Class Special Functions

Copy Constructor

- Called automatically when the object is copied.
- For a class MyClass has the signature:
  MyClass(const MyClass& other)

```
MyClass a;       // Calling default constructor.
MyClass b(a);    // Calling copy constructor.
MyClass c = a;   // Calling copy constructor.
```

Copy Assignment Operator

- Copy Assignment Operator is called automatically when the object is assigned a new value from a Lvalue
- For class MyClass has a signature:
  MyClass& operator=(const MyClass& other)
- Returns a reference to the changed object.
- Use *this from within a function of a class toga reference to the current object.

```
MyClass a;       // Calling default constructor.
MyClass b(a);    // Calling copy constructor.
MyClass c = a;   // Calling copy constructor.
a = b;           // Calling copy assignment operator.
```

Move Constructor

- Called automatically when the object is moved.
- For a class MyClass has a signature:
  MyClass(MyClass&& other)

```
MyClass a;                  // Default constructors.
MyClass b(std::move(a));    // Move constructor.
MyClass c = std::move(a);   // Move constructor.
```

Move Assignment Operator

- Called automatically when the object is assigned a new value from a Rvalue
- For class MyClass has a signature:
  MyClass& operator=(const MyClass&& other)
- Returns a reference to the changed object.

```
MyClass a;                    // Default constructors.
MyClass b(std::move(a));      // Move constructor.
MyClass c = std::move(a);     // Move constructor.
b = std::move(c);             // Move assignment operator.
```

```
class MyClass {
 public:
  MyClass() { cout << "default" << endl; }
  // Copy(&) and Move(&&) constructors
  MyClass(const MyClass& other) {
    cout << "copy" << endl;
  }
  MyClass(MyClass&& other) {
    cout << "move" << endl;
  }
  // Copy(&) and Move(&&) operators
  MyClass& operator=(const MyClass& other) {
    cout << "copy operator" << endl;
  }
  MyClass& operator=(MyClass&& other) {
    cout << "move operator" << endl;
  }
};

int main() {
  MyClass a;                  // Calls DEFAULT constructor
  MyClass b = a;              // Calls COPY constructor
  a = b;                      // Calls COPY assignment operator
  MyClass c = std::move(a); // Calls MOVE constructor
  c = std::move(b);           // Calls MOVE assignment operator
}
```

Do I need to define all of them?

- The constructors and operators will be **generated automatically**
- Six special functions for class `MyClass`:
  - `MyClass()`
  - `MyClass(const MyClass& other)`
  - `MyClass& operator=(const MyClass& other)`
  - `MyClass(MyClass&& other)`
  - `MyClass& operator=(MyClass&& other)`
  - `~MyClass()`
- **None** of them defined: **all** auto-generated
- **Any** of them defined: **none** auto-generated

Rule of all or nothing

- Try to define **none** of the special functions
- If you **must** define one of them **define all**
- Use `=default` to use default implementation

```
1 class MyClass {
2  public:
3   MyClass() = default;
4   MyClass(MyClass&& var) = default;
5   MyClass(const MyClass& var) = default;
6   MyClass& operator=(MyClass&& var) = default;
7   MyClass& operator=(const MyClass& var) = default;
8 };
```