

# **Getting Started**

- I assume students taking this course, have a programming background either in **C++** or **Processing**.
  - These are the two programming languages that have been taught in ENGG 233 in the last few years.
  - Learning C is somewhat easier for those who have a C++ background. But the good news is that many of the basic constructs of the three languages are similar.
- The development environment and tools that we will use in our ICT 320 lab include:
  - Operating System: **Windows**
  - Development Environment: **Cygwin**
    - Cygwin is used to give you a very basic understanding of Linux operating system.
    - Cygwin is a Unix-like environment and command-line interface for Microsoft Windows.
- You might be already familiar with the development environments such Visual Studio, Eclipse , Xcode , etc.
  - If these are your preferred tools, for most of the assignments you should be OK.
  - Exceptionally, we may recommend you use an specific tool.

# **Similarities Between C, C++, and Processing**

- To help you realizing how a C program is similar or different from C++ or Processing, we will write a simple program in these languages.
- For the simplicity purposes our program will not interact with the user.
  - We will write a program that calculates and displays the value of  $x^n$  ( $x$  to the power of  $n$ )
    - First we will try it in **Processing**, for those who have take ENGG 233 last fall.
    - then in **C++**, for those who have taken ENGG 233 in fall 2014 or earlier.
    - And, finally we will take a quick look at its C version of the program.

# **Screenshot: A Simple Processing Program**

The screenshot shows the Processing 3.0.2 IDE interface. At the top, there are three circular icons (top-left), a file icon followed by "simpleProgram | Processing 3.0.2" (top-center), and a circular icon with two vertical bars and a dropdown menu labeled "Java" (top-right). Below the header is a toolbar with four circular buttons: play, stop, run, and square. The main workspace is titled "simpleProgram" and contains the following Java code:

```
1
2 void setup()
3 {
4     float x = 5;
5     float n = 2;
6     float result;
7     println("This program displays the result of x to the power of n.");
8
9     result = pow(x, n);
10    print("The result of " + x + " to the power of " + n + " is: " + result);
11 }
```

Below the code, a message "Auto Format finished." is displayed. In the bottom right corner of the workspace, the output of the program is shown: "This program displays the result of x to the power of n." and "The result of 5.0 to the power of 2.0 is: 25.0". At the bottom of the screen, there are two tabs: "Console" (with a right-pointing arrow icon) and "Errors" (with a warning triangle icon).

# **Screenshot: A Simple C++ Program**

Using Visual Studio IDE

# Editing and Running a Simple C++ Program Using Visual C++

The screenshot shows the Microsoft Visual Studio interface with the title bar "simpleProgram - Microsoft Visual Studio". The menu bar includes FILE, EDIT, VIEW, PROJECT, BUILD, DEBUG, TEAM, TOOLS, TEST, ANALYZE, WINDOW, and HELP. The toolbar has icons for file operations like Open, Save, and Build. The status bar at the bottom shows "This program displays the result of x to the power of n." and "The result of 5 to the power of 2 is: 25".

The code editor window displays the file "simpleProgram.cpp" with the following content:

```
#include <iostream>
#include <cmath>
using namespace std;

int main(void)
{
    cout << "This program displays the result of x to the power of n." << endl;
    float x = 5;
    float n = 2;
    float result;
    result = pow(x, n);
    cout << "The result of " << x << " to the power of " << n << " is: " << result << endl;
    return 0;
}
```

This program displays the result of x to the power of n.  
The result of 5 to the power of 2 is: 25

# **Screenshot: A Simple C Program**

Using Notepad++ Editor

## Editing a Simple C Program, Using Notepad++, and Cygwin Terminal

The screenshot shows the Notepad++ interface with the file `simpleProgram.c` open. The code is a simple C program that includes `<stdio.h>` and `<math.h>`, defines a function `main` that prints a message and calculates  $x^n$  using `pow(x, n)`, and returns 0. The status bar at the bottom shows the file has 284 length, 12 lines, and is in Ln:1 Col:4 Sel:0|0 mode, with encoding set to Dos\Windows and ANSI as UTF-8.

```
1 #include <stdio.h>
2 #include <math.h>
3 int main(void)
4 {
5     printf("This program displays the result of x to the power of n.\n");
6     float x = 5;
7     float n = 2;
8     float result;
9     result = pow(x, n);
10    printf("The result of %f to the power of %f is: %f", x, n, result);
11    return 0;
12 }
```

The screenshot shows a Cygwin terminal window titled `~/Desktop` displaying the output of the program. The output shows the program's message and the result of calculating  $5^2$ .

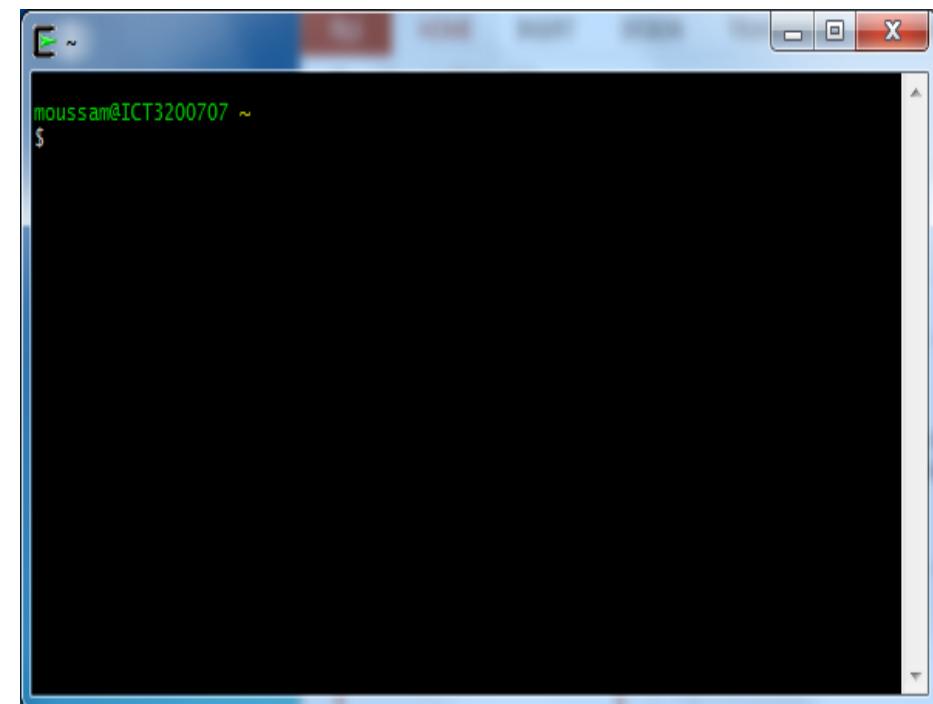
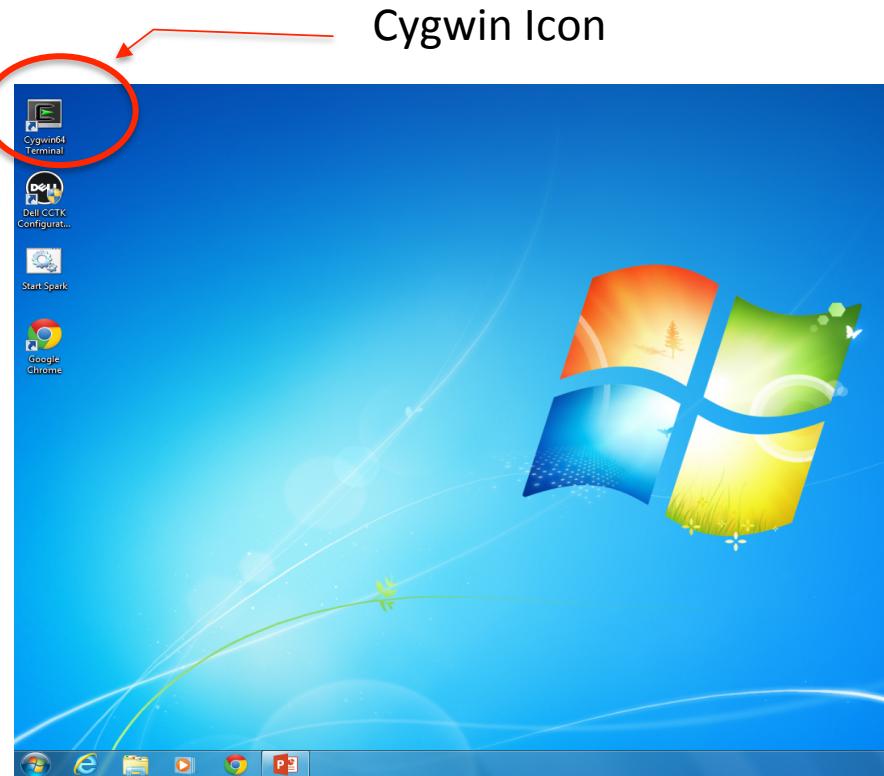
```
This program displays the result of x to the power of n.
The result of 5.000000 to the power of 2.000000 is: 25.000000
moussam@ICT3201101 ~/Desktop
```

# Introduction to Program Development Environment in ICT 320 Computer Lab

# **Quick Introduction to Cygwin**

## How to Start Cygwin Terminal Window

- The easiest way to open a Cygwin terminal window under Windows operating systems is to double click on the Cygwin-Terminal icon that appears on your desktop window on the computers in our ICT 320 lab.
- Here is an example of Cygwin Terminal Window



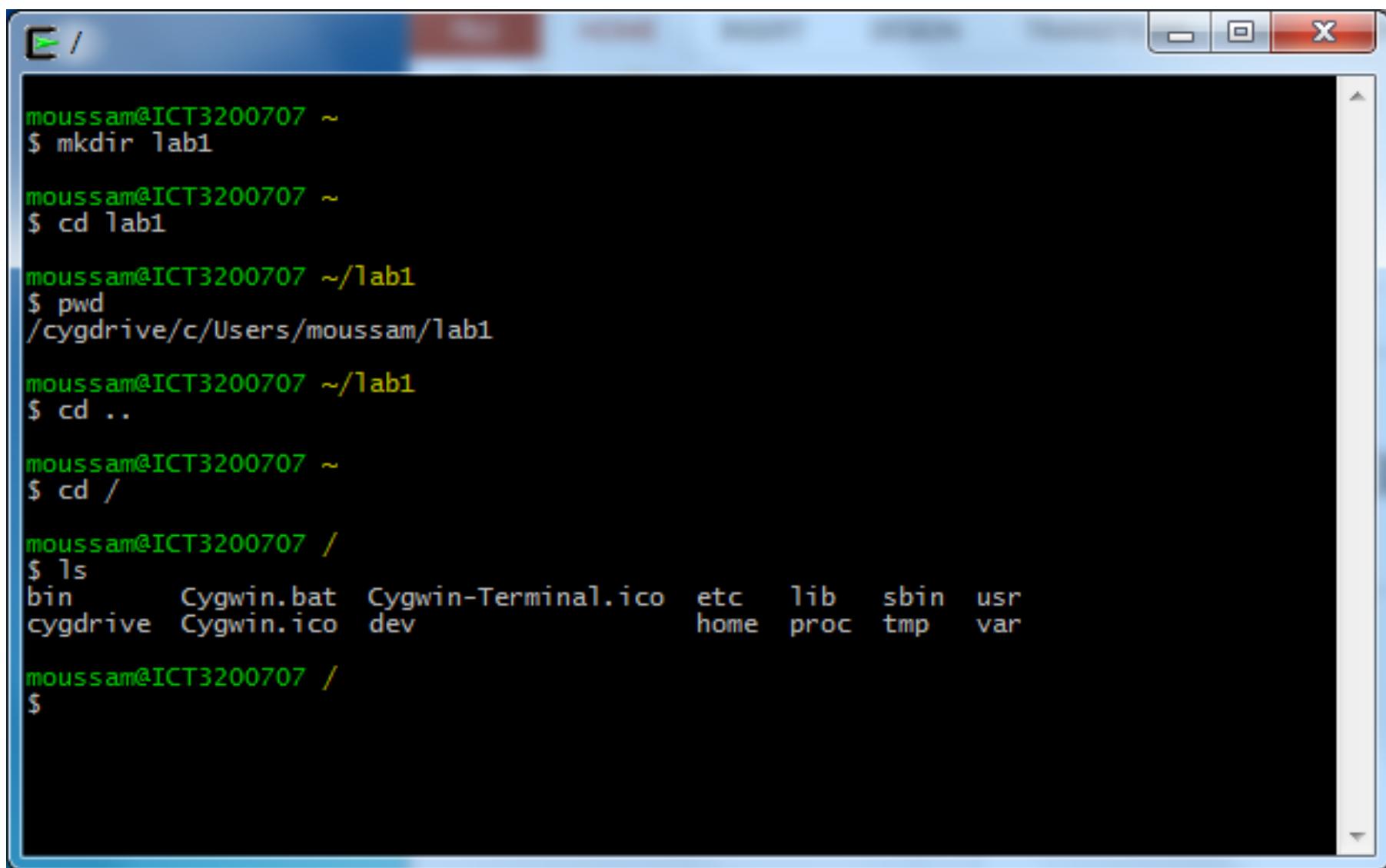
## Linux File System

- The Linux file system is composed of directories and files.
  - They are arranged in a hierarchical structure, as specified by the user.
  - The root directory is slash: /
- File names can be up to 255 characters.
  - Can use case-sensitive letters, numbers, “\_”, “.”, “ ”, ‘,’ .
  - To see the list of files in your folder use: **ls**
- Pathnames are specified using directory names separated by a “/”.
  - E.g. **/home/moussam/lab1/test.c**

# Examples of Linux Commands

- *pwd* print working directory
- *cd* change to another directory
- *mkdir* create a new subdirectory
- *rmdir* remove the specified directory
- *mv* move a file to a new name or new directory
- *rm* remove the specified file (careful!!)
- *ls* list the files in the current directory
- *cp* copy files
- *man* Linux manual for help
- *gcc* compiles a C program

## Examples of Linux Commands on the Cygwin Terminal Window:



The screenshot shows a Windows-style terminal window titled 'E /' containing a session of Linux commands. The session starts with creating a directory 'lab1', changing into it, printing the current working directory, changing back to the home directory, listing the contents of the root directory, and finally ending with a prompt.

```
moussam@ICT3200707 ~
$ mkdir lab1

moussam@ICT3200707 ~
$ cd lab1

moussam@ICT3200707 ~/lab1
$ pwd
/cygdrive/c/Users/moussam/lab1

moussam@ICT3200707 ~/lab1
$ cd ..

moussam@ICT3200707 ~
$ cd /
moussam@ICT3200707 /
$ ls
bin      Cygwin.bat  Cygwin-Terminal.ico  etc      lib      sbin    usr
cygdrive  Cygwin.ico  dev                  home    proc    tmp     var

moussam@ICT3200707 /
$
```

# **Writing and Running a C Program**

## How to Write and Run A C-Program

- To develop and run a C program, you may use an Integrated Development Environment (IDE) tool.
- Or just edit your code in a text-editor. Then, compile and run it from a command line in a terminal window.
  - **This is a preferred method in ENSF 337**, as it helps you to understand some of the details of compilation and execution of a C program.
  - Here are the steps to follow:
    - Use a Text Editor like Notepad++ to edit your program
    - Save it with the **.c** extension. For example: myFirst\_c\_program.c
    - Compile it with the gcc command.  
`gcc -Wall myFirst_c_program.c`
    - Run the executable file which its name by default is **a.exe**  
`./a.exe`
  - **Note:** More details will be discussed in the lab 1 assignment.
- Lets now take a quick look at a simple C program that reads two integer numbers and displays the sum of these numbers.

# **More on Similarities/Differences Between C, C++, and Processing Languages**

## Similarities Amongst Three Languages

- Many of the basic constructs amongst C, C++ and Processing are similar:
  - Rules for naming variables
  - Selection structures: if ... else statements
  - Switch statements
  - Repetition structures: while loop, do loop, for loop
  - Jump statements: break, continue
  - General format of function: rules for the function name, arguments, and return value(s).
  - Most of the operators:
    - Arithmetic operators: addition, subtraction, multiplication, modulus, division: +, -, \*, %, /
    - Relational operators: <, >, <=, >=, ==, etc.
    - Logical operators: &&, ||, !
    - Assignment and updating operators: =, +=, -=, \*=, /=, %=
    - Increment and decrement operators: ++, -- (prefix and post-fix)

## Differences Amongst Three languages

- Standard Input output
- Minor differences among some data type
- C and C++ doesn't support data type **byte**
- C is a procedural programming language and doesn't support **class** type.
- C doesn't support any library class such as: String (in processing), string (in C++), vector (in C++), etc.
- C uses different syntax for dynamic allocation of memory, doesn't support garbage collection approach like processing, and uses different syntax for memory de-allocation.
- The three languages use different methods and libraries for File I/O
- The three languages use more or less different syntax to **import** or **include** files necessary to use library functions or data type.
- While C and C++ support pointer, Processing doesn't.
- There are more difference that we will discuss as we go.

# **Anatomy of a Simple C Program**

## Problem Statement

- Lets write another simple program that interacts with the user: Asks the user to enter the length in feet and displays the value of the length in inches.
- Here is a sample run of the program on the screen, when user enters 10 for the length:

```
Enter the length in feet: 10
10 feet equals 120 inches
```

- Now lets review the organization and the anatomy of the program on the next slide

## Anatomy of a Simple Computer Program

```
// File: convertFeetToInch.cc } comments
// My first C program

#include <stdio.h> } Including standard library info

int main (void)
{
    int length_ft; } Variables declaration
    int length_inches;

    const int conversion_factor = 12; } Constant declaration

    printf("Enter the length in feet:"); } Prompt for input

    scanf("%d", &length_ft); } Read user input from keyboard

    length_inches = length_ft * conversion_factor; } Arithmetic expression

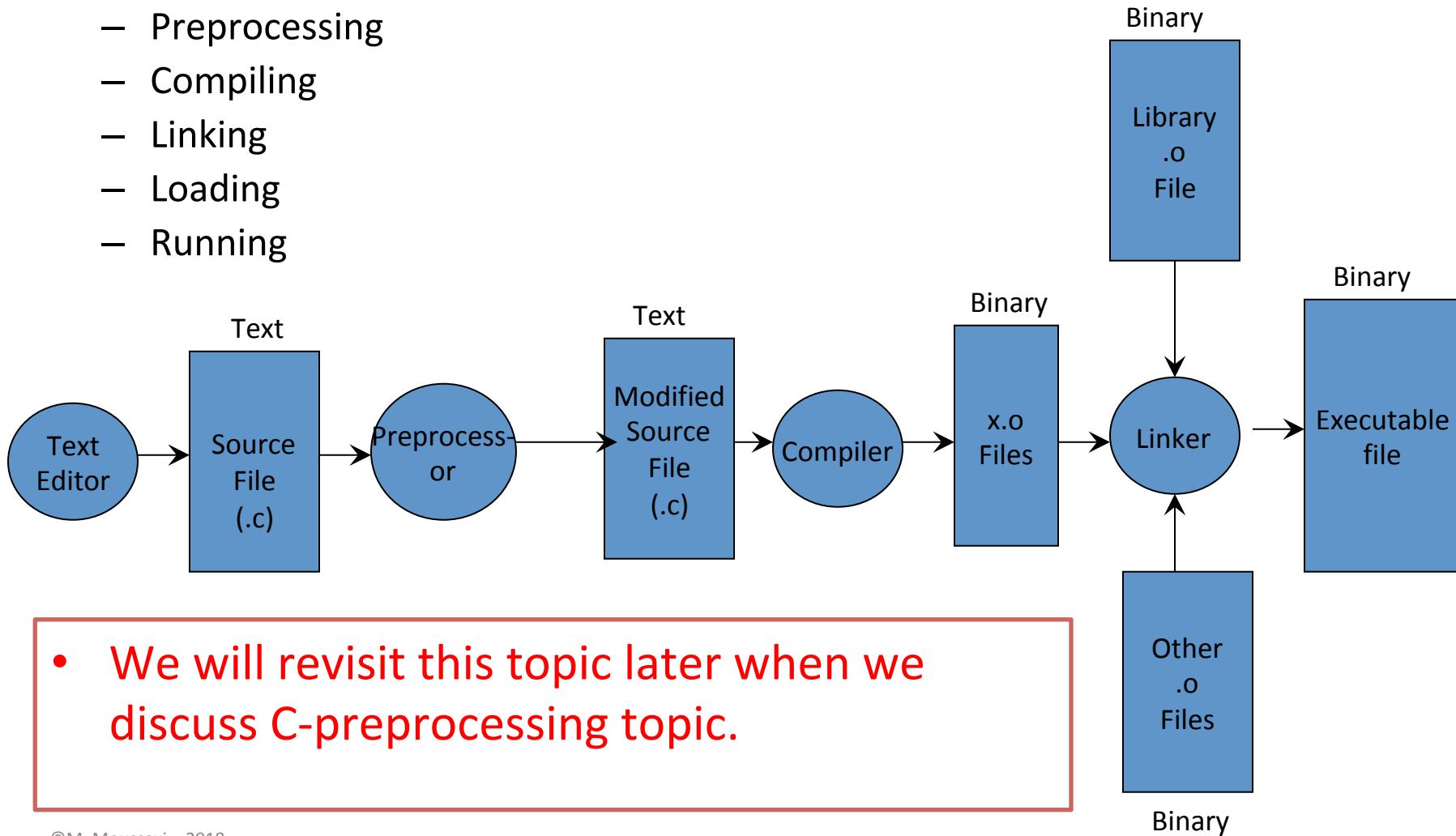
    printf( "%d feet equals: %d.\n", length_ft, length_inches);

    return 0;
}
```

# **Quick Look at the Development Process**

# Program Development

- The process of developing a program and running an executable file consists of several operations:
  - Editing
  - Preprocessing
  - Compiling
  - Linking
  - Loading
  - Running



# Do It Yourself!



- Write a C program that prompts the user to enter two integer numbers and displays the sum of two integers.

## Using Notepad++ - Text Editor

- Notepad++ is the recommended text editor for work in the ENSF337 labs. Note that Notepad++ is a free and powerful editor, which is designed to be used for writing code.

## Solution: Program Edited in Notepad++

The screenshot shows a Notepad++ window displaying a C program. The file is named "myFirst\_C\_Program.c". The code includes comments explaining various parts of the program, such as the inclusion of stdio.h and the use of printf and scanf functions.

```
1 #include <stdio.h> } Preprocessor includes the content of stdio.h
2
3 int main(void)
4 {
5     int a, b; } Local variable declaration
6     printf("please enter an integer number: ");
7     scanf("%d", &a);
8     printf("The value of %d is stored in a.", a);
9
10    printf("\nplease enter another integer number: ");
11    scanf("%d", &b);
12    printf("The value of %d is stored in b.\n", b);
13
14    printf("Sum of two numbers %d and %d is %d", a, b, a+b);
15    return 0;
16 }
```

Annotations in red:

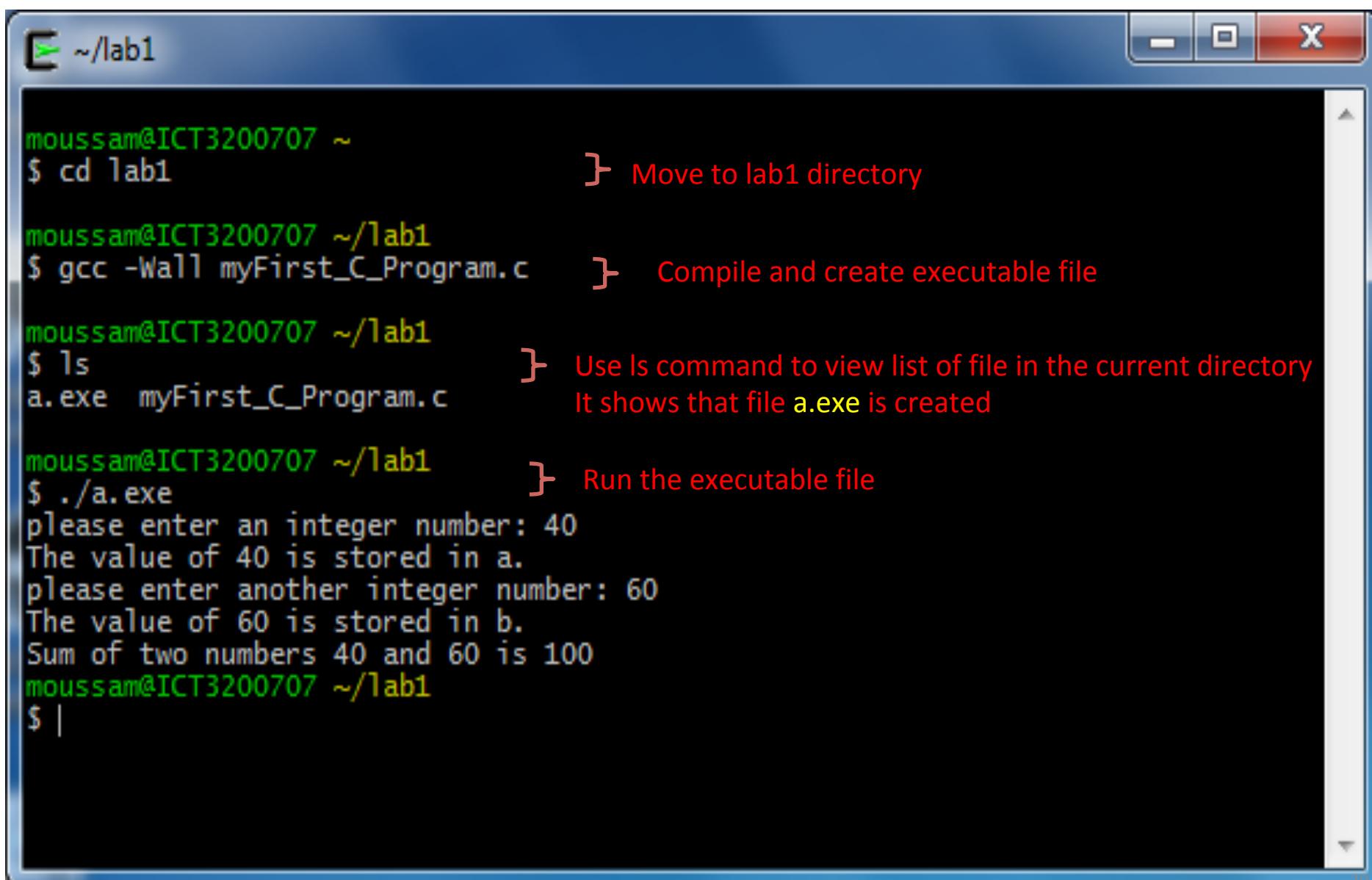
- A brace on the line "#include <stdio.h>" is followed by the text "Preprocessor includes the content of stdio.h".
- A brace on the line "int a, b;" is followed by the text "Local variable declaration".
- A brace on the line "scanf("%d", &a);" is followed by the text "Call to the library function".

Notepad++ status bar:

length : 365 lines Ln : 7 Col : 21 Sel : 0 | 0 Dos\Windows ANSI as UTF-8 INS

## Compiled and Executed in Cygwin

- Assume the source file that was edited in Notepad++, was saved as myFirstProgram.c in the directory called lab1



The screenshot shows a Cygwin terminal window with a blue title bar and a black body. The title bar displays the path `~/lab1`. The terminal window contains the following text:

```
moussam@ICT3200707 ~
$ cd lab1
moussam@ICT3200707 ~/lab1
$ gcc -Wall myFirst_C_Program.c
moussam@ICT3200707 ~/lab1
$ ls
a.exe  myFirst_C_Program.c
moussam@ICT3200707 ~/lab1
$ ./a.exe
please enter an integer number: 40
The value of 40 is stored in a.
please enter another integer number: 60
The value of 60 is stored in b.
Sum of two numbers 40 and 60 is 100
moussam@ICT3200707 ~/lab1
$ |
```

Red annotations are present on the right side of the terminal output:

- `}] Move to lab1 directory` points to the command `$ cd lab1`.
- `}] Compile and create executable file` points to the command `$ gcc -Wall myFirst_C_Program.c`.
- `}] Use ls command to view list of file in the current directory  
It shows that file a.exe is created` points to the command `$ ls`.
- `}] Run the executable file` points to the command `$ ./a.exe`.

# **Saving Your Files When Working in School Labs**

## Saving accessing you files in our computer labs:

- With a file system you can have hierarchy of directories.
  - Terms folder and directory are the same.
- Microsoft Windows uses drive letters such as C, D, E, ... to each of the file systems accessible on a computer.
- You should not save your files to the C: drive, as you may not use the same computer next time and files on this drive will not be permanently kept.
- Your best choice is to use the H: drive or save your work on a USB memory stick.
  - When you plug a USB memory stick " into a computer, it will most likely appear as E: drive (or it may get some other letter).
  - When using USB always eject the drive safely and take it with you when you leave the lab!.
- Each student has a H: drive. You can access your files from any of the computer labs at the School of Engineering on this drive.

## How to Navigate between Directories/Folder in Cygwin

- Cygwin uses a special directory called cygdrive to provide access to the Windows drives. The following table shows the equivalent Cygwin name for each C:, E:, ad H: drive under the Windows.

Drive	Cygwin equivalent
C:	/cygdrive/c
E:	/cygdrive/e
H:	/cygdrive/c

- For example if you currently working in a directory in the Window's C: drive and want to move to your Window's H: drive, you can enter the following command on the Cygwin command line:

```
cd /cygdrive/h
```

- Now if you want to again back to the C: drive, you need to enter the command:

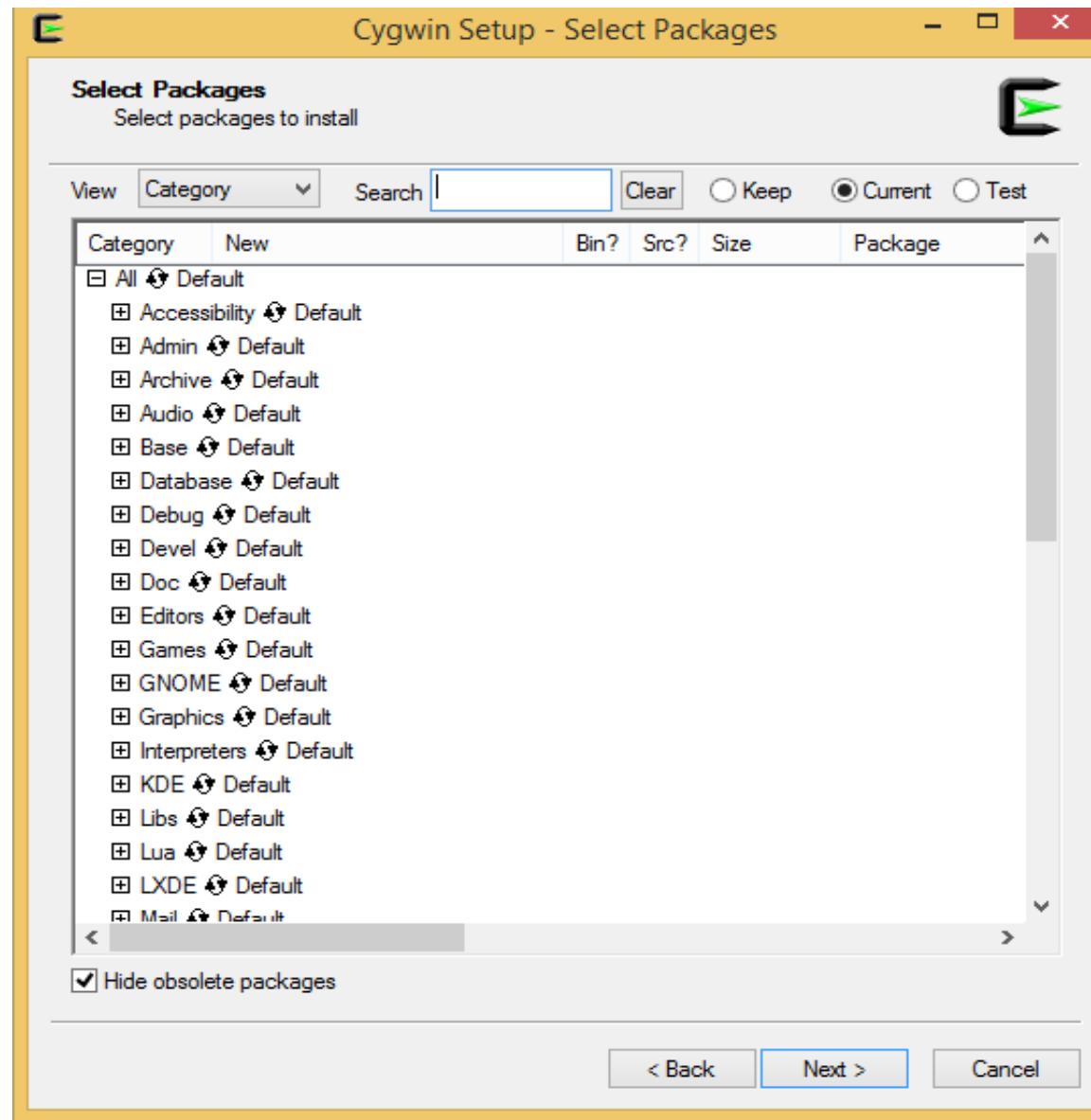
```
cd /cygdrive/c
```

# How to Install Cygwin

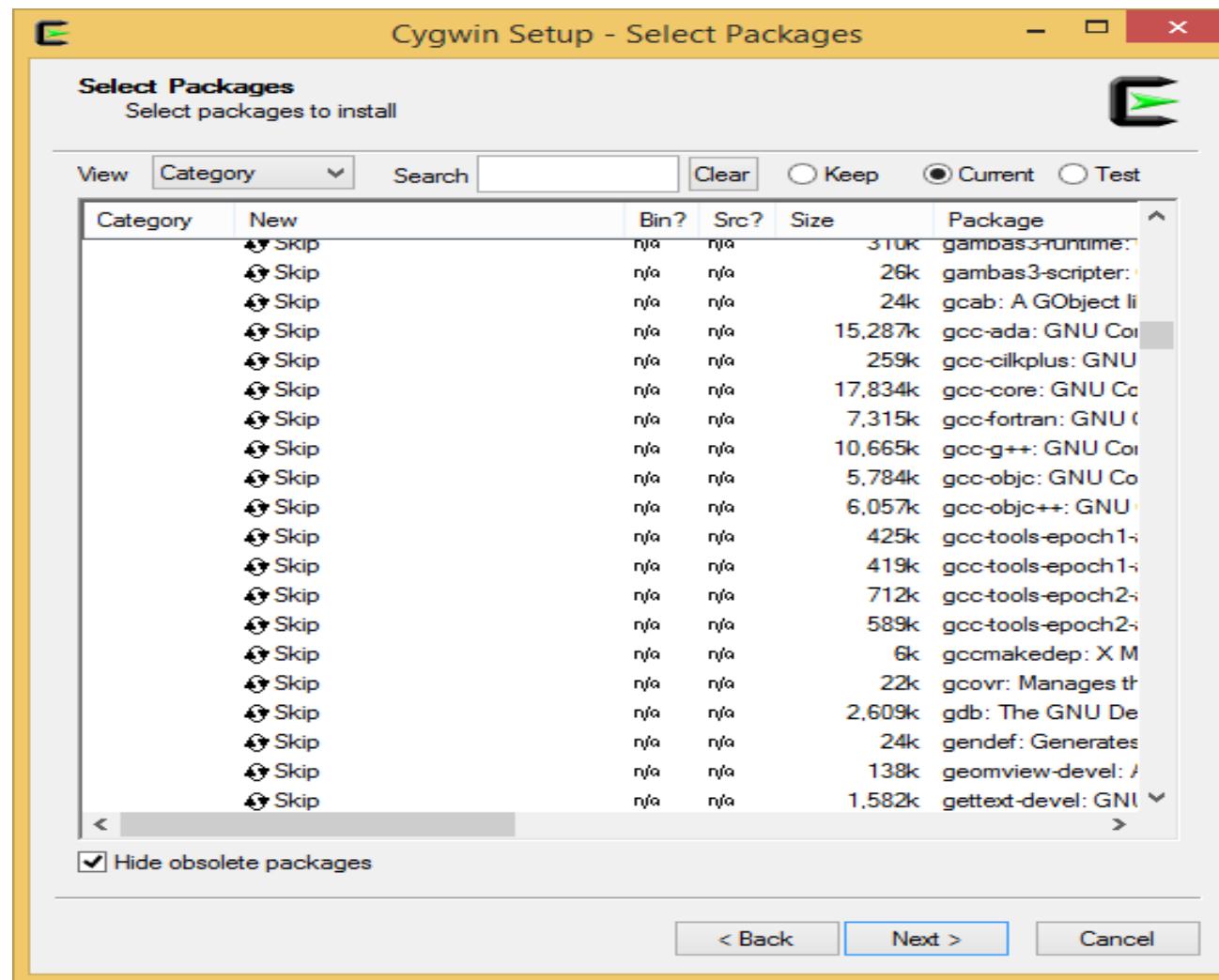
## Installing Cygwin:

- To install Cygwin, or updating installation visit:  
[https://www.cygwin.com/install.html.](https://www.cygwin.com/install.html)
- In this page download the setup-x86\_64.exe and run this executable file.
- You can also use the **setup-x86\_64.exe** any time you want to update Cygwin package.
- You don't need to install all the packages, but for the purpose of ENSF 337 your need to have at least two packages:
  1. The packages in the “Base” category that is a default package.
  2. The package, `gcc-core`, `gcc-g++`, `clang` in the “Devel” category. These packages are needed for compiling C and C++ programs.

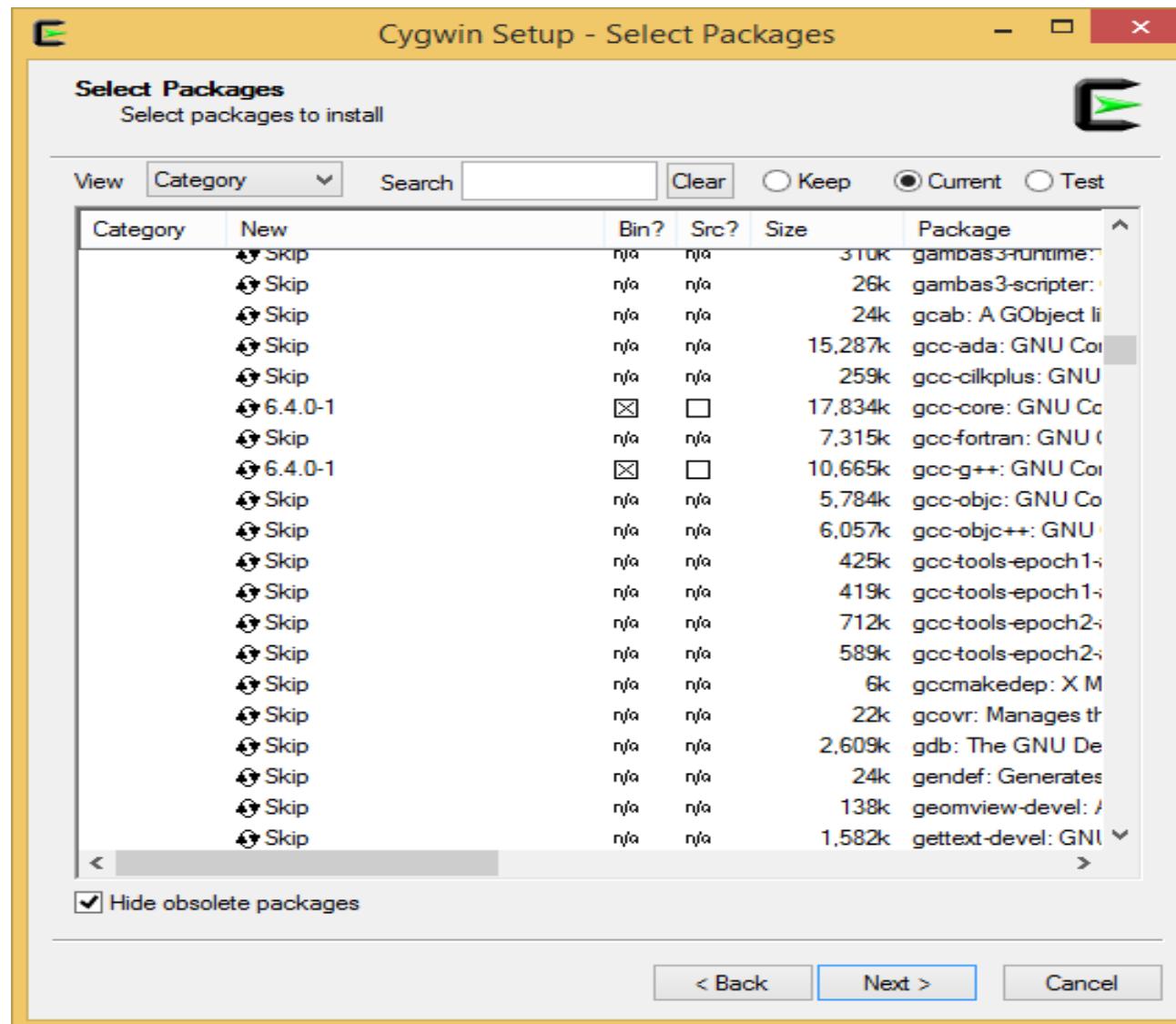
# a screenshot of the page that allows you to select packages:



If you expand the category Devel, and scroll down you will see the packages shown in the following figure:



If you select **gcc-core** and **gcc-g++** your screen will look like:



And, shows the version numbers of each package.