lab_intro : Ineluctable Introduction
Week #1 – August 29-31, 2018

Welcome to your first CS 225 Lab!

Course Website: https://courses.engr.illinois.edu/cs225/labs

Overview

Each week, you'll join a small group of your peers and several members of the course staff for a small, hands-on programming experience.

Your TA (a graduate student in Computer Science) is your primary point of contact for your CS 225 course needs. They can help you with concerns about your course grade, logistics, and other grade-related matters:

Your TA: _____

Lab CAs (fellow undergraduates who have taken CS 225 in the past and coming back to the course to help you succeed).

Your CA(s): _____

Grading in Lab Sections

Each lab is worth a total of 10 points towards your course grade and will be broken down into two components:

- **4**/10 points for attendance *(earned for a complete worksheet)*
- **6**/10 points for based on the grade of your lab assignment

There are no attendance points for lab_intro. The lab assignment is worth 10 points.

Any points earned above 100 points are earned as extra credit in CS 225! This means by attending all of the labs and fully completing all of the assignments, you will earn 140 points -- **+40 points of extra credit**.

Due to the large amount of extra credit, attendance credit can never be "made up" or "excused". **If you miss or are late for a lab**, you miss out on those +4 points – but you can miss 10 labs and still get a full 100/100 for your lab score!

Classes in C++

In lecture, you created your first C++ class and included several *member variables* as part of that class.

Exercise 1: Write the C++ code for a `Square` class that contains a `width` public member variable and a member function `getArea`.

Square.h		Square.cpp	
1		1	
2		2	
3		3	
4		4	
5		5	
6		6	
7		7	
8		8	
9		9	
10		10	
11		11	
12		12	
13		13	
14		14	
15		15	
16		16	
17		17	
18		18	

Identify which line in the code you created contain the follow elements required in a C++ class:

- Pragma Directive:
  In the file _____ on line ____.
- The class definition for `Square`:
  In the file _____ on lines ____ through ____.
- The definition for the function `Square::getArea()`:
  In the file _____ on lines ____ through ____.
- The declaration for the function `Square::getArea()`:
  In the file _____ on line ____.
- The code denoting that both members are public:
  In the file _____ on line ____.

Class Constructor

A constructor is a nameless special member function that is **always** the first function invoked when the object is created. The constructor is the standard way to set the initial value of a object and perform any initialization logic.

A constructor (often abbreviated ctor) may have zero or more function parameters and the constructors with the matching arguments are called when the object is created:

main.cpp

```
1  #include "Square.h"
2
3  int main() {
4    Square c;  // Zero arguments ➔ zero parameter ctor
5    Square c(2); // One argument ➔ one param ctor
6    // ...
7  }
```

The zero parameter constructor has a special name and is known as the **default constructor**.

Exercise 2: Declare and define the two constructors used by the code in main.cpp above. Have the default constructor initialize the square to be a unit square (width=1).

Square.h

```
1  #pragma once
2
3  class Square {
4    public:
5
6
7
8
9
10
11
12   // ...
…  };
26
27
28
```

Square.cpp

```
1  #include "Square.h"
2
3
4
5
6
7
8
9
10
11
12
13
14
15 // ...
…
```

Namespaces

In C++, large projects and libraries are organized into namespaces to avoid name collisions (there's probably hundreds of `Square` classes in the world)!

Namespaces are defined by placing the class and member function definitions into a namespace scope. For example:

Square.h

```
1  #pragma once
2
3  namespace cs225 {
4    class Square {
5      // ...
…    }
26 };
27
28
29
```

Square.cpp

```
1  #include "Square.h"
2
3  namespace cs225 {
4    double Square::getArea(){
5      return width_ * width_;
     }
6  }
7
8  // ...
…
```

When you need to use a class in a namespace, you have two options:

- Refer to the namespace and the class, separated with a scope resolution operator (::), such as: `cs225::Square::getArea`
- Import the class from the namespace so a scope resolution operator is no longer needed: `using cs225::Square;`

In the programming part of this lab, you will:

- Learn about the HSL color space
- Create a class to contain an HSL pixel that's part of the cs225 namespace
- Create several constructors for your HSLAPixel class
- Use the HSLAPixel class, along with a provided PNG class, to modify an image in unique and fun ways.

As your TA and CAs, we're here to help with your programming for the rest of this lab section! ☺