Computer Science 260
Lab: Introduction to Linux and Code::Blocks

Important: Linux is case sensitive. "cs" != "CS"

1) Login to the campus Windows network by opening a web browser and going to "view.truman.edu". Alternatively, you can go to the nerdery (vh2110) and use the computers there.

2) We are going to use Windows to connect to another computer system: the computer science department's Linux server named "ice". The software we will be using to connect is called `VNC`.

3) Click on the Windows start button at the lower left of the screen. Find "TightVNC" and click on that selection. Then, click on "TightVNC Viewer" to start VNC.

4) When the "New TightVNC Connection" dialog opens, click on the Remote Host: box, and type "ice.truman.edu:4". Click connect.

5) "ice.truman.edu" is the domain name of the host. The suffix ":4" specifies the display size you want to use. Ice supports various display sizes numbered from 0 to 5. If you don't like the display size that appears, close the "x11 – TightVNC Viewer" window and try again.

6) The window that appears should say "Welcome to ice". Before you log in, click on the "Start WM" menu at the lower left and select "xfce4-session". The, click in the Login Name: text box and type your Truman user name. Hit <enter>. (Don't click "Go!") Type your password and hit <enter> again.

7) If you did everything correctly, you should soon see the Linux Desktop

8) The Linux system may ask you some initial configuration questions. It will only ask these questions once. If you are asked to choose a password for a "new keyring", click "cancel". If you are asked to configure panels, choose the default.

9) **Important:** When you are finished with your Linux session, do not click the close box on the VNC window. It is very important that you properly end your session either by logging off of the Linux system when you are done.

   Go the `Applications` menu in Linux, and choose `Log Out`. When the "Log Out" menu appears, click the appropriate button. After a few seconds, you will be returned to the Windows desktop.
   If you do not log off of the Linux system when you are done, you will lose any unsaved work, and you can corrupt your Linux settings, which may make it impossible for you to login again without having the systems administrator reset your account. Simply closing the VNC window without logging out is equivalent to pulling the plug on your desktop; do not do this.

If you access Linux using VNC from a laptop, it is very important that you logout of Linux before closing your laptop. Closing the lid on most laptops puts the laptop to sleep. Before it sleeps, the laptop closes all of its network connections, including your VNC network connection. This can destroy your Linux session.

10) Use VNC to log in to ice again.

11) Once your connection is established and you see the Linux desktop, open the `Applications` menu at the top of the Linux desktop. Click on `Applications > Development > Code::Blocks IDE`. Code::Blocks is an Integrated Development Environment. That means you can edit, compile, run, and debug your programs all from within Code::Blocks.

12) Every program in Code::Blocks is referred to as a "Project". A project contains all the files that make up the program: Header files, class implementation files, and the main program.

13) Let's write our first program. It will contain the Rectangle class, along with a main program.

14) Click on the `Code::Blocks` File menu, and choose New > Project. A new window should open. This Window means the Project Wizard is running. It will help you configure `Code::Blocks` for you project.

15) Find the icon that says "Console application" and click on it. Then click "Go". A console application is one where all the input comes from the keyboard, and all the output is text.

16) Click through the Wizard until it asks you which language you want to use. Choose C++ and click Next.

17) When the wizard asks you for a Project Title, enter "Rectangle_Project" and click Next.

18) When the Wizard asks you which Compiler you want, open up the menu and choose "LLVM Clang Compiler". Make sure the checkbox next to "Create Debug configuration" is not checked. Make sure the checkbox next to "Create Release configuration" is checked. Click Finish.

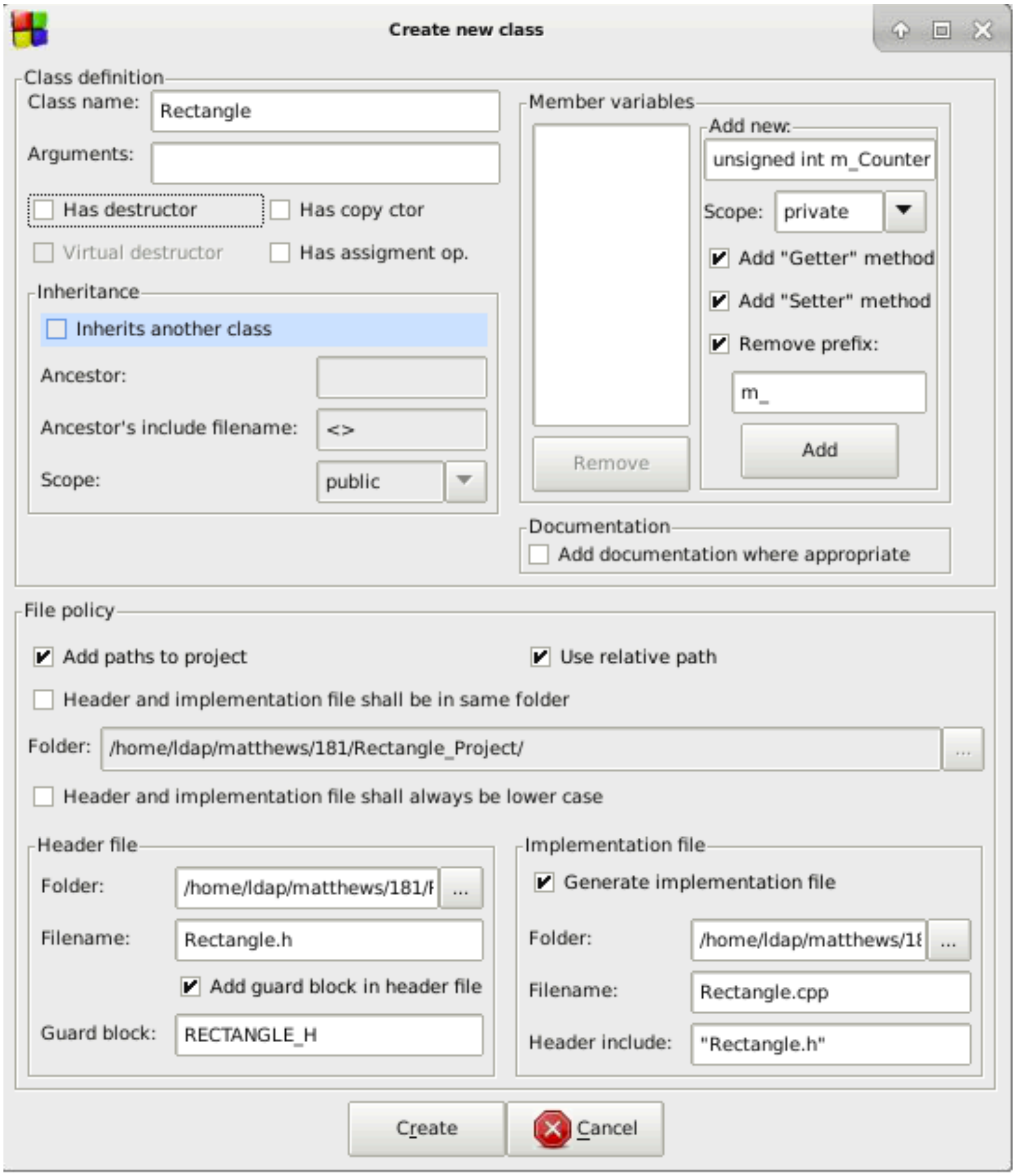
19) Now, we need to add some files that contain code to our project. Click the File menu and choose New > Class. The "Create new class" window should open.

20) The New Class dialog contains many options. We'll talk about some of these over the next few weeks.

    For now, do the following:

    a. Enter "Rectangle" for the class name.
    b. Uncheck "Has destructor" and "Virtual Destructor".

    Your Create new class window should look like this:



Class definition

Class name: Rectangle

Arguments:

☐ Has destructor ☐ Has copy ctor

☐ Virtual destructor ☐ Has assigment op.

Inheritance

☐ Inherits another class

Ancestor:

Ancestor's include filename: <>

Scope: public

Member variables

Add new: unsigned int m_Counter

Scope: private

☑ Add "Getter" method

☑ Add "Setter" method

☑ Remove prefix: m_

Documentation

☐ Add documentation where appropriate

File policy

☑ Add paths to project ☑ Use relative path

☐ Header and implementation file shall be in same folder

Folder: /home/ldap/matthews/181/Rectangle_Project/

☐ Header and implementation file shall always be lower case

Header file

Folder: /home/ldap/matthews/181/F

Filename: Rectangle.h

☑ Add guard block in header file

Guard block: RECTANGLE_H

Implementation file

☑ Generate implementation file

Folder: /home/ldap/matthews/18

Filename: Rectangle.cpp

Header include: "Rectangle.h"

21) Click Create. The Wizard will create the declaration (.h) and implementation (.cpp) files for the class. When asked, add the files to the project.

22) The editor panel in the center of the screen should now have two new tabs: “Rectangle.h” and “Rectangle.cpp”. Click on these tabs to switch back and forth between the files.

23) Edit “Rectangle.h” so that is looks like this:

```
#ifndef RECTANGLE_H
#define RECTANGLE_H

class Rectangle
{
   private:
         double length;
         double width;

   public:
         Rectangle ();

         void setLength (double l)
         {
               length = l;
         }

         void setWidth (double w)
               {width = w;}

         double getLength() const;
         double getWidth() const;
         double getArea() const;
}

endif;
```

24) Edit the implementation file so that is looks like this:

```
#include "Rectangle.h"

/* Construct a new rectangle */
 Rectangle::Rectangle() :
        length(0.0), width(0.0)
{}

/* Get length of rectangle */
double Rectangle::getLength() const
{
    return length;
}

/* Get width of rectangle */
double Rectangle::getWidth() const
{
    return width;
}

/* Get area of rectangle */
double Rectangle::getArea() const
{
    return length * width;
}
```

25) Once you have created both files required for the class, you can check your syntax. Make sure the implementation file (Rectangle.cpp) is visible in the edit pane by clicking on its tab. Then, click on the Build menu and choose “Compile current file”.

If you did everything correctly, the Build Messages pane at the bottom of the window should say something like "Build finished: 0 error(s), 0 warning(s)". If it did, congratulations! You have just written your first C++ class.

If there are any errors, correct them and recompile.

26) Whenever you create a new project, you will get a main program ("A Hello World.") for free. We'll now edit the main program to make it do what we want.

    In the Management pane on the left, click the small triangle next to the "Rectangle_Project". When it opens you should see two additional folders: "Sources" and "Headers". Open both of these, and then open the folders that they contain. You should see "Rectangle.cpp", "main.cpp" and "Rectangle.h".

    Double-click on "main.cpp" to open it for editing. Enter the following code in the window.

```
#include <iostream>

#include "Rectangle.h"

using namespace std;

int main()
{
    double length, width;

    Rectangle r;
    cout << "Enter rectangle length: ";
    cin >> length;

    cout << "Enter rectangle width: ";
    cin >> width;

    r.setLength (length);
    r.setWidth (width);

    cout << "The rectangle width is " << r.getWidth() << endl;
    cout << "The rectangle length is " << r.getLength() << endl;
    cout << "The rectangle area is " << r.getArea() << endl;
}
```

27) Now that we have a complete program (main program and class) we are ready to build the entire program. Building compiles all the source files and then combines them into an executable. Do this by clicking on the "Build" menu and choosing "Build".

    If you program builds correctly, you will something like "0 errors(s), 0 warning(s)" in the Build messages pane. If you didn't, correct your errors and try again.

28) Once the program correctly builds, you are ready to execute it. Go to the "Build" menu and choose "Run". A new window will open. Interact with the program in this window.

29) Once your done running your program, exit Code::Blocks by choosing “Quit” from the file menu.

30) Log out of the Linux system by choosing "Log Out" from the Linux applications menu. Remember that it is very important that you suspend or exit VNC correctly. DO NOT JUST LOG OUT OF WINDOWS OR CLOSE THE VNC WINDOW.

    Once you have logged out of the ice, don't forget to log out of Windows. You do not have to submit anything for this lab.

Getting VNC

You can get a copy of the VNC software for your computer if you wish. This avoids the need to go to view.truman.edu. Instead, you can connect to ice directly for your desktop or laptop system.

VNC is available for many different operating systems. Go to “realvnc.com” to obtain the software.