Welcome to the course !

In this course, we will learn about C programming. The main emphasis of the lectures will be on introducing you to various aspects of C Language along with how to solve problems at an abstract level using computer. In the laboratory, you will implement the features of C and the ideas taught in class into real programs using the C programming language.

The lab assignments will be uploaded on my course webpage as well as on moodle (Will be functional soon). There will be several graded lab assignments in the course where you will have to submit C programs before a fixed deadline. You will be uploading your assignments on moodle. The assignments will be graded by the TAs and the marks (along with comments) will be entered on moodle, where you can see them. Make sure your programs are correct and running before submitting them. Now to today's lab.

This is the first lab and you should use this opportunity to get used to the programming environment. All the machines in the lab run some version of the LINUX operating system. There are a lot of interesting things you can do on LINUX, but for the course it is enough that you are able to write programs in C and execute them on the computer as shown in the demo in the class.

A beginner's UNIX/LINUX tutorial can be found here:
http://info.ee.surrey.ac.uk/Teaching/Unix/

The first few chapters will get you started. To write, save and edit programs (which are text files), you need a text editor. The text editor available on UNIX is VI (or, alternatively VIM). A very good VIM tutorial is available here: **http://www.linuxconfig.org/Vim_Tutorial**

Do go through the video links in the tutorial to understand how VI is used in practice. If you are unable to follow something, or are completely new to computers, feel free to ask questions. The TAs will be available during lab hours and will be more than happy to answer your questions.

One more thing before we proceed further. Feel free to use the internet and the library to learn more about UNIX and C programming language. The classic text on C programming is the following:

The C programming language by Brian W. Kernighan and Dennis M. Ritchie.

In this lab, we will run a few very basic C programs, run them on the computer and understand their behaviour.

The first is the well-known **"Hello World"** program, also displaying multiple ways to use printf format string. The text between /* ---- */ is a comment and will not affect the outcome of your program.

```
-------------------------------------------------------------------------
#include <stdio.h>

void main() {

        /* I am printing three times */
        /* printf ( " <format string> ", .. , .. , ..); */
        /* We have learnt : \n , %d, printf */
        /* %d --> base 10 integer , %f --> floating point number */
        /* %e --> scientific notation , %g --> chooses between %e and %f*/
        /* %x --> hexadecimal (base 16), %o --> octal (base 8) */
        /* %# --> enhances readability of %x and %o */
        /* %c --> character (single letter). It is necessary to write a
           character between
           ' and ' */
        /* %s --> string */
        /* %.2f --> 2 digits after the decimal point */
        /* %6d --> Print 6 digits (if digits are not there, replace by spaces
           */
        /* You want to print %, Write %% in the format string */
        /* You want to print \, write \\ in the format string */

        printf("Hello, %s \n", "World");

}
------------------------------------------------------------------------------
```

Type this program into a file **"hello_world.c"** using vi editor for example **"vi hello_world.c"** on the terminal. To compile, type **"gcc hello_world.c"**. This will produce an executable file **"a.out"** in the same directory. Now, to run the program, just type **"./a.out"**.

Congrats !!! you have successfully completed your first program and welcome to the world of programming. Now try various format strings of the printf statement, for example **"printf("%.2f \n", 1.234567)"**

The second program takes two integers as input from the user and swaps them i.e., interchanges their values.

```
----------------------------------------------------------------------------

/* This program reads two integers a and b and swaps them */

#include <stdio.h>

int main() {

        int a, b;
        int temp; /* The temporary variable used to swap the two numbers */

        printf("Enter the first number: \n");
        scanf("%d", &a);
        printf("Enter the second number: \n");
        scanf("%d", &b);

        printf("Before swapping : a = %d, b = %d\n", a, b);

        /* Do the swap */
        temp = a;
        a = b;
        b = temp;

        printf("After swapping : a = %d, b = %d\n", a, b);
}

----------------------------------------------------------------------------
```

Copy this program to a file "swap.c" on your terminal. Then compile and run it as you did on the last example. As an exercise, try to understand the meaning of various statements in this program and how it works. You can freely use the internet for this.

The third program takes input as a dividend and divisor. If the divisor is equal to zero then it stops the execution with a "return" statement from the main function. Observe the use of the control statement "if".

```
------------------------------------------------------------------------------

#include <stdio.h>

int main() {

        /* In C : first declare a variable and then use it */
        /* Format --> type variable_name */
        /* scanf, use of the & operator before a variable name */
        /* variables : declaration, definition */
        /* variables follow rules of algebra */
        /* mathematical operators: + - * / % */
        /* operater short cuts : += *= -= /= %= */
        /* int --> integer, float --> floating point, char --> character */
        /* variables : naming, start with [a-z,A-Z,_] after that they can
           contain [a-z,A-Z,_,0-9]*/
        /* ASCII Character Set: to find out print a character as an integer */
        /* Conversion (typecast) operator: integer_variable = (int)
```

```
           float_variable , and similar */
        /* i++ --> i = i + 1 return old value of i, ++i --> i=i+1, return new
           value */
        /* = means assigned to, == is equal to */


        /* PROGRAM STARTS */
        int dividend, divisor, quotient;
        printf ("Enter dividend and divisor: ");
        scanf ("%d %d", &dividend, &divisor);

        if (divisor == 0) { // a curly bracket
                printf ("Invalid value of divisor. Bye bye !!! \n");
                return;
        } // curly bracket --> end of the if part

        quotient = dividend / divisor;

        printf ("Result is : %d",quotient);
        /* PROGRAM ENDS */
}

------------------------------------------------------------------------------
```

Run the programs like the previous two programs. Modify this program to print the remainder as well. Hint: search for the remainder operator in C.

That's all for today. Once you are adjusted to the programming environment, we will start actual programming. You can also try some of the following practice commands and programs as given below. All the best **:-)**

Practice Commands

- Find the C compiler version of your system.
- Use man command to see the predefined functions of stdio
- Use man command to see the syntax of printf function

Practice Programs

- Write the program to find the size of all data types.
- Write the program to find the minimum and maximum value of the integer and float.
- Write a program to analyze and find the maximum size integer can support in your system compiler.