CS 225

Data Structures

January 16 – Classes and Reference Variables

Wade Fagen-Ulmschneider, Craig Zilles

Cube.h

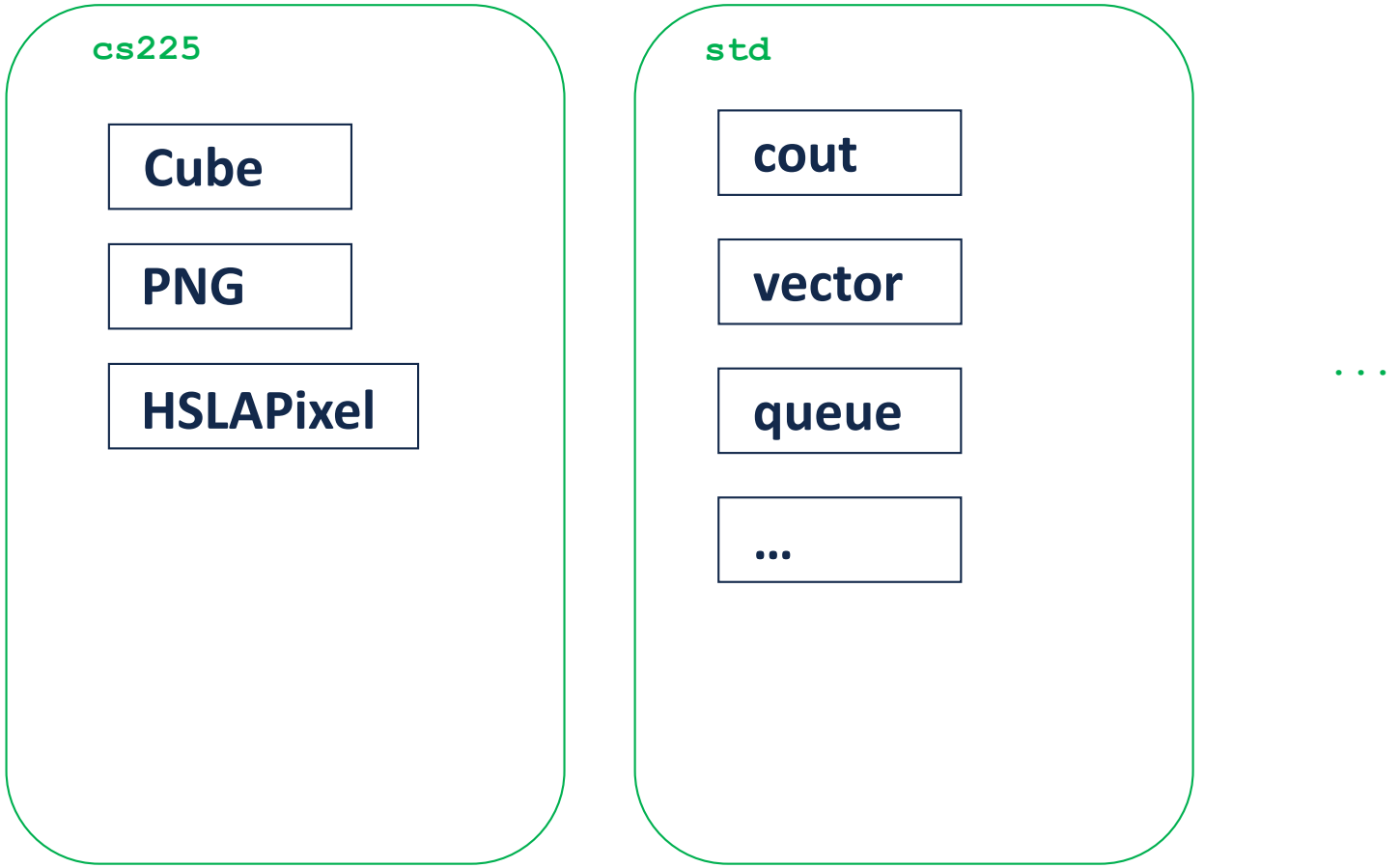
```
 1 #pragma once
 2
 3 class Cube {
 4   public:
 5     double getVolume();
 6
 7
 8
 9
10
11   private:
12
13
14 };
15
16
17
18
19
20
```

Cube.cpp

```
 1 #include "Cube.h"
 2
 3 double Cube::getVolume() {
 4
 5
 6 }
 7
 8
 9
10
11
12
13
14
15
16
17
18
19
20
```

Namespaces

Namespaces



Cube.h

```
 1 #pragma once
 2
 3 namespace cs225 {
 4   class Cube {
 5     public:
 6       double getVolume();
 7       double getSurfaceArea();
 8
 9
10
11
12     private:
13       double length_;
14
15   };
16 }
17
18
19
20
```

Cube.cpp

```
 1 #include "Cube.h"
 2
 3 namespace cs225 {
 4   double Cube::getVolume() {
 5     return length_ * length_ *
           length_;
 6   }
 7
 8   double
     Cube::getSurfaceArea() {
 9     return 6 * length_ *
           length_;
10   }
11 }
12
13
14
15
16
17
```

Cube.h

```
1  #pragma once
2
3  namespace cs225 {
4    class Cube {
5      public:
6        double getVolume();
7        double getSurfaceArea();
8
9
10
11
12     private:
13
14
15
16 }
17
18
19
20
```

Cube.cpp

```
1  #include "Cube.h"
2
3  namespace cs225 {
4    double Cube::getVolume() {
5      return length_ * length_ *
               length_;
6    }
7
8    double
     Cube::getSurfaceArea() {
9      return 6 * length_ *
```

```
1  #include "Cube.h"
2  #include <iostream>
3
4  int main() {
5    cs225::Cube c;
6    std::cout << "Volume: " << c.getVolume() << std::endl;
7    return 0;
8  }
```

main.cpp

```
1  #include "Cube.h"
2  #include <iostream>
3
4  int main() {
5    cs225::Cube c;
6    std::cout << "Volume: " << c.getVolume() << std::endl;
7    return 0;
8  }
```

main.cpp

```
1  #include "Cube.h"
2  #include <iostream>
3
4  int main() {
5    cs225::Cube c;
6    std::cout << "Volume: " << c.getVolume() << std::endl;
7    return 0;
8  }
```

main.cpp

```
1  #include "Cube.h"
2  #include <iostream>
3
4  int main() {
5    cs225::Cube c;
6    std::cout << "Volume: " << c.getVolume() << std::endl;
7    return 0;
8  }
```

Hate typing cout:: and cs225:: multiple times?

Useful Shortcut:

```
using std::cout;   // Imports cout into global scope
using std::endl;   // Imports endl into global scope
using cs225::Cube; // Imports Cube into global scope
```

Strongly Discouraged Shortcut

```
using namespace std;  // Imports all of std:: into
                      // global scope!
                      // ...THOUSANDS of things!
```

main.cpp

```
 1 #include "Cube.h"
 2 using cs225::Cube;
 3 #include <iostream>
 4 using std::cout;
 5 using std::endl;
 6
 7 int main() {
 8   Cube c;
 9   cout << "Volume: " << c.getVolume() << endl;
10   return 0;
11 }
```

CS 225 – Office Hours

Lab Sections – Meet with your TA and CAs every week!

Open Office Hours – Held in the basement of Siebel Center by TAs and CAs, ramping up over the next week. First open office hours start this Thursday. *(Great place for both conceptual and programming questions!)*

Faculty Office Hours –

Craig's Office Hours: This week, Thursday 9-11am in Siebel 4112

Wade's Office Hours: TBA

CS 225 – Exam 0

First exam is coming up *next week*!

"Exam 0"

- Low-stress introduction to the CBTF exam environment.
- This exam is worth only 40 points
- Focuses primarily on foundational knowledge you have from your prerequisite classes.

Full Details:

https://courses.engr.illinois.edu/cs225/sp2019/exams/

CBTF-based Exams

All CS 225 exams are held in the Computer Based Testing Facility (CBTF):

- You can choose which day to take your exam within the exam window for a given exam.

- Sign up for your exam here:
  https://cbtf.engr.illinois.edu/

Constructor

Cube.h

```
 1 #pragma once
 2
 3 namespace cs225 {
 4   class Cube {
 5     public:
 6       Cube();
 7       double getVolume();
 8       double getSurfaceArea();
 9
10
11
12
13     private:
14       double length_;
15
16   };
17 }
18
19
20
```

Cube.cpp

```
 1 #include "Cube.h"
 2 namespace cs225 {
 3   Cube::Cube() {
 4
 5
 6   }
 7
 8   double Cube::getVolume() {
 9     return length_ * length_ *
10            length_;
11   }
12
13   double
14   Cube::getSurfaceArea() {
15     return 6 * length_ *
16            length_;
17   }
18 }
19
20
```

Cube.h

```
 1 #pragma once
 2
 3 namespace cs225 {
 4   class Cube {
 5     public:
 6       Cube(double length);
 7       double getVolume();
 8       double getSurfaceArea();
 9
10
11
12
13     private:
14       double length_;
15
16   };
17 }
18
19
20
```

Cube.cpp

```
 1 #include "Cube.h"
 2 namespace cs225 {
 3   Cube::Cube(double length) {
 4
 5
 6   }
 7
 8   double Cube::getVolume() {
 9     return length_ * length_ *
10            length_;
11   }
12
13   double
14   Cube::getSurfaceArea() {
15     return 6 * length_ *
16            length_;
17   }
18 }
19
20
```

Cube.h

```
1  #pragma once
2
3  namespace cs225 {
4    class Cube {
5      public:
6        Cube(double length);
7        double getVolume();
8        double getSurfaceArea();
9
10
11
12
13
14
15
16
17 }
18
19
20
```

Cube.cpp

```
1  #include "Cube.h"
2  namespace cs225 {
3    Cube::Cube(double length) {
4      length_ = length;
5
6    }
7
8    double Cube::getVolume() {
```

puzzle.cpp

```
1  #include "Cube.h"
2  using cs225::Cube;
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  int main() {
8    Cube c;
9    cout << "Volume: " << c.getVolume() << endl;
10   return 0;
11 }
```

Cube.h

```
1  #pragma once
2
3  namespace cs225 {
4    class Cube {
5      public:
6
7        Cube(double length);
8        double getVolume();
9        double getSurfaceArea();
10
11
12
13
14
15
16
17
18 }
19
20
```

Cube.cpp

```
1  #include "Cube.h"
2  namespace cs225 {
3
4
5
6
7    Cube::Cube(double length) {
8      length_ = length;
9    }
10
11   double Cube::getVolume() {
```

puzzle.cpp

```
7  int main() {
8    Cube c;
9    cout << "Volume: " << c.getVolume() << endl;
10   return 0;
11 }
```

Cube.h

```
1  #pragma once
2
3  namespace cs225 {
4    class Cube {
5      public:
6
7        Cube(double length);
8        double getVolume();
9        double getSurfaceArea();
10
11
12
13
14
15
16
17
18 }
19
20
```

Cube.cpp

```
1  #include "Cube.h"
2  namespace cs225 {
3
4
5
6
7    Cube::Cube(double length) {
8      length_ = length;
9    }
10
11   double Cube::getVolume() {
```

puzzle.cpp

```
7  int main() {
8    Cube c;
9    cout << "Volume: " << c.getVolume() << endl;
10   return 0;
11 }
```

Pointers and References

Pointers and References

A variable containing an instance of an object:

```
1 Cube s1;
```

A reference variable of a Cube object:

```
1 Cube & s1;
```

A variable containing a pointer to a Cube object:

```
1 Cube * s1;
```

Reference Variable

A reference variable is an alias to an existing variable.

Key Idea: Modifying the reference variable modifies the variable being aliased.

Reference Variable

A reference variable is an alias to an existing variable.

```
 1 #include <iostream>
 2
 3 int main() {
 4   int i = 7;
 5   int & j = i;   // j is an alias of i
 6
 7   j = 4;
 8   std::cout << i << " " << j << std::endl;
 9
10   i = 2;
11   std::cout << i << " " << j << std::endl;
12   return 0;
13 }
```

Reference Variable

Three facts about reference variables:

1.

2.

3.

CS 225 – Things To Be Doing

Exam 0 starts on Thursday, Jan. 24th
Ensure you sign up for your CBTF timeslot for Exam 0!

lab_intro is due this Sunday (Jan. 20th)
Make sure to attend your lab section every week – they're worth points and EC!

MP1 is released Friday!
Due: Monday, Jan. 28th (~10 days after release)

Ensure you are on our Piazza
Details on the course website: https://courses.engr.illinois.edu/cs225/

See you Friday!