```

- You can define a collection of data using "struct", for example:

```
struct Books {
  char  title[50];
  char  author[50];
  char  subject[100];
  int   book_id;
};
struct Books b1;
```