```
    1: //
    2: //  original.cpp
    3: //  ps1
    4: //
    5: //  Created by Jingxian Shi on 1/30/18.
    6: //  Copyright Â© 2018 Jingxian Shi. All rights reserved.
    7: //
    8:
    9: #include "original.hpp"
   10: #include "SFML/Graphics/Shape.hpp"
   11: #include <cmath>
   12:
   13: Original::Original(int depth, double side)
   14: {
   15:     recursion_depth = depth;
   16:     _top_left = sf::Vector2f(side, side);
   17:     _top_right = sf::Vector2f(side*2, side);
   18:     _bot_left = sf::Vector2f(side, side*2);
   19:     _bot_right = sf::Vector2f(side*2, side*2);
   20:     outline_square(_top_left, _top_right, _bot_right, _bot_left);
   21: }
   22:
   23: void Original::outline_square(sf::Vector2f p1, sf::Vector2f p2, sf::Vector2f p3,
sf::Vector2f p4)
   24: {
   25:     setPointCount(4);
   26:     setPoint(0, p1);
   27:     setPoint(1, p2);
   28:     setPoint(2, p3);
   29:     setPoint(3, p4);
   30:     setOutlineColor(sf::Color::Black);
   31:     setOutlineThickness(-1);
   32: }
   33:
   34: Original::Original(int depth, sf::Vector2f top_left, sf::Vector2f top_right, sf::
Vector2f bot_right, sf::Vector2f bot_left)
   35: {
   36:     recursion_depth = depth;
   37:     _top_left = top_left;
   38:     _top_right = top_right;
   39:     _bot_left = bot_left;
   40:     _bot_right = bot_right;
   41:
   42:     outline_square(_top_left, _top_right, _bot_right, _bot_left);
   43: }
   44:
   45: void Original::draw(sf::RenderTarget& target, sf::RenderStates states) const
   46: {
   47:     target.draw((sf::ConvexShape)(*this), states);
   48:     if(recursion_depth <= 0)
   49:     {
   50:         return;
   51:     }
   52:     else
   53:     {
   54:         Original top_left(recursion_depth-1,
   55:                           sf::Vector2f(_top_left.x-(_top_right.x-_top_left.x)/2.0
,
   56:                                        _top_left.y+(_top_left.y-_bot_left.y)/2.0)
,
   57:                           sf::Vector2f(_top_left.x,
   58:                                        _top_left.y+(_top_left.y-_bot_left.y)/2.0)
,
   59:                           _top_left,
   60:                           sf::Vector2f(_top_left.x-(_top_right.x-_top_left.x)/2.0
```

```
,
   61:                                          _top_left.y));
   62:         top_left.draw(target, states);
   63:
   64:         Original top_right(recursion_depth-1,
   65:                            sf::Vector2f(_top_right.x, _top_right.y+(_top_right.y-
_bot_right.y)/2.0),
   66:                            sf::Vector2f(_top_right.x+(_top_right.x-_top_left.x)/2
.0,
   67:                                         _top_right.y+(_top_right.y-_bot_right.y)/
2.0),
   68:                            sf::Vector2f(_top_right.x+(_top_right.x-_top_left.x)/2
.0,
   69:                                         _top_right.y),
   70:                            _top_right);
   71:         top_right.draw(target, states);
   72:
   73:         Original bot_left(recursion_depth-1,
   74:                           sf::Vector2f(_bot_left.x-(_bot_right.x-_bot_left.x)/2.0
, _bot_left.y),
   75:                           _bot_left,
   76:                           sf::Vector2f(_bot_left.x, _bot_left.y+(_bot_left.y-_top
_left.y)/2.0),
   77:                           sf::Vector2f(_bot_left.x-(_bot_right.x-_bot_left.x)/2.0
,
   78:                                        _bot_left.y+(_bot_left.y-_top_left.y)/2.0)
);
   79:         bot_left.draw(target, states);
   80:
   81:         Original bot_right(recursion_depth-1,
   82:                            _bot_right,
   83:                            sf::Vector2f(_bot_right.x+(_top_right.x-_top_left.x)/2
.0,
   84:                                         _bot_right.y),
   85:                            sf::Vector2f(_bot_right.x+(_top_right.x-_top_left.x)/2
.0,
   86:                                         _bot_right.y+(_bot_right.y-_top_right.y)/
2.0),
   87:                            sf::Vector2f(_bot_right.x,
   88:                                         _bot_right.y+(_bot_right.y-_top_right.y)/
2.0));
   89:         bot_right.draw(target, states);
   90:     }
   91: }
```

```
    1: //
    2: //  original.hpp
    3: //  ps1
    4: //
    5: //  Created by Jingxian Shi on 1/30/18.
    6: //  Copyright Â© 2018 Jingxian Shi. All rights reserved.
    7: //
    8:
    9: #ifndef original_hpp
   10: #define original_hpp
   11:
   12: #include <stdio.h>
   13: #include <SFML/Graphics.hpp>
   14: #include <SFML/Graphics/ConvexShape.hpp>
   15: #include <iostream>
   16:
   17: using namespace std;
   18:
   19: class Original : public sf::ConvexShape
   20: {
   21: public:
   22:     Original(int depth, double side);
   23:     Original(int depth, sf::Vector2f top_left, sf::Vector2f top_right, sf::Vector
2f bot_left, sf::Vector2f bot_right);
   24:     virtual void draw(sf::RenderTarget& target, sf::RenderStates states) const;
   25:     void outline_square(sf::Vector2f p1, sf::Vector2f p2, sf::Vector2f p3, sf::Ve
ctor2f p4);
   26: private:
   27:     int recursion_depth;
   28:     sf::Vector2f _top_left, _top_right, _bot_left, _bot_right;
   29: };
   30:
   31: #endif /* original_hpp */
```