Lecture 6

CS 244 - Fall 2020

Review from Lecture 5

- Intro to Classes + Objects
- The ‘string’ class

Classes, Objects, Member Functions

Remember Data Types?

- They describe different types of variables we can use in our program
- `int`
- `float`
- `double`
- `char`
- `etc...`

Classes

- **Classes are essentially *user defined data types***
- Unlike traditional data types, classes can have **member functions and data members**
- Classes are the building blocks of Object Oriented Programming

```
Car myCar;
```

Classes

built in data type declaration


```
int myVariable;
```

Declaration of class “Car”

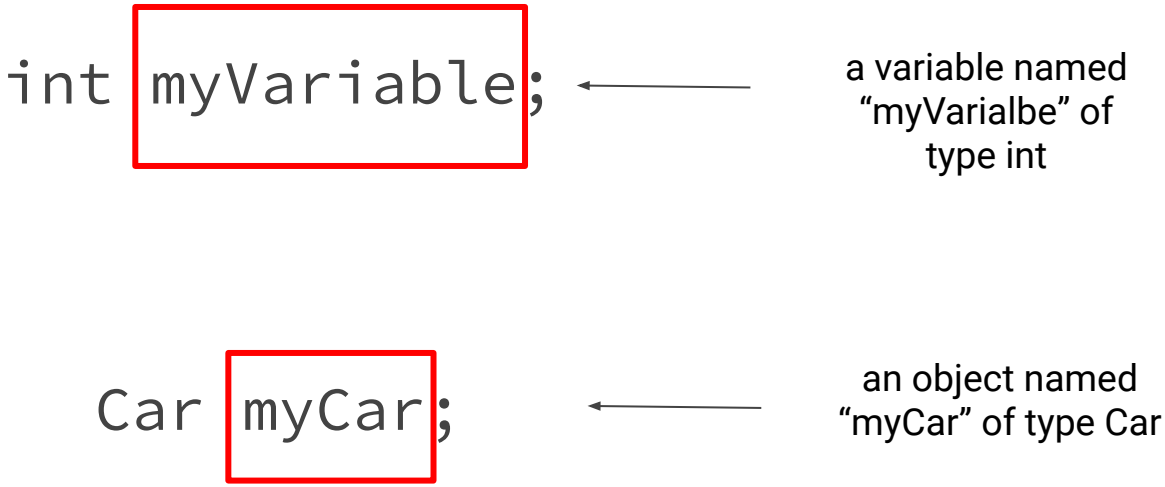

```
Car myCar;
```

Objects

- **An object is an instantiation of a Class**
- You can think of them as “special” variables, and it’s type would be it’s class

```
Car myCar;
```

Objects

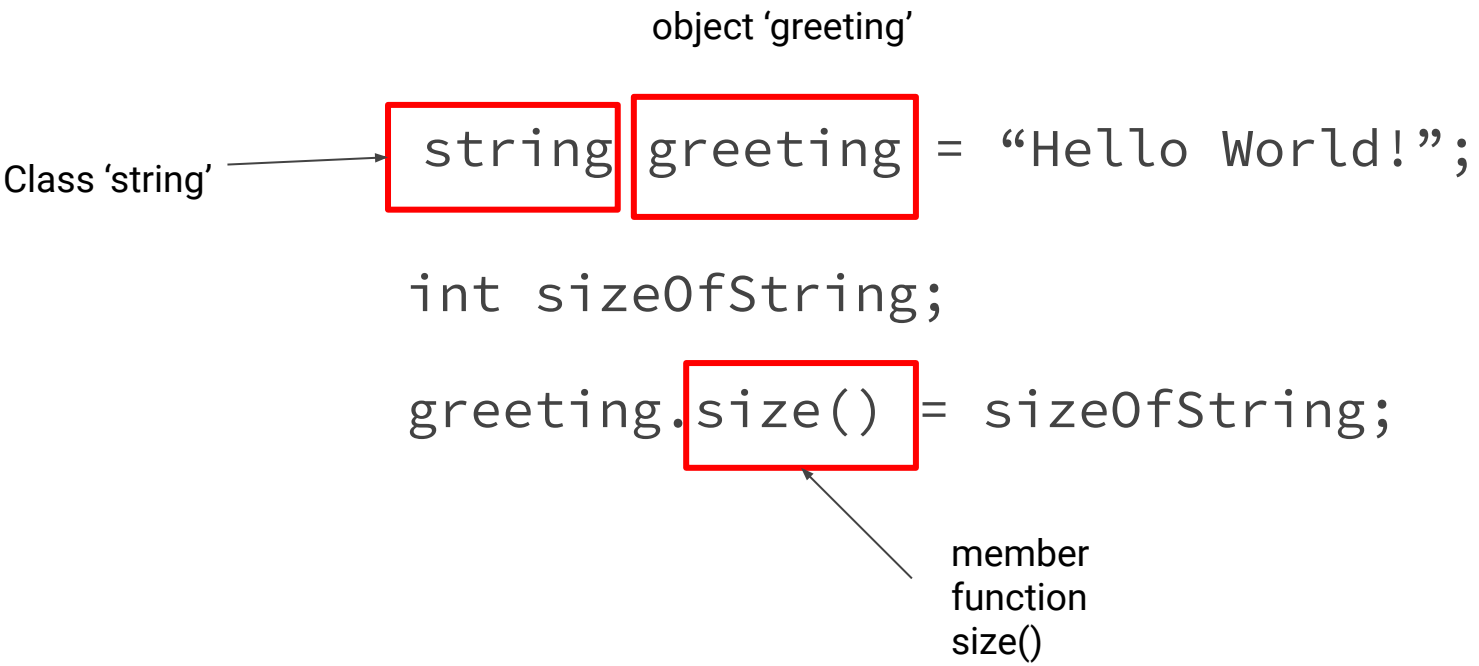

int myVariable;
a variable named “myVarialbe” of type int
Car myCar;
an object named “myCar” of type Car

Classes

- Each class can have a number of **data members** and **member functions**
- **Data members** are variables that are associated with the class
- **Member functions** are functions that manipulate those data members
- Together these data members and member functions defines the behavior of the objects in a Class

C++ string Class

- Technically ‘string’ is a Class in C++ that behaves like a basic data type

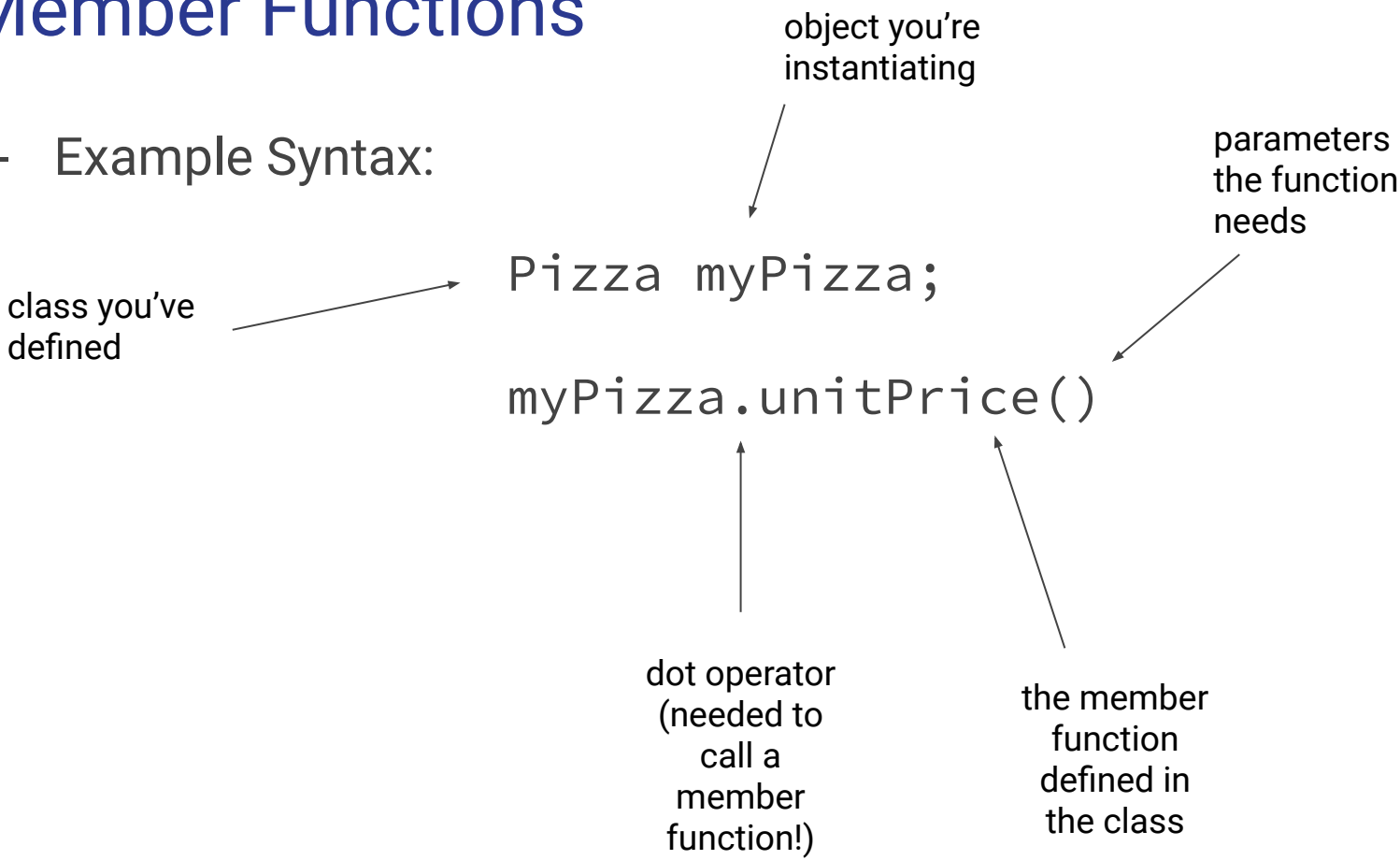


Member Functions

- Member functions of an object are the member functions of its class
- The class determines the member functions of the object
- **Calling a member function requires specifying the object containing the function**

Member Functions

- Example Syntax:



Built in Classes

- There are many built-in classes you can use within your programs, the ‘string’ class being one of the most popular
- In order to use it’s member functions, you’ll have to read it’s documentation or Class definitions to know how to properly use it

Built in Classes

- We’re going to learn one more built in class, “File Streams”

File Input and Output (File I/O)

- File Input: Reading from a file
- File Output: Writing to a File
- File I/O is performed via File Streams
    - Functions are almost exactly like the console streams cin and cout

File Streams

- Input-file streams are used to read from a file
  - **Input-file streams are of type Class ifstream**
  - behaves extremely similarly as “cin”

- Ouput-file streams are used to write to a file
  - **Ouput-file streams are of type Class ofstream**
    - behaves extremely similarly as “cout”

File Streams are Objects

- A stream is a special kind of variable called an object
- In order to use the Classes ‘ifstream’ and ‘ofstream’, you’ll need to import the library ‘fstream’

```
#include <fstream>

using namespace std;
```

Declaring Streams

- The first line declares a stream object that **streams data from a file**
- The second line declares a stream object that **streams data to a file**

```
ifstream fin;

ofstream fout;
```

Declaring Streams

- “ifstream” and “ofstream” are different Classes

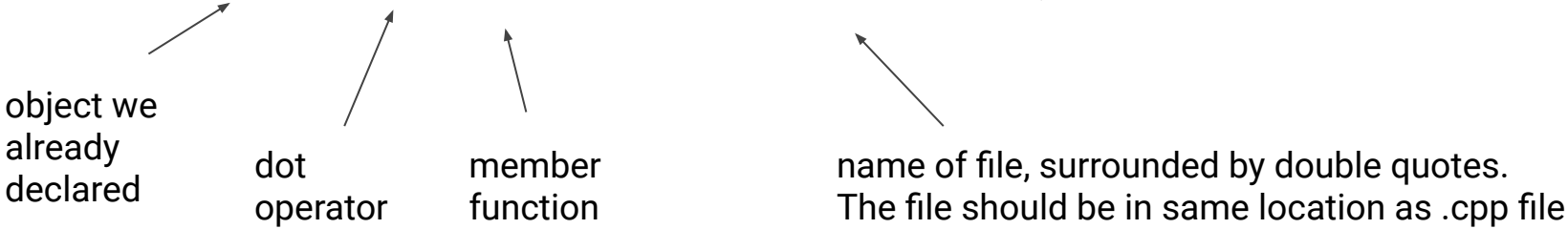
```
ifstream fin;

ofstream fout;
```

Connecting to a file

- Once a stream object is created, connect it to a file
- Connecting a stream to a file is opening the file
- Use the `open` member function of the stream object

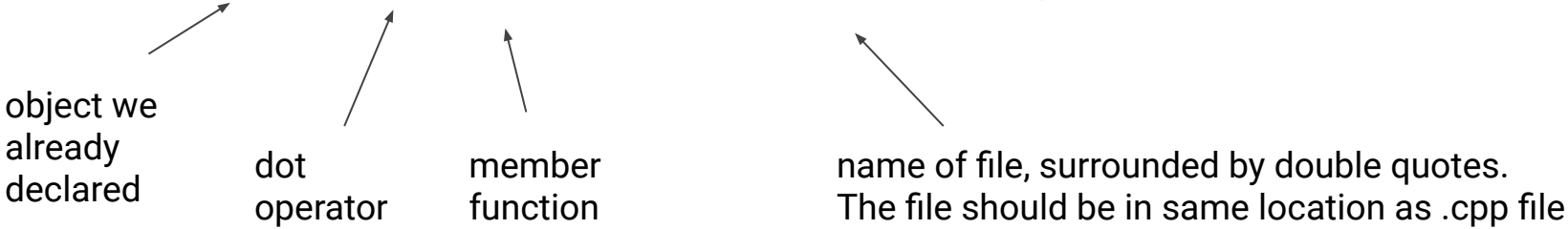
```
fin.open("input.txt");
```



Connecting to a file

- Once a stream object is created, connect it to a file
- Connecting a stream to a file is opening the file
- Use the `open` member function of the stream object

```
fin.open("input.txt");
```



Using fail and exit

- Immediately following the call to open, check that the operation was successful
- fail() is another member function within the ifstream Class
- could be multiple reasons it failed (file not existing, misspelled, etc.)

```
fin.open(“input.txt”);
if (fin.fail()){
    cout << “Failed to open file” << endl;
    exit(1);
}
```

Using getline

- You can use the function “getline()” to read each line in the file, like so:

```
 string line;
 ifstream fin;
 fin.open("input.txt");

while (getline(fin, line))
 {
     cout << line << endl;
 }
```

Closing a file

- After completing a read/write to a file, we should close it so that the operating system is notified and its resources become available again
- We use it’s member function “close()”

```
fin.close();
```

Reading for File Input/Output

http://www.cplusplus.com/doc/tutorial/files/

Lab

- Let’s write a program takes in a list of numbers and outputs it’s square numbers to another file

HW 2 will be assigned this week

Midterm will be take home (similar to a HW assignment in a couple weeks)