LAB 10

OOP (Object Oriented Programming)

- The concept of OOP (Object Oriented Programming) is the process of creating objects that contains both data and functions

Why OOP? What are the advantages?

- Faster and easier to execute
- Provides a clear structure for programs
- Makes code easier to maintain, modify and debug
- Create reusable applications with less code and shorter development time

OOP

Two main important aspects of OOP are: **classes** and **objects**

Template for objects

Instance of a class

Class

car

Object

Volvo
Audi
Toyota

Classes

- It's a user defined data type
- It consists of **attributes** and **methods**
- Access specifiers for class members: **public** and **private**
- By default class members are private unless otherwise specified
- *"Private"* means that class members cannot be used outside the class
- *"Public"* means that class members can be used outside of the class

How do we create classes? (.h files)

- Using header files or files that have a (.h) extension
- Header files allow us to put declarations in one location and then import them wherever we need them
- Notice that header files would only include declarations
- Header files contain header guards to ensure that definitions are not included multiple times
- If the header file has the name “square.h”:

```
#ifndef SQUARE_H
#define SQUARE_H
 Class square {

       };

    #endif
```

.cpp files

- Functions declared in header files should be defined in .cpp file
- The related header should be included

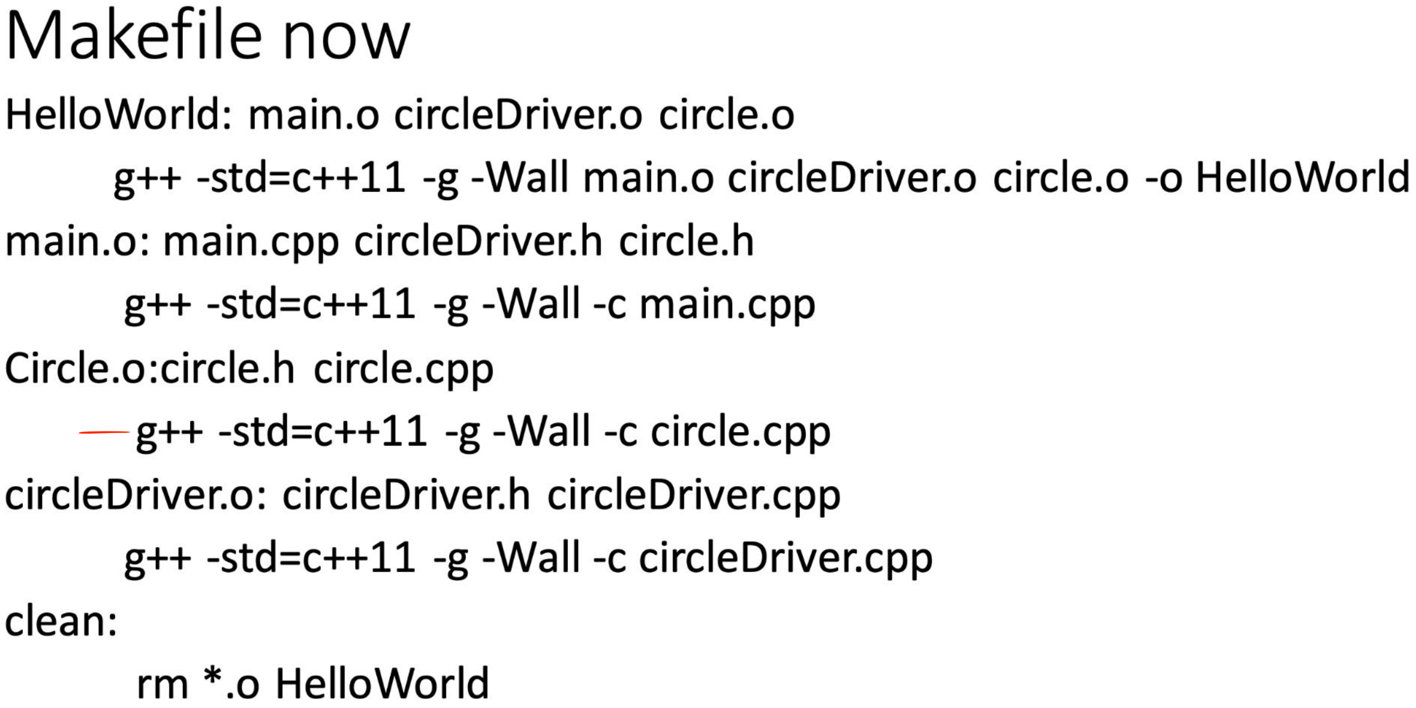
```
#include “square.h”
double square::area( )
{
//body
//return <double_variable>;
}
```

Remember to Update Makefile!!

https://wiki.ittc.ku.edu/ittc_wiki/index.php?title=EECS268:Makefiles

- Go to the class wiki page and look under the section: Compiling with Class Files to know how you'll have to change your makefile as you're adding in header files and additional .cpp files

Updated Makefile



Makefile now

```
HelloWorld: main.o circleDriver.o circle.o
        g++ -std=c++11 -g -Wall main.o circleDriver.o circle.o -o HelloWorld
main.o: main.cpp circleDriver.h circle.h
        g++ -std=c++11 -g -Wall -c main.cpp
Circle.o:circle.h circle.cpp
        g++ -std=c++11 -g -Wall -c circle.cpp
circleDriver.o: circleDriver.h circleDriver.cpp
        g++ -std=c++11 -g -Wall -c circleDriver.cpp
clean:
        rm *.o HelloWorld
```

Exercise

Two classes: Circle class , CircleDriver class