Member Initialization

In C++, all constructors must initialize all primitive types. A C++ constructor does not need to initialize any object members (like `string` or `vector`).

This is **exactly the opposite from Java**, where you must initialize all of the object instance variables, or they are set to `null` (an invalid object). In Java, all primitive instance variables are automatically initialized to `0` like this:

```java
public class Point
{
    private String name;
    int x, y;
    public Point() {}
}

Point p = new Point(); // x,y->0, name is null (invalid)
```

In C++, if you fail to initialize a primitive data member, then it assumes **whatever random value** was in memory; if you don't initialize an object, such as `string` or `vector`, its default constructor will **automatically** run, and it is still a valid object.

```cpp
class Point
{
public:
    Point() {}
private:
    string name;
    int x, y;
};

Point p; // x,y->random, name is valid empty string
```

Of course, if you provide in-definition initializers for your primitive data members, they **will** automatically be initialized, even if your construct does not explicitly initialize them.