CS 225
Data Structures

January 25th- Parameters
Wade Fagen-Ulmschneider, Craig Zilles
int *x;
int size = 3;
x = new int[size];
for (int i = 0; i < size; i++) {
    x[i] = i + 3;
}
delete[] x;
/*
 * Creates a new Cube that contains the exact volume
 * of the volume of the two input Cubes.
 */

Cube joinCubes(Cube c1, Cube c2) {
    double totalVolume = c1.getVolume() + c2.getVolume();
    double newLength = std::pow( totalVolume, 1.0/3.0 );
    Cube result(newLength);
    return result;
}

int main() {
    Cube *c1 = new Cube(4);
    Cube *c2 = new Cube(5);
    Cube c3 = joinCubes(*c1, *c2);
    return 0;
}
/*
 * Creates a new Cube that contains the exact volume
 * of the volume of the two input Cubes.
 */

Cube joinCubes(Cube * c1, Cube * c2) {
    double totalVolume = c1->getVolume() + c2->getVolume();
    double newLength = std::pow( totalVolume, 1.0/3.0 );
    Cube result(newLength);
    return result;
}

int main() {
    Cube *c1 = new Cube(4);
    Cube *c2 = new Cube(5);
    Cube c3 = joinCubes(c1, c2);
    return 0;
}
/*
 * Creates a new Cube that contains the exact volume
 * of the volume of the two input Cubes.
 */

Cube joinCubes(Cube & c1, Cube & c2) {
    double totalVolume = c1.getVolume() + c2.getVolume();
    double newLength = std::pow( totalVolume, 1.0/3.0 );
    Cube result(newLength);
    return result;
}

int main() {
    Cube *c1 = new Cube(4);
    Cube *c2 = new Cube(5);
    Cube c3 = joinCubes(*c1, *c2);
    return 0;
}
## Parameter Passing Properties

|                          | By Value
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>void foo(Cube a) { ... }</td>
<td></td>
</tr>
</tbody>
</table>
|                          | By Pointer
| void foo(Cube *a) { ... } |          |
|                          | By Reference
| void foo(Cube &a) { ... } |          |

### Exactly what is copied when the function is invoked?

### Does modification of the passed in object modify the caller’s object?

### Is there always a valid object passed in to the function?

### Speed

### Programming Safety
Due: Monday, January 28th (11:59pm)

Share your artwork:
• On our piazza, in the “MP1 - Artwork Sharing” thread
• On social media:
  • If your post is public and contains #cs225, Wade will throw it a like/heart and so will some of your peers! 😊

My promise: Wade will look at all the artwork after the submission deadline. Course staff and Wade will give +1 to all that stand out!
Using `const` in function parameters
/*
 * Creates a new Cube that contains the exact volume
 * of the volume of the two input Cubes.
 */
Cube joinCubes(const Cube c1, const Cube c2) {
    double totalVolume = c1.getVolume() + c2.getVolume();
    double newLength = std::pow( totalVolume, 1.0/3.0 );
    Cube result(newLength);
    return result;
}

int main() {
    Cube *c1 = new Cube(4);
    Cube *c2 = new Cube(5);
    Cube c3 = joinCubes(*c1, *c2);
    return 0;
}
/*
 * Creates a new Cube that contains the exact volume
 * of the volume of the two input Cubes.
 */

Cube joinCubes(const Cube * c1, const Cube * c2) {
    double totalVolume = c1->getVolume() + c2->getVolume();
    double newLength = std::pow( totalVolume, 1.0/3.0 );
    Cube result(newLength);
    return result;
}

int main() {
    Cube *c1 = new Cube(4);
    Cube *c2 = new Cube(5);
    Cube c3 = joinCubes(c1, c2);
    return 0;
}
/*
 * Creates a new Cube that contains the exact volume
 * of the volume of the two input Cubes.
 */

Cube joinCubes(const Cube & c1, const Cube & c2) {
    double totalVolume = c1.getVolume() + c2.getVolume();
    double newLength = std::pow( totalVolume, 1.0/3.0 );
    Cube result( newLength );
    return result;
}

int main() {
    Cube *c1 = new Cube(4);
    Cube *c2 = new Cube(5);
    Cube c3 = joinCubes(*c1, *c2);
    return 0;
}
waf@siebl-2215-02:/mnt/c/Users/waf/Desktop/cs225/_lecture/05-parameters$
make
clang++ -std=c++1y -stdlib=libc++ -O0 -Wall -Wextra -pedantic -lpthread -lm joinCubes-byValue-const.cpp cs225/Cube.cpp -lm -o joinCubes-byValue-const
joinCubes-byValue-const.cpp:16:24: error: member function 'getVolume' not viable: 'this' argument has type 'const cs225::Cube', but function is not marked const
    double totalVolume = c1.getVolume() + c2.getVolume();
       ^
./cs225/Cube.h:9:14: note: 'getVolume' declared here
    double getVolume();
       ^
joinCubes-byValue-const.cpp:16:41: error: member function 'getVolume' not viable: 'this' argument has type 'const cs225::Cube', but function is not marked const
    double totalVolume = c1.getVolume() + c2.getVolume();
                       ^
./cs225/Cube.h:9:14: note: 'getVolume' declared here
    double getVolume();
       ^
2 errors generated.
Makefile:19: recipe for target 'joinCubes-byValue-const' failed
make: *** [joinCubes-byValue-const] Error 1
waf@siebl-2215-02:/mnt/c/Users/waf/Desktop/cs225/_lecture/05-parameters$
const as part of a member functions’ declaration
#pragma once

namespace cs225 {
    class Cube {
        public:
            Cube();
            Cube(double length);
            double getVolume();
            double getSurfaceArea();

        private:
            double length_;  
    }; 

}
/**
 * Creates a new Cube that contains the exact volume
 * of the volume of the two input Cubes.
 */
Cube joinCubes(const Cube c1, const Cube c2) {
    double totalVolume = c1.getVolume() + c2.getVolume();
    double newLength = std::pow( totalVolume, 1.0/3.0 );
    Cube result(newLength);
    return result;
}

int main() {
    Cube *c1 = new Cube(4);
    Cube *c2 = new Cube(5);
    Cube c3 = joinCubes(*c1, *c2);
    return 0;
}
Copy Constructor

[Purpose]:
All copy constructors will
Copy Constructor

Automatic Copy Constructor

Custom Copy Constructor
#pragma once

namespace cs225 {
    class Cube {
        public:
            Cube();
            Cube(double length);
            Cube(const Cube & other);
            double getVolume() const;
            double getSurfaceArea() const;
        private:
            double length_; 
    };
}

namespace cs225 {
    Cube::Cube() {
        length_ = 1;
        cout << "Default ctor" << endl;
    }
    Cube::Cube(double length) {
        length_ = length;
        cout << "1-arg ctor" << endl;
    }
... // ...