## Outline

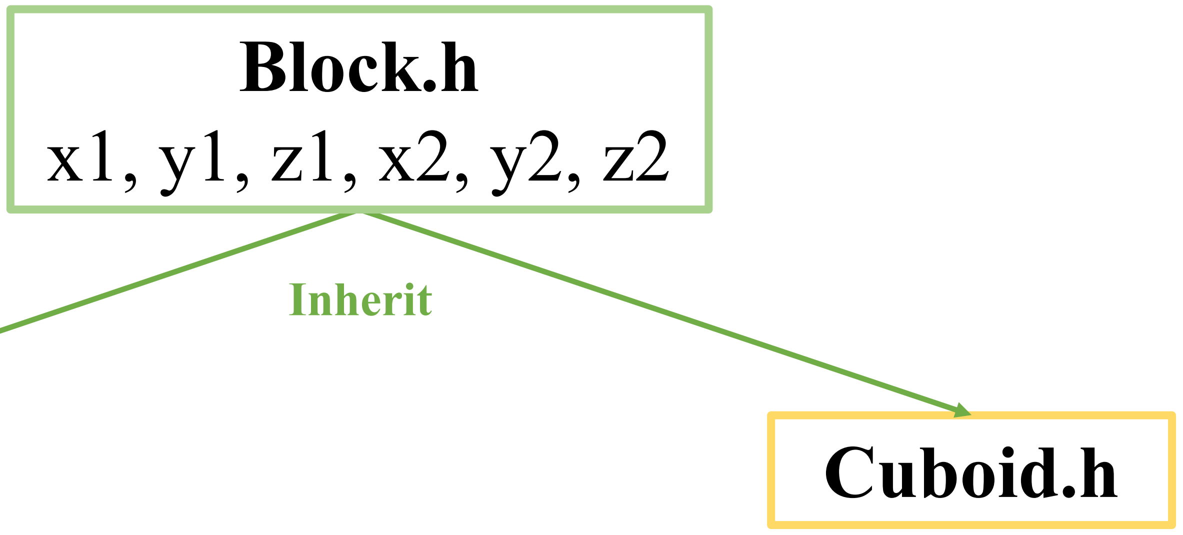
- Main goal
- Inherit & polynomial
- Operator Overloading
- File hierarchy
- Validation

## Main goal

- You should implement with polynomial and operator overloading
- Your program should support cube and cuboid

# Inherit & polynomial

- Different graphics have different attributes
- Use Inherit and polynomial to achieve cube and cuboid

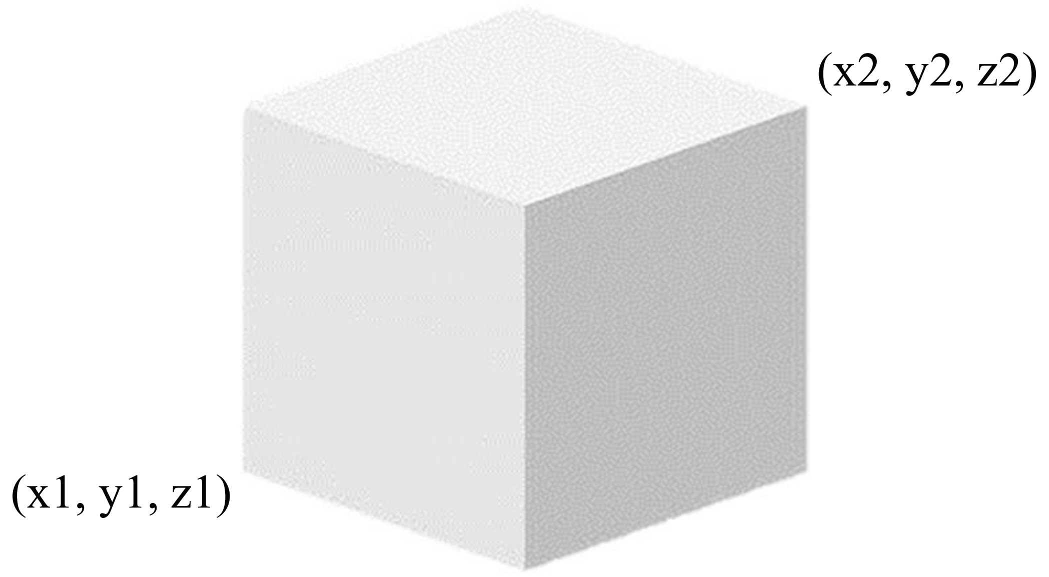


## Block.h

- Member data
- Member function
- Constructor

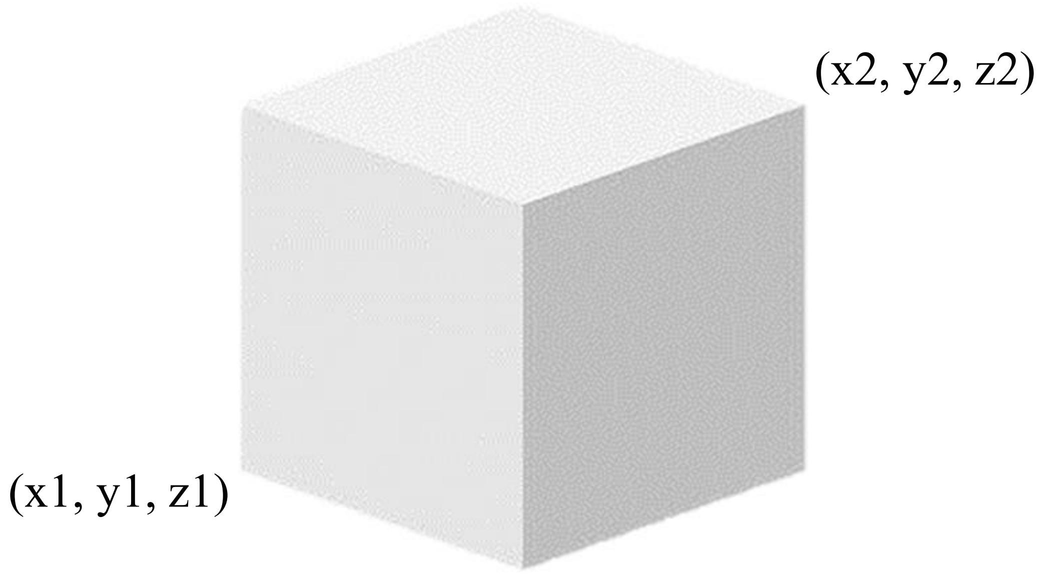
# Member data

- Two points on the diagonal
  - (x1, y1, z1)
  - (x2, y2, z2)
- Not necessarily at the same absolute position
  - Ex. (x1, y1, z1) don't need to be always at "Left Under" position



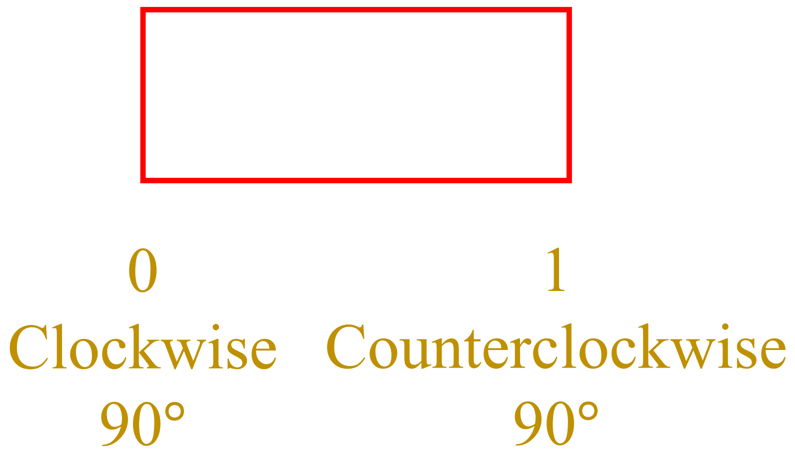
## Member function

- void setpoint(float x1, float y1, float z1, float x2, float y2, float z2);
  - Set x1, y1, z1, x2, y2, z2 respectively

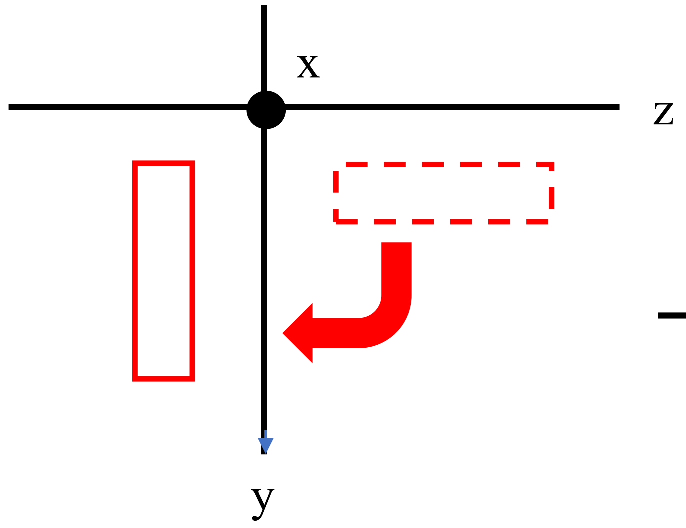


## Member function

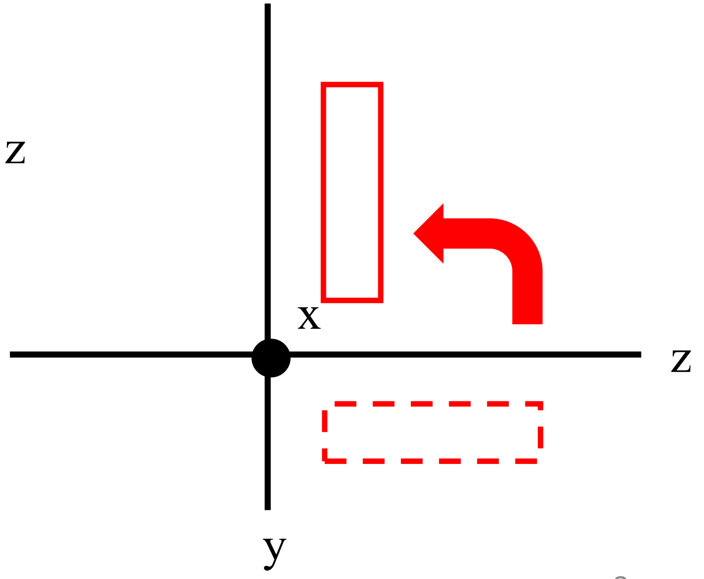
- void rotate(int p, bool d);
  - p denotes to the axis that the block rotates to
    - 0: x, 1: y, 2: z, default: z
  - d denotes to the direction of rotation
    - clockwise or counterclockwise



rotate(0, 0);

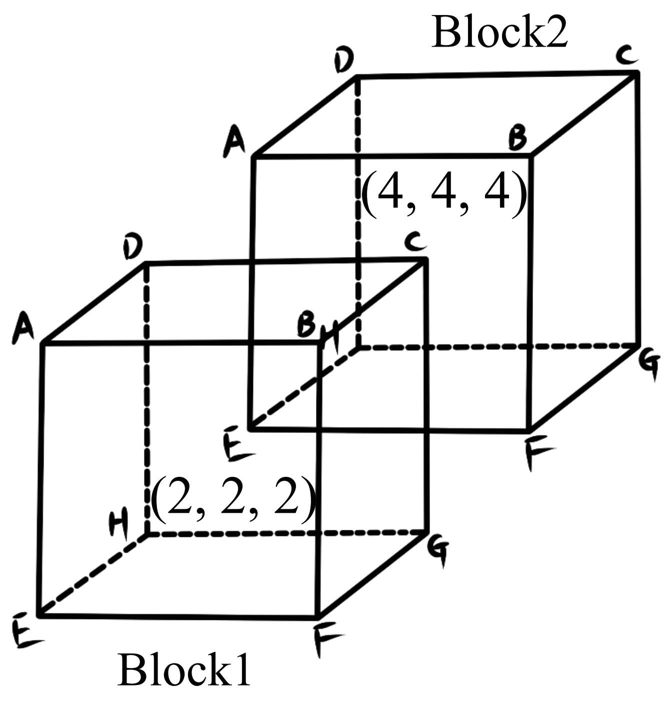


rotate(0, 1);



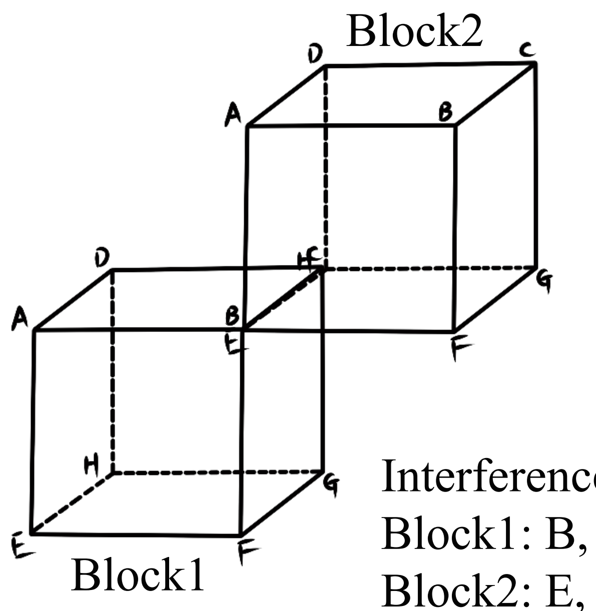
# Member function

- int interVolume(float x1, float y1, float z1, float x2, float y2, float z2);
- Take the graph for example: interference points are
  - Block1's point C and Block2's point E
- Function return the volume of the interference part
  - $2 \times 2 \times 2 = 8$

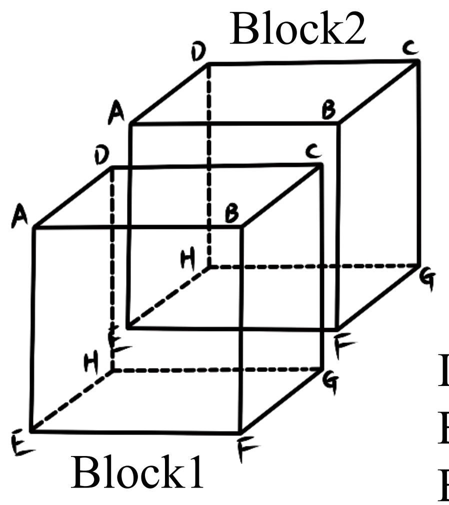


## Special Cases

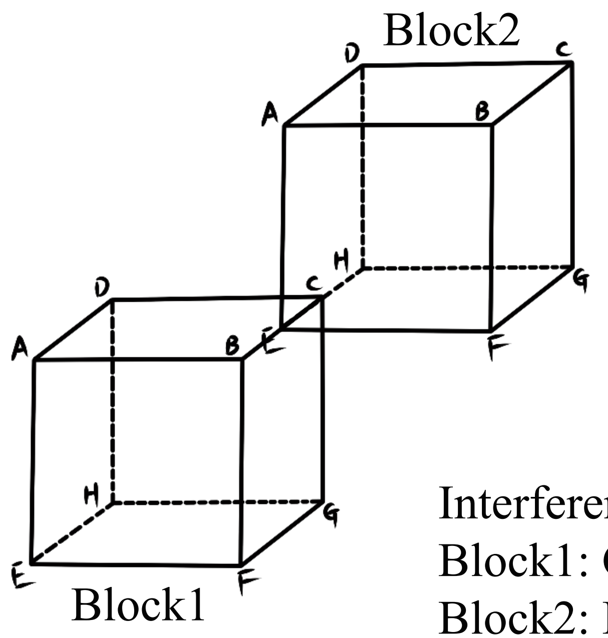
- Points of Block1 are on the surface, edge or same points of the Block 2



Interference P
Block1: B, C
Block2: E, H



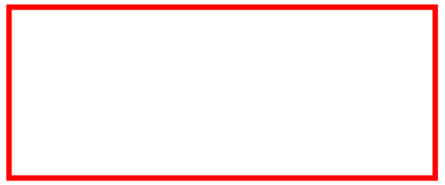
Interference P
Block1: B, C
Block2: E, H

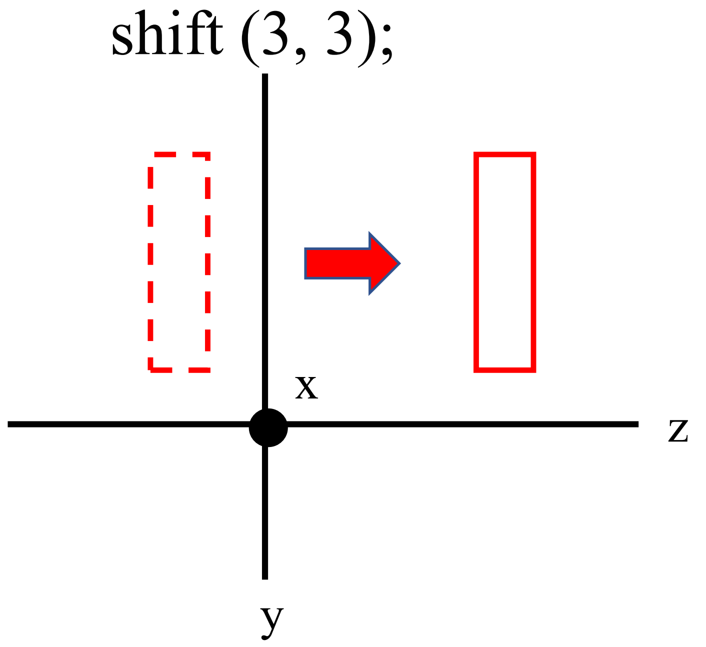


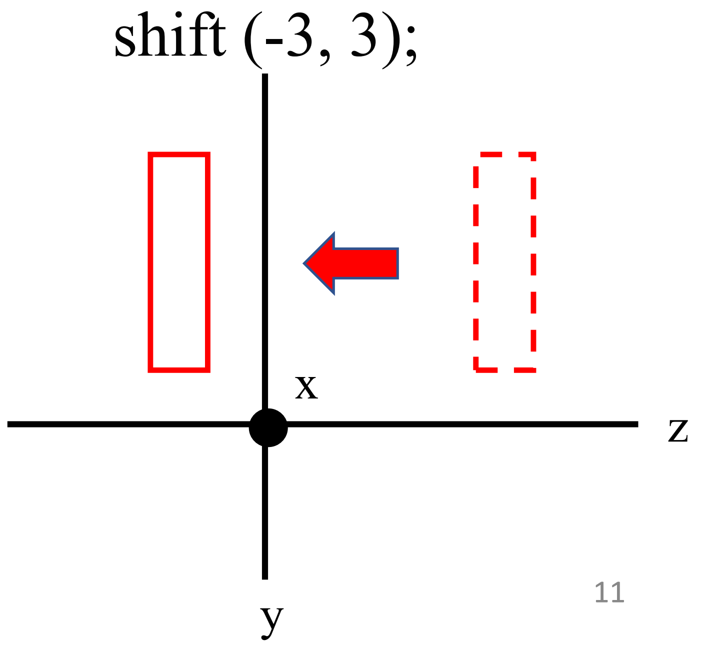
Interference P
Block1: C
Block2: E

## Member function

- virtual void shift(int p, float d)
  - p denotes to the axis that the block moves along
    - 1: x, 2: y, 3: z, default: Keep the value
    - -1: -x, -2: -y, -3: -z default: Keep the value
  - d denotes to the distance that the block moves(Distance must be positive)







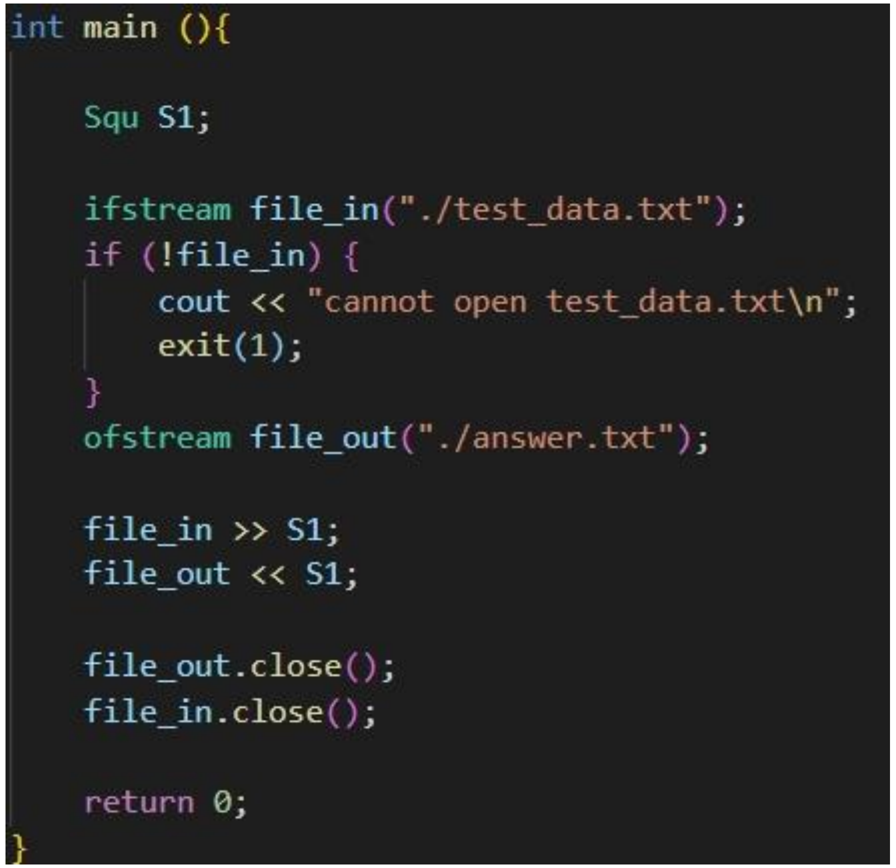
## Constructor

- Cube();
  - x1 = 0, y1 = 0, z1 = 0, x2 = 4, y2 = 4, z2 = 4
- Cube(float x1, float y,1 float z1, float x2, float y2, float z2);
  - Set x1, y1, z1, x2, y2, z2
- Cuboid();
  - x1 = 0, y1 = 0, z1 = 0, x2 = 2, y2 = 4, z2 = 4
- Cuboid (float x1, float y,1 float z1, float x2, float y2, float z2);
  - Set x1, y1, z1, x2, y2, z2
- Block(const Block & G);
  - Copy constructor
  - Set x1, y1, z1, x2, y2, z2 with those in G

## Operator Overloading

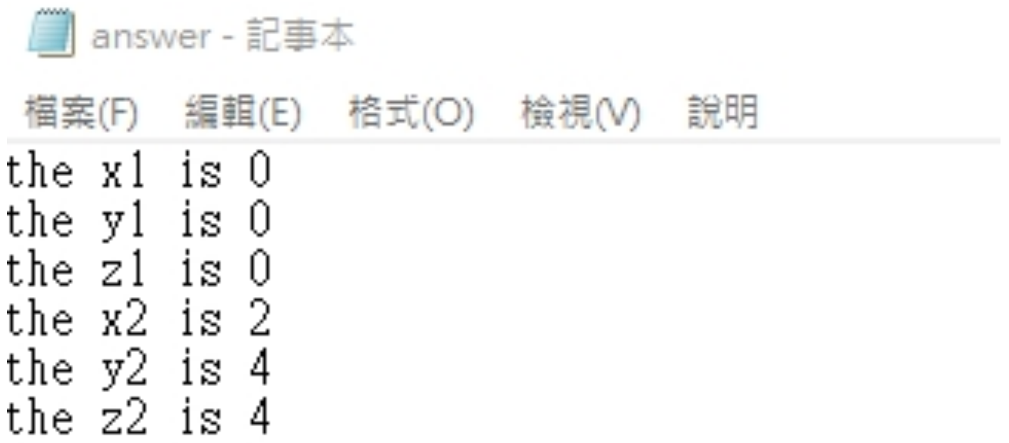
- >> : Store the information of block
- << : Show the information of block
- Support fstream
- For Cube:
    - Input: x1, y1, z1, x2, y2, z2
    - Output: x1, y1, z1, x2, y2, z2
- For Cuboid:
    - Input: x1, y1, z1, x2, y2, z2
    - Output: x1, y1, z1, x2, y2, z2

# Operator Overloading



```cpp
int main (){

    Squ S1;

    ifstream file_in("./test_data.txt");
    if (!file_in) {
        cout << "cannot open test_data.txt\n";
        exit(1);
    }
    ofstream file_out("./answer.txt");

    file_in >> S1;
    file_out << S1;

    file_out.close();
    file_in.close();

    return 0;
}
```

$$\begin{tikzcd}
0 \arrow[r] & x1 \\
0 \arrow[r] & y1 \\
0 \arrow[r] & z1 \\
2 \arrow[r] & x1 \\
4 \arrow[r] & y1 \\
4 \arrow[r] & z1
\end{tikzcd}$$

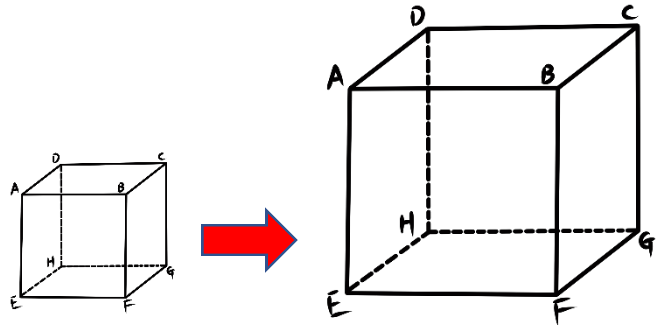


```
the x1 is 0
the y1 is 0
the z1 is 0
the x2 is 2
the y2 is 4
the z2 is 4
```

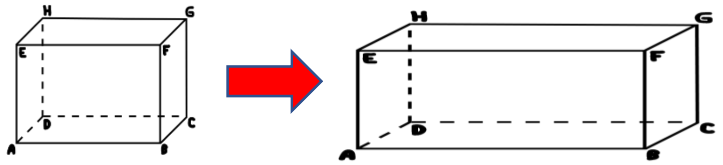
# Operator Overloading

- * : Enlarge the size of the block
    - Fix the point(x1, y1, z1)
    - Return the larger volume (float, become n times larger than original one)
    - x2 may be change

For Cube(n = 8):
Multiply all edges length by $\sqrt[3]{n}$



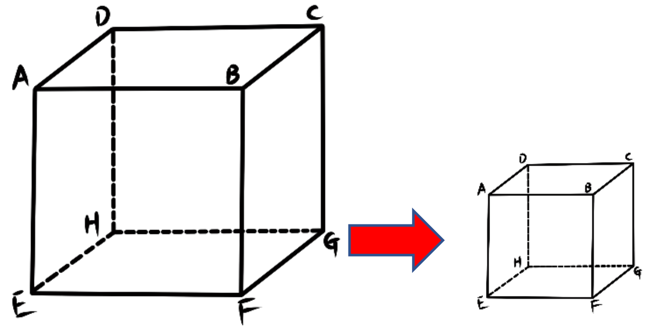
For Cuboid(n = 8):
Multiply the length  of edge in x direction by n,
while the remaining two directions doesn't change



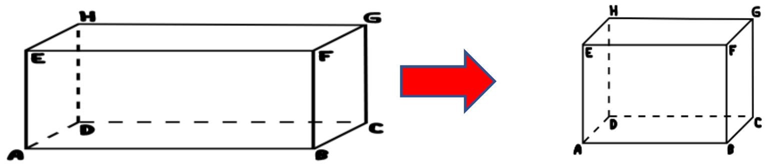
# Operator Overloading

- / : Shrink the size of the block
  - Fix the point (x1, y1, z1)
  - Return the smaller volume (float, become 1/n times smaller than original one)
  - x2 may be change

For Cube(n = 8):
Divide all edges length by $\sqrt[3]{n}$



For Cuboid(n = 8):
Divide the length  of edge in x direction by n,
while the remaining two directions doesn't change



## File hierarchy

- your_ID (replace it by your student ID)
  - main.cpp
  - Block.h
  - Block.cpp
  - Cube.h
  - Cube.cpp
  - Cuboid.h
  - Cuboid.cpp
  - Makefile (executable must be named “main”)

You can download this file from Moodle
Do not follow this rule may be taken 2 pts off
Do not hand in other files! (*.exe or *.o)