```cpp
#include <iostream>
#include <string.h>
#include <math.h>
#include <vector>
#define pi 3.1416
using namespace std;
class Point {
public:
    float x, y;
    Point() {
        x = 0;
        y = 0;
    }
    Point(int x, int y){
        this->x = x;
        this->y = y;
    }
    void operator<<(char s[]){
        cout<< s << "(" << this->x << ", " << this->y << ")" << endl;
    }
};

class Shape{
protected:
    char* name;
    float side;
public:
    Shape(){
        name = new char(1);
        strcpy(name, "");
        side = 1;
    }
    Shape(float a, char *name = ""){
        this->name = new char(strlen(name) + 1);
        strcpy(this->name, name);
        side = a;
    }
    ~Shape(){
        delete name;
    }
    char* getName(){
        return name;
    }
    float getAside(){
        return side;
    }
    virtual float findPerimeter() = 0;
};
class TwoDimensionalShape:public Shape{
protected:
    Point startVertex;
public:
    TwoDimensionalShape(): Shape(){}
    TwoDimensionalShape(Point vertex, float a, char*name = ""): Shape(a, name){
        startVertex = vertex;
    }
    virtual float findArea() = 0;
};

class Circle:public TwoDimensionalShape{
public:
    Circle():TwoDimensionalShape(){}
    Circle(Point ct, float r, char *ner = ""):TwoDimensionalShape(ct, r, ner){}
    float findArea(){
        return side*side*pi;
    }
    float findPerimeter(){
        return 2*side*pi;
    }
    void setCenterPoint(Point ct){
        startVertex = ct;
    }
    void setRadius(float radius){
        side = radius;
    }
    void getData(){
        startVertex<<"CP";
        cout<<"Name: "<<name<<endl;
        cout<<"Radius: "<<side<<endl;
    }
};
class Square:public TwoDimensionalShape{
private:
    Point B;
```

```
85        Point C;
86        Point D;
87        void calcOtherPoints(){
88           B.x = startVertex.x + side;
89           B.y = startVertex.y;
90           C.x = startVertex.x + side;
91           C.y = startVertex.y - side;
92           D.x = startVertex.x;
93           D.y = startVertex.y - side;
94        }
95    public:
96        Square():TwoDimensionalShape(){
97            Point A(-1, 1);
98            startVertex = A;
99            calcOtherPoints();
100       }
101       Square(Point lt_p, float a, char *name = ""):TwoDimensionalShape(lt_p, a, name){
102           setLTpoint(lt_p);
103       }
104       float findArea(){
105           return side*side;
106       }
107       float findPerimeter(){
108           return 4*side;
109       }
110       void setAside(float a){
111           side = a;
112           calcOtherPoints();
113       }
114       void setLTpoint(Point lt_p){
115           startVertex = lt_p;
116           calcOtherPoints();
117       }
118       void getData(){
119           startVertex<<"A";
120           B<<"B";
121           C<<"C";
122           D<<"D";
123           cout<<"Name: "<<name<<endl;
124           cout<<"A side: "<<side<<endl;
125       }
126   };
127   class RightTriangle:public TwoDimensionalShape{
128   private:
129       Point B;
130       Point C;
131       //BC taliig Ox tenkhlegtei parallel gj uzsen bolno.
132       void calcOtherPoints(){
133           B.x = startVertex.x + side/2;
134           B.y = startVertex.y - sqrt(3)/2*side;
135           C.x = startVertex.x - side/2;
136           C.y = startVertex.y - sqrt(3)/2*side;
137       }
138   public:
139       RightTriangle(): TwoDimensionalShape(){
140           Point A(0, 1);
141           startVertex = A;
142           calcOtherPoints();
143       }
144       RightTriangle(Point pt, float a, char *name = ""):TwoDimensionalShape(pt, a, name){
145           setTpoint(pt);
146       }
147       float findArea(){
148           return sqrt(3)*side*side/4;
149       }
150       float findPerimeter(){
151           return 3*side/2;
152       }
153       void setAside(float a){
154           side = a;
155           calcOtherPoints();
156       }
157       void setTpoint(Point pt){
158           startVertex = pt;
159           calcOtherPoints();
160       }
161       void getData(){
162           startVertex<<"A";
163           B<<"B";
164           C<<"C";
165           cout<<"Name: "<<name<<endl;
166           cout<<"A side: "<<side<<endl;
167       }
168   };
```

```
169
170 struct Shape_square{
171     char name[10];
172     float area;
173 };
174 typedef Shape_square Shape_square;
175
176 void add_shape_area(Shape_square* sh_a, char *n, float a){
177     sh_a->area = a;
178     strcpy(sh_a->name, n);
179 }
180
181 int main(){
182     Point ct(1, 1), st(-2, 2), tt(2, 2), ct1(1, -1);
183     vector<TwoDimensionalShape*>vec;
184
185     cout<<"Unsorted shapes: "<<endl;
186     Circle c1(ct, 3, "C1");
187     vec.push_back(&c1);
188     cout<<"\nCircle: "<<endl;
189     c1.getData();
190     cout<<"Area: "<<c1.findArea()<<endl;
191     cout<<"Perimeter: "<<c1.findPerimeter();
192     cout<<"\n";
193
194     Circle c2(ct, 2, "C2");
195     vec.push_back(&c2);
196     cout<<"\nCircle1: "<<endl;
197     c2.getData();
198     cout<<"Area: "<<c2.findArea()<<endl;
199     cout<<"Perimeter: "<<c2.findPerimeter();
200     cout<<"\n";
201
202     Square s1(st, 2, "S1");
203     vec.push_back(&s1);
204     cout<<"\nSquare: "<<endl;
205     s1.getData();
206     cout<<"Area: "<<s1.findArea()<<endl;
207     cout<<"Perimeter: "<<s1.findPerimeter();
208     cout<<"\n";
209
210     RightTriangle rt1(tt, 2, "RT1");
211     vec.push_back(&rt1);
212     cout<<"\nRightTrangle: "<<endl;
213     rt1.getData();
214     cout<<"Area: "<<rt1.findArea()<<endl;
215     cout<<"Perimeter: "<<rt1.findPerimeter();
216     cout<<"\n";
217
218     int i, j;
219     TwoDimensionalShape *key;
220
221     for (i = 1; i < vec.size(); i++)
222     {
223         key = vec[i];
224         j = i - 1;
225
226         while (j >= 0 && vec[j]->findArea() > key->findArea())
227         {
228             vec[j + 1] = vec[j];
229             j = j - 1;
230         }
231         vec[j + 1] = key;
232     }
233
234
235     cout<<"\n\nSorted shapes: "<<endl;
236     for (vector<TwoDimensionalShape*>::iterator i = vec.begin(); i != vec.end(); ++i) {
237         cout<<"Name: "<<(*i)->getName()<<" Area:"<<(*i)->findArea()<<endl;
238     }
239
240     return 0;
241 }
242
243
244
```

```
Unsorted shapes:

Circle:
CP(1, 1)
Name: C1
Radius: 3
Area: 28.2744
Perimeter: 18.8496

Circle1:
CP(1, 1)
Name: C2
Radius: 2
Area: 12.5664
Perimeter: 12.5664

Square:
A(-2, 2)
B(0, 2)
C(0, 0)
D(-2, 0)
Name: S1
A side: 2
Area: 4
Perimeter: 8

RightTrangle:
A(2, 2)
B(3, 0.267949)
C(1, 0.267949)
Name: RT1
A side: 2
Area: 1.73205
Perimeter: 3


Sorted shapes:
Name: RT1 Area:1.73205
Name: S1 Area:4
Name: C2 Area:12.5664
Name: C1 Area:28.2744

Process returned 0 (0x0)   execution time : 0.345 s
```