

Lecture 06

Classes Vs Structs

Abstract Data Types (ADTs)

- Programmer-Defined data types that specify
 - legal values that can be stored
 - operations that can be done on the values
- The user of an abstract data type (ADT) does not need to know any implementation details (*e.g.*, how the data is stored or how the operations on it are carried out)

Type in C++

- Mechanism for user defined types are
 - Structures
 - Classes
- Built-in types are like int, float and double
- User defined type can be
 - Student in student management system
 - Circle in a drawing software

Class

- Class is a tool to realize objects
- Class is a tool for defining a new type

Revision- C Structures

```
struct account {  
    int account_number;  
    char *first_name;  
    char *last_name;  
    double balance;  
};
```

- **struct** account s;
- can be accessed like s.account_number

C++ Structures with Functions

```
struct SomeStruct {  
    int x, y, z;  
  
    void someFunction() {  
        x++;  
        y++;  
        x++;  
    }  
}  
  
struct SomeStruct s;  
s.someFunction();
```

Structure Vs Class

```
struct SomeStruct {  
    int x, y, z;  
  
    void someFunction() {  
        x++;  
        y++;  
        x++;  
    }  
}  
  
struct SomeStruct s;  
s.someFunction();
```

```
class SomeClass {  
    int x, y, z;  
  
    void someFunction() {  
        x++;  
        y++;  
        x++;  
    }  
}  
  
SomeClass s;  
s.someFunction();
```

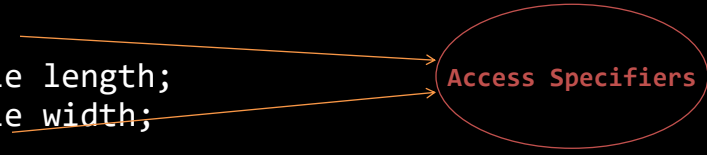
Introduction to Classes

- A class is a programmer-defined data type used to define objects
- A class consists of a set of variables called member variables and a set of functions called member functions.
- A class definition begins with the keyword *class* and a name.

Defining a New User Defined Type

```
class ClassName
{
    ...
    DataType MemberVariable;
    ReturnType MemberFunction();
    ...
};
```

```
class Rectangle
{
    private:
        double length;
        double width;
    public:
        void setDimensions(double l, double w)
        {
            length = l;
            width = w;
        }
        double getArea()
        {
            return length*width;
        }
};
```



Access Specifiers

- Used to control access to members of the class.
- Each member is declared to be either:
 - **public**: A can be accessed by functions outside of the class
 - **private**: can only be called by or accessed by functions that are members of the class
 - **protected**: can only be called by or accessed by functions that are members of the derived class

Access Specifiers (Contd.)

- Can be listed in any order in a class
- Can appear multiple times in a class
- If not specified, the default is **private!!!**

Types of Member Functions

- **Accessor, get, getter function:** uses but does not modify a member variable
e.g.; `getArea()`
- **Mutator, set, setter function:** modifies a member variable
e.g.; `setDimensions()`

Accessing members

- Members of an object can be accessed using
 - dot operator (.) to access via the variable name
 - arrow operator (->) to access via a pointer to an object
- Member variables and member functions are accessed in a similar fashion

Introduction to Objects

- An object is an instance of a class
- Defined just like other variables
`Rectangle r1, r2;`
- Can access members using dot operator
`r1.setDimensions(5,6);`
`cout << r1.getArea();`

Default access specifiers

- When no access specifier is mentioned then by default the member is considered private member

Example

```
class Rectangle
{
    double Length;
    double Width;
};

class Rectangle
{
    private:
    double Length;
    double Width;
};
```

Classes Vs Structs

- When to use a struct
 - Use a struct for things that are mostly about the data
- When to use a class
 - Use a class for things where the behavior is the most important part
 - Prefer classes when dealing with encapsulation/polymorphism

Conventions and Suggestions

Conventions:

- Member variables are usually `private`
- Member functions are usually `public`
- Use 'get' in the name of accessor functions, 'set' in the name of mutator functions

Your Turn

- Create a class named `Employee`. Declare some of its attributes and member functions.



Quiz coming next week