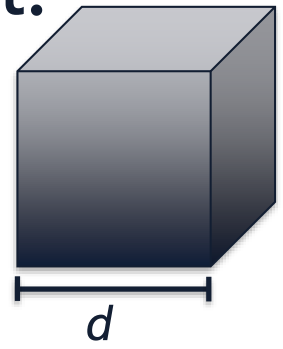
# Class Constructors

Prof. Wade Fagen-Ulmschneider

ILLINOIS

When an instance of a class is created, the **class constructor** sets up the initial state of the class.

**Object:**



**Default: Unit Cube (d=1)**

## Automatic Default Constructor

If we do not provide any custom constructors, the C++ compiler provides an **automatic default constructor** for our class for free!

The automatic default constructor will only initialize all member variables to their default values.

## cpp-std/Cube.h

```cpp
#pragma once

namespace uiuc {
  class Cube {
    public:
      double getVolume();
      double getSurfaceArea();
      void setLength(double length);

    private:
      double length_;
  };
}
```

# Custom Default Constructor

The simplest constructor we can provide is a **custom default constructor** that specifies the state of the object when the object is constructed.  We define one by creating:

- A member function with the same name of the class
- The function takes zero parameters.
- The function does not have a return type.

```
Cube::Cube()   // custom default constructor
```

## cpp-ctor/ex1/Cube.h

```cpp
#pragma once

namespace uiuc {
  class Cube {
    public:
      Cube(); // Custom default constructor

      double getVolume();
      double getSurfaceArea();
      void setLength(double length);

    private:
      double length_;
  };
}
```

## cpp-ctor/ex1/Cube.cpp

```cpp
#include "Cube.h"

namespace uiuc {
  Cube::Cube() {
    length_ = 1;
  }

  ...
```

## cpp-ctor/ex1/main.cpp

```cpp
#include "Cube.h"
#include <iostream>

int main() {
  uiuc::Cube c;
  std::cout << "Volume: " << c.getVolume() << std::endl;
  return 0;
}
```

## Custom Constructors

We can also specify custom, non-default constructors that require client code to supply arguments:

```
Cube::Cube(double length)
 // one-argument ctor specifying initial length
```

## cpp-ctor/ex2/Cube.h

```cpp
#pragma once

namespace uiuc {
  class Cube {
    public:
      Cube(); // Custom default constructor
      Cube(double length); // One argument constructor

      double getVolume();
      double getSurfaceArea();
      void setLength(double length);

    private:
      double length_;
  };
}
```

## cpp-ctor/ex2/Cube.cpp

```cpp
#include "Cube2.h"

namespace uiuc {
  Cube::Cube() {
    length_ = 1;
  }

  Cube::Cube(double length) {
    length_ = length;
  }

  ...
```

## cpp-ctor/ex2/main.cpp

```cpp
#include "Cube.h"
#include <iostream>

int main() {
  uiuc::Cube c(2);
  std::cout << "Volume: " << c.getVolume() << std::endl;
  return 0;
}
```

## Automatic Default Constructor

If any custom constructor is defined, an automatic default constructor is not defined.

## cpp-ctor/ex3/main.cpp

```cpp
#include "Cube.h"
#include <iostream>

int main() {
  uiuc::Cube c; // !!!
  std::cout << "Volume: " <<
    c.getVolume() <<
    std::endl;
  return 0;
}
```

ex3/Cube.h

```cpp
#pragma once

namespace uiuc {
  class Cube {
    public:
      Cube(double length);

      double getVolume();
      double getSurfaceArea();
      void setLength(double);

    private:
      double length_;
  };
}
```