1

C++ Structs and Enumerations

CSC 340 - Credit to Hui Yang

February 17, 2016

2

Overview

- **Structs**
- Enumerations

3

Structs

- Review Basic C++ lecture slides
- Quick refresher:
  - Data members default to public
  - Syntax:

```
struct Person
{
    string first_name;
    string last_name;
    string ssn;
    int age;
};
```

Overview

- Structs
- **Enumerations**

4

Enumerations

- Often used to represent related sets of constants
- Two forms:
  - (C++11) enum class: Strongly typed and recommended
    ```
    enum class Color { red, blue, green }
    ```
  - enum: implicitly converts each value to an integer
    ```
    enum Color { red, blue, green }
    int color = green; // color gets value 2
    ```


5

Note that the first is specific to C++11, to this point in lectures we have only been looking at C++98
enums are zero indexed, altho we can specify index in declaration if we want:
enum Color { red = 2, blue, green }
blue is 3, green is 4

Enum Class

- Strongly typed
  - Example declaration
    ```
    enum class Color { red, blue, green };
    ```
  - Correct Code:
    ```
    Color color_one = Color::red;
    ```
  - Illegal Code:
    ```
    Color x = red; // Not specific
    int i = Color::red // Strongly typed
    ```


6

Note we are talking about enum *classes* here (some of this code is legal with enums…)

Enum class

- By default, enum class only has assignment, initialization, and comparison.
- We can overload meaningful operators (more on this when we discuss overloading…)

7

Conclusion

- Introduction to C++ user-defined types

8