```
/*
  File: rectangles.cpp
  Created by: Tan Qi Hao
  Creation Date: 4/18/2019
  Synopsis: This program create a list of axis-aligned rectangles defined
in a
  2D-space.
*/

#include <iostream>
#include <string>
#include <vector>

using namespace std;

class Point
{
private:
      double px;
      double py;

public:
      void setX(const double x);
      void setY(const double y);
      double getX() const;
      double getY() const;
};

class Rectangle
{
private:
      string name;
      Point blPoint;
      double length, height;

public:
      // member functions
      void setName(const string & inName);
      void setBottomLeft(const double x, const double y);
      void setDimensions(const double inLength, const double inHeight);

      string getName() const;
      Point getBottomLeft() const;
      double getLength() const;
      double getHeight() const;

      double area() const;
      double perimeter() const;
      Point midPoint() const;
      void scaleBy3();
      void display() const;
};

// FUNCTION PROTOTYPES GO HERE:
```

```
void welcome();
bool promp_read(string prompt, string error_inp, string error_used, string
& rectangle_name, vector <Rectangle> & list);
void promp_bl(string prompt_bl, double & x, double & y);
void promp_dimensions(string prompt_dimen, double & length, double &
height);
void adds_rectangle(string rectangle_name, double x, double y, double
length, double height, vector <Rectangle> & list);
void displayRec(vector <Rectangle> list);

int main()
{
      // Define your local variables, e.g. a vector of class Rectangle
  string prompt = "Enter the name of the first rectangle: "; //Prompt for
input
  string prompt2 = "Enter the name of the next rectangle: "; //Prompt for
the next input

 //Error_inp prompt when user input the wrong input
  string error_inp = "Invalid input. Type 'rec' followed by the name or
'stop' if done.";
  //error_used prompt when user type the name again
  string error_used = "This name is already being used! ";

  string rectangle_name; //Name of the rectangle
  string prompt_bl; //Prompt when ask for the bottom low point
  string prompt_bl1 = "Enter "; //First part of the prompt_bl
  string prompt_bl2 = "'s bottom left x and y coords: "; //Second part of
the prompt_bl
  string prompt_dimen; //Prompt when ask for dimension
  string prompt_dimen1 = "Enter "; //First part of prompt_dimen
  string prompt_dimen2 = "'s length and height: "; //Second part of
prompt_dimen

  vector<Rectangle> list; //Vector list of rectangle

  double x, y, length, height; //x-coordinates, y-coordinates, length of
rectangle and height of rectangle

      // Display welcome banner
  welcome();

      /* Prompt user for first rectangle or 'stop' */
      // WHILE user input is invalid
                  // Display "Try again! "

  while( !promp_read(prompt, error_inp, error_used, rectangle_name,
list)){
    cout << endl;
      cout << "Try again! ";

    }

      // IF user input is not 'stop'
```

```
                // Extract rectangle name from user input
                // Prompt for bottom left point
                // Prompt for length and height
                // Add rectangle to the rectangle list

  if( rectangle_name != "stop"){
    prompt_bl = prompt_bl1 + rectangle_name + prompt_bl2;
    promp_bl(prompt_bl, x, y);

    prompt_dimen = prompt_dimen1 + rectangle_name + prompt_dimen2;
    promp_dimensions(prompt_dimen, length, height);
    adds_rectangle(rectangle_name, x, y, length, height, list);

  }

      /* Prompt user for next rectangle or 'stop' */
      // WHILE user input is not 'stop'
                // Display "Thank you! "

  while(rectangle_name != "stop"){

    cout << "Thank you! ";

                // WHILE user input is invalid
                          // Display "Try again! "

 while( !promp_read(prompt2, error_inp, error_used, rectangle_name,
list)){

   cout << endl;
      cout << "Try again! ";

    }
      // IF user input is not 'stop'
                                  // Extract rectangle name from user
input
                                  // Prompt for bottom left point
                                  // Prompt for length and height
                                  // Add rectangle to the rectangle list

 if( rectangle_name != "stop"){

   prompt_bl = prompt_bl1 + rectangle_name + prompt_bl2;
    promp_bl(prompt_bl, x, y);

    prompt_dimen = prompt_dimen1 + rectangle_name + prompt_dimen2;
    promp_dimensions(prompt_dimen, length, height);
    adds_rectangle(rectangle_name, x, y, length, height, list);


 }

  }
```

```
      // IF the rectangle list is not empty
  if(list.size() != 0){

                // Display all rectangles in the rectangle list

    displayRec(list);


  }
      // ELSE
                // Display that no rectangles are in the list

  else{

    cout << "You have no rectangles in your list." << endl;

  }

  return 0;
}

// FUNCTION DEFINITIONS GO HERE:
//This function welcome the user.
void welcome(){
  cout << "Welcome! Create your own list of rectangles." << endl;
  cout << "You will be asked to provide information about each rectangle
in your list by name." << endl;
  cout << "Type the word 'stop' for the rectangle name when you are done."
<< endl;
  cout << endl;
}

//This function prompt and reads from the user the name of the rectangle
or stop.
bool promp_read(string prompt, string error_inp, string error_used, string
& rectangle_name, vector <Rectangle> & list){

  string input; //user's input of of Rec

  cout << prompt;
  cin >> input;

  if (input == "rec"){

    string rec_name; //user input of name
    cin >> rec_name;

    for(int i = 0; i < list.size(); i++){

      if(rec_name == list[i].getName()){

      cout << error_used;
      return false;
```

```
      }

    }

    for(int j = 0; j < rec_name.size(); j++){

      if(isalpha(rec_name[j]) == false){

              cout << error_inp;
              return false;

      }

    }

    rectangle_name = rec_name;
    return true;

  }

  else if(input == "stop"){

    rectangle_name = "stop";
    return true;

  }

  cout << error_inp;
  return false;

}

//This function prompt the bottom low xy coordinates.
void promp_bl(string prompt_bl, double & x, double & y){

  cout << prompt_bl;
  cin >> x;
  cin >> y;

}

//This function prompt and read the dimension of the rectangle.
void promp_dimensions(string prompt_dimen, double & length, double &
height){

  cout << prompt_dimen;
  cin >> length;
  cin >> height;

  while(length <= 0 || height <= 0){

    cout << "Make length and height positive values. Try again. " << endl;
    cout << prompt_dimen;
    cin >> length;
```

```
    cin >> height;

  }
  cout << endl;
}

//This function adds a rectangles to back of a vector rectangles
void adds_rectangle(string rectangle_name, double x, double y, double
length, double height, vector<Rectangle> & list){

  Rectangle rec; //Rectangle variable

  rec.setName(rectangle_name);
  rec.setBottomLeft(x, y);
  rec.setDimensions(length, height);
  list.push_back(rec);

}

//This function display rectangle
void displayRec(vector <Rectangle> list){

  cout << "You have " << list.size() << " rectangle(s) in your list: " <<
endl;

  cout << endl;

  for(int j = 0; j < list.size(); j++){

    cout << "Rectangle '" << list[j].getName() << "':";
    list[j].display();

    cout << "     After scale by 3:";
    list[j].scaleBy3();
    list[j].display();

    cout << endl;
  }

}

// CLASS MEMBER FUNCTION DEFINITINOS GO HERE:

void Point::setX(const double x)
{
     px = x;
}

void Point::setY(const double y)
{
     py = y;
}

double Point::getX() const
```

```
{
      return (px);
}

double Point::getY() const
{
      return (py);
}

void Rectangle::setName(const string & inName)
{
  name = inName;
}

void Rectangle::setBottomLeft(const double x, const double y)
{
  blPoint.setX(x);
  blPoint.setY(y);
}

void Rectangle::setDimensions(const double inLength, const double
inHeight)
{
  length = inLength;
  height = inHeight;
}

string Rectangle::getName() const
{
  return name;
}

Point Rectangle::getBottomLeft() const
{
  return blPoint;
}

double Rectangle::getLength() const
{
  return length;
}

double Rectangle::getHeight() const
{
  return height;
}

double Rectangle::area() const
{
  return height * length;
}

double Rectangle::perimeter() const
{
```

```
  return ((2 * height) + (2 * length));
}

Point Rectangle::midPoint() const
{
  Point mid_pt; //mid point
  double midx, midy; //x & y coordinate of mid point

  midx = (blPoint.getX() + (blPoint.getX() + length))/2;
  midy = (blPoint.getY() + (blPoint.getY() + height))/2;

  mid_pt.setX(midx);
  mid_pt.setY(midy);

  return mid_pt;

}

void Rectangle::scaleBy3()
{
  double x, y; //The xy-coordinate after scale by 3

  x = blPoint.getX() + 0.5 * length;
  y = blPoint.getY() + 0.5 * height;

  setDimensions(length * 3, height * 3);

  blPoint.setX(x - 0.5 * length);
  blPoint.setY(y - 0.5 * height);


}

void Rectangle::display() const
{

  cout << "': Location is (" << blPoint.getX()
       << ", " << blPoint.getY() << "), Length is " << length
       << ", Height is " << height << "; Area is " << area() << ", "
       << "Perimeter is " << perimeter() << ", Midpoint is located at ("
       << midPoint().getX() << ", " << midPoint().getY() << ")"
       << endl;



}
```