Initializers for Instances

- The member initializer list specifies the values for the attributes.
- The initialization happens before the body of the constructor.
- The attributes are initialized in the order in which they are declared.

```
class Account{
public:
    Account(double b): balance(b), minBalance(25.0){ id++; }
private:
    double balance;
    const double minBalance;
    static int id;            // declaration
. . . .
};

int Account::id{};           // definition
```

Initializers for Instances

- Rules:

1. Non-`static` attributes which are declared `const` or as reference must be initialized in the member initializers list.
2. The sequence of initialization corresponds to the sequence in which the attributes were declared.
3. `static` attributes as class members are only declared in the class body.

Initialization of the Class Attributes

- Class members can be directly initialized.

```
class MyClass{
    const static int oldX = 5;                  // C++98
    int newX = 5;                               // C++11
    vector<int> myVec{1, 2, 3, 4, 5};           // C++11
};
```

- If class members are initialized in the initializer list of the constructor and in the class body, the initializer list of the constructor has higher priority.

```
struct MyClass{
   MyClass() = default;
   MyClass(int n): newX(n){}
   int newX = 5;
};
```

classMemberInitializer.cpp