Lecture 2

Basic Elements of Programming Language C
Variables. Declaration and Initialization. Basic Data Types.

A variable is a name of location in the memory of computer for storing one or more values. If at this location only one value is stored, corresponding variable is named **simple or scalar variable**. And if at the given location more values are stored corresponding variable is named **composed or structured variable.**

In C language variables can be of different data types**. Data type** determines specific properties of variables of given type such as size of memory for one variable, range of data and operations on them. There are **5 basic data types** in C language determined by corresponding keyword: **int** (2 bytes) for integers, **float** (4 bytes) and **double** (8 bytes) for real numbers, **char** (1 byte) for characters and **void** (empty) used especially for functions.

In C language variables have to be declared at the very beginning of the block (immediately after opening brace). Declaration of a simple variable has general form:

type name_of_varifble**;** Ex.: **int** a**; float** b, c; **double** d; **char** e, f , h;

It is possible **initialize** a variable during declaration. Ex.: int a=2; float b=4.5, c=-6.7.

Using Variables

So far you should be able to write a simple program to display information typed in by you, the programmer and to describe your program with comments. That's great, but what about interacting with your user? Fortunately, it is also possible for your program to accept input.

But first, before you try to receive input, you must have a place to store that input. In programming, input and data are stored in variables. There are several different types of variables; when you tell the compiler you are declaring a variable, you must include the data type along with the name of the variable. Several basic types include char, int, and float. Each type can store different types of data.

A variable of type char stores a single character, variables of type int store integers (numbers without decimal places), and variables of type float store numbers with decimal places. Each of these variable types - char, int, and float - is each a keyword that you use when you declare a variable. Some variables also use more of the computer's memory to store their values.

It may seem strange to have multiple variable types when it seems like some variable types are

redundant. But using the right variable size can be important for making your program efficient because some variables require more memory than others. For now, suffice it to say that the different variable types will almost all be used!

Before you can use a variable, you must tell the compiler about it by declaring it and telling the compiler about what its "type" is. To declare a variable you use the syntax <variable type> <name of variable>;. (The brackets here indicate that your replace the expression with text described within the brackets.) For instance, a basic variable declaration might look like this:

```
int myVariable;
```

Note once again the use of a semicolon at the end of the line. Even though we're not calling a function, a semicolon is still required at the end of the "expression". This code would create a variable called myVariable; now we are free to use myVariable later in the program.

It is permissible to declare multiple variables of the same type on the same line; each one should be separated by a comma. If you attempt to use an undefined variable, your program will not run, and you will receive an error message informing you that you have made a mistake.

Here are some variable declaration examples:

```
int x;
int a, b, c, d;
char letter;
float the_float;
```

While you can have multiple variables of the same type, you cannot have multiple variables with the same name. Moreover, you cannot have variables and functions with the same name.

A final restriction on variables is that variable declarations must come before other types of statements in the given "code block" (a code block is just a segment of code surrounded by { and }). So in C language (not in C++) you must declare all of your variables before you do anything else:

Wrong

```
#include <stdio.h>
int main( )
{
   /* wrong!  The variable declaration must appear first */
   printf( "Declare x next" );
   int x;

   return 0;
}
```