GEOMETRÍA

0

Generated by Doxygen 1.9.1

Chapter 1

Main Page

Given two rectangles and a sequence of points this program calculates which points are inscribed within the intersection of the two rectangles

- programmatically sets the data of the first rectangle
- reads the second rectangle from keyboard
- calculates the intersection
  - if the intersection is empty, it ends
  - otherwise it reads the points and for each point
    - Check if the point belongs the intersection
    - Counts the number of points inscribed in the intersection
    - The sequence ends when a point with any negative coordinate is read

```
      B{(1,6) (6,1)}
6+     +-----------------------------o
 |     |                             |
 |     |    A{(2,5) (5,2)}           |
5+     |    +------------------o     |
 |     |    |                  |     |
 |     |    |                  |     |
4+     |    |           o      |     |
 |     |    |                  |     |
 |     |    |                  |     |
3+     |    |     o            |     |
 |     |    |                  |     |
 |     |    |                  |     |
2+     |    o------------------+     |
 |     |                             |
 |     |                             |
1+     o-----------------------------+
 |
 |
 o-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
 0     1     2     3     4     5     6     7     8     9     10
 A={(2,5) (5,2)}     B={(1,6) (6,1)}
 I={(2,5) (5,2)}
 P={(0,0), (1,1), (2,2), (3,3), (4,4), (5,5), (6,6)}
 P(I)={(2,2), (3,3), (4,4), (5,5)}
```

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 Point2D Class Reference

To represent a point in a two-dimensional space.

```
#include <Point2D.h>
```

Public Member Functions

- Point2D ()

  Basic constructor.
- Point2D (int x, int y)

  Constructor with initialization parameters.
- void setX (int px)

  Initializes the X coordinate.
- void setY (int py)

  Initializes the Y coordinate.
- int getX () const

  Queries the X coordinate.
- int getY () const

  Queries the Y coordinate.
- void read ()

  Reads the XY value from keyboard.
- void print () const

  Prints the XY values in the screen in the form (X,Y)

4.1.1 Detailed Description

To represent a point in a two-dimensional space.

Definition at line 13 of file Point2D.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Point2D()

```
Point2D::Point2D (
            int x,
            int y )
```

Constructor with initialization parameters.

Parameters

x	Coordinate
y	Coordinate

Definition at line 16 of file Point2D.cpp.

```
                              {
    px = x;
    py = y;
}
```

4.1.3 Member Function Documentation

4.1.3.1 getX()

```
int Point2D::getX ( ) const
```

Queries the X coordinate.

Returns

Value of X

Definition at line 29 of file Point2D.cpp.

```
                         {
    return px;
}
```

4.1.3.2 getY()

```
int Point2D::getY ( ) const
```

Queries the Y coordinate.

Returns

Value of Y

Definition at line 33 of file Point2D.cpp.

```
                         {
    return py;
}
```

4.1.3.3 setX()

```
void Point2D::setX (
              int px )
```

Initializes the X coordinate.

Parameters

px	New value for X

Definition at line 21 of file Point2D.cpp.

```
                          {
      this->px = px;
  }
```

4.1.3.4 setY()

```
void Point2D::setY (
              int py )
```

Initializes the Y coordinate.

Parameters

py	New value for Y

Definition at line 25 of file Point2D.cpp.

```
                          {
      this->py = py;
  }
```

The documentation for this class was generated from the following files:

- include/Point2D.h
- src/Point2D.cpp

4.2 Rectangle Class Reference

To represent a rectangle in a two-dimensional space as a pair or points, the top-left corner and the bottom-right one.

```
#include <Rectangle.h>
```

Public Member Functions

- Rectangle ()

  Basic constructor.
- Rectangle (int x, int y, int w, int h)

  Constructor with parameters.
- void setGeometry (int x, int y, int w, int h)

  Initializes the data of the rectangle.
- void setGeometry (const Point2D &tl, const Point2D &br)

  Initializes the data of the rectangle.
- Point2D getTopLeft () const

  Queries the top-left corner.
- Point2D getBottomRight () const

  Queries the bottom-right corner.
- bool isEmpty () const

  For a rectangle to be valid this condition must hold topleft.getX() $<$ = bottomright.getX() && topleft.getY() $>$ = bottomright.getY() otherwise it is an empty (incorrect) rectangle.
- void read ()

  Reads the two points of the rectangle.
- void print () const

  Prints the rectangle in the form [Point2D - Point2D].

Friends

- Rectangle doOverlap (const Rectangle &r1, const Rectangle &r2)

  Calculates the rectangle intersection of the two given rectangles. If there is no intersection, an empty rectangle is returned instead.

4.2.1 Detailed Description

To represent a rectangle in a two-dimensional space as a pair or points, the top-left corner and the bottom-right one.

Definition at line 15 of file Rectangle.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Rectangle()

```
Rectangle::Rectangle (
              int x,
              int y,
              int w,
              int h )
```

Constructor with parameters.

Parameters

x	XY Coordinates of top-left corner
y	
w	Width of the rectangle
h	Height of the rectangle

Definition at line 14 of file Rectangle.cpp.

```
                                                  {
    topleft.setX(x);
    topleft.setY(y);
    bottomright.setX(x+w);
    bottomright.setY(y-h);
    // setGeometry(x,y,w,h);
}
```

4.2.3 Member Function Documentation

4.2.3.1 getBottomRight()

```
Point2D Rectangle::getBottomRight ( ) const
```

Queries the bottom-right corner.

Returns

The point

Definition at line 38 of file Rectangle.cpp.

```
                                          {
    return bottomright;
}
```

4.2.3.2 getTopLeft()

```
Point2D Rectangle::getTopLeft ( ) const
```

Queries the top-left corner.

Returns

The point

Definition at line 34 of file Rectangle.cpp.

```
                                     {
    return topleft;
}
```

4.2.3.3 isEmpty()

```
bool Rectangle::isEmpty ( ) const
```

For a rectangle to be valid this condition must hold topleft.getX() $<=$ bottomright.getX() && topleft.getY() $>=$ bottomright.getY() otherwise it is an empty (incorrect) rectangle.

Returns

Whether the rectangle is empty or not

Definition at line 42 of file Rectangle.cpp.

```
                        {
    return topleft.getX()>bottomright.getX() || topleft.getY() < bottomright.getY();
}
```

4.2.3.4 setGeometry() [1/2]

```
void Rectangle::setGeometry (
            const Point2D & tl,
            const Point2D & br )
```

Initializes the data of the rectangle.

Parameters

tl	Top-left point
br	Bottom-right corner

Definition at line 29 of file Rectangle.cpp.

```
                                                                          {
    topleft = tl;
    bottomright = br;
}
```

4.2.3.5 setGeometry() [2/2]

```
void Rectangle::setGeometry (
            int x,
            int y,
            int w,
            int h )
```

Initializes the data of the rectangle.

Parameters

x	XY Coordinates of top-left corner
y	
w	Width of the rectangle
h	Height of the rectangle

Definition at line 22 of file Rectangle.cpp.

```
                                                      {
    topleft.setX(x);
    topleft.setY(y);
    bottomright.setX(x+w);
    bottomright.setY(y-h);
}
```

4.2.4 Friends And Related Function Documentation

4.2.4.1 doOverlap

```
Rectangle doOverlap (
            const Rectangle & r1,
            const Rectangle & r2 )  [friend]
```

Calculates the rectangle intersection of the two given rectangles. If there is no intersection, an empty rectangle is returned instead.

Parameters

r1	One rectangle
r2	Other rectangle

Returns

The rectangle given by the intersection of `r1` and `r2`

Note

This is an external function to the class Rectangle but since it is also friend, this function is allowed access to private data/methods

Definition at line 59 of file Rectangle.cpp.

```
                                                              {
    Rectangle result;
    Point2D rTL, rBR;
    /* NO FRIEND
        rTL.setX(max(r1.getTopLeft().getX(),r2.getTopLeft().getX()));
        rTL.setY(max(r1.getTopLeft().getY(),r2.getTopLeft().getY()));
        rBR.setX(min(r1.getBottomRight().getX(),r2.getBottomRight().getX()));
        rBR.setY(min(r1.getBottomRight().getY(),r2.getBottomRight().getY()));
     */
    rTL.setX(max(r1.topleft.getX(),r2.topleft.getX()));
    rTL.setY(min(r1.topleft.getY(),r2.topleft.getY()));
    rBR.setX(min(r1.bottomright.getX(),r2.bottomright.getX()));
    rBR.setY(max(r1.bottomright.getY(),r2.bottomright.getY()));
    result.setGeometry(rTL,rBR);
    return result; // Read more
}
```

The documentation for this class was generated from the following files:

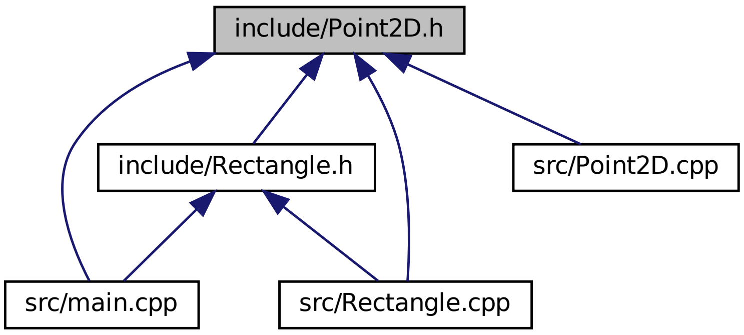
- include/Rectangle.h
- src/Rectangle.cpp

Chapter 5

File Documentation

5.1 include/Point2D.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- class Point2D

  To represent a point in a two-dimensional space.

5.1.1 Detailed Description

Author

lcv

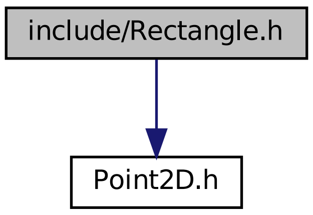
Date

16 de enero de 2020, 20:03

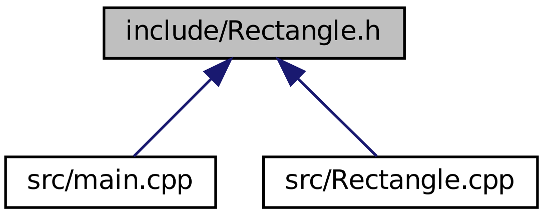
5.2 include/Rectangle.h File Reference

```
#include "Point2D.h"
```

Include dependency graph for Rectangle.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Rectangle

  To represent a rectangle in a two-dimensional space as a pair or points, the top-left corner and the bottom-right one.

Functions

- bool isInside (const Point2D &p, const Rectangle &r)

  Calculates whether a point is internal to a rectangle.

5.2.1 Detailed Description

Author

lcv

Date

16 de enero de 2020, 20:04

5.2.2 Function Documentation

5.2.2.1 isInside()

```
bool isInside (
            const Point2D & p,
            const Rectangle & r )
```

Calculates whether a point is internal to a rectangle.

Parameters

p	The point
r	The rectangle

Returns

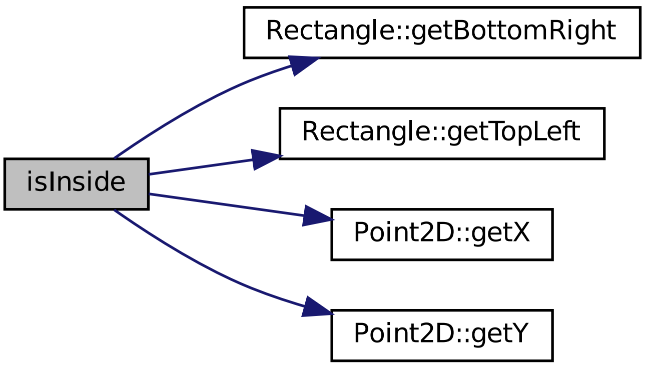
Return values

true	if `p` is inscribed within `r`,
false	otherwise

Definition at line 76 of file Rectangle.cpp.

```
                                                         {
     return r.getTopLeft().getX() <= p.getX() && p.getX()<=r.getBottomRight().getX() &&
            r.getTopLeft().getY() >= p.getY() && p.getY()>=r.getBottomRight().getY();
}
```

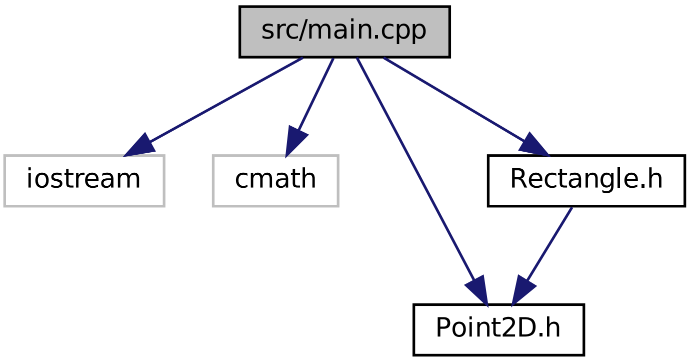
Here is the call graph for this function:



5.3 src/main.cpp File Reference

```
#include <iostream>
#include <cmath>
#include "Point2D.h"
#include "Rectangle.h"
```

Include dependency graph for main.cpp:



Functions

- int main ()

  Main function.

5.3.1 Detailed Description

Author

DECSAI

Note

To be implemented (partially) by students Videotutorial https://drive.google.com/file/d/1-KiBquuuHJ5_zNeSLqWH88PqazuoTVm

5.3.2 Function Documentation

5.3.2.1 main()

```
int main ( )
```

Main function.

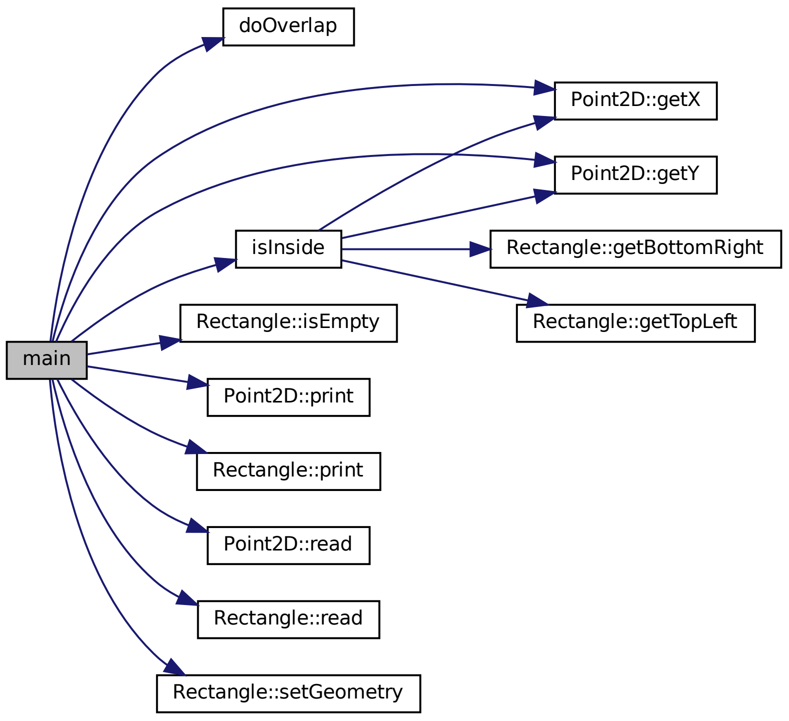
Returns

Always 0

Definition at line 65 of file main.cpp.

```
65         {
66     Rectangle A, B, Intersection;
67     Point2D p;
68     int count;
69 
70 
71 
72     A.setGeometry(2,5,3,3);
73     cout « "First rectangle is ";
74     A.print();
75     cout « endl « "Type second rectangle: ";
76     B.read();
77     cout « endl « "Calculating intersection of: ";
78     A.print();
79     cout « " and ";
80     B.print();
81     cout « endl;
82     Intersection = doOverlap(A,B);
83     if (Intersection.isEmpty()) {
84         cerr « "Empty intersection"«endl;
85     } else {
86         cout « "The intersection is: ";
87         Intersection.print();
88         count = 0;
89         cout « endl « "Reading points...";
90         p.read();
91         while (p.getX()>=0 && p.getY()>=0) {
92             if (isInside(p,Intersection)) {
93                 p.print();
94                 count ++;
95             }
96             p.read();
97         }
98         if (count > 0)
99             cout « " fall within the intersection ("« count«" total)" « endl;
100         else
101             cout « " None of them falls within the intersection "«endl;
102     }
103 
104     return 0;
105 }
```

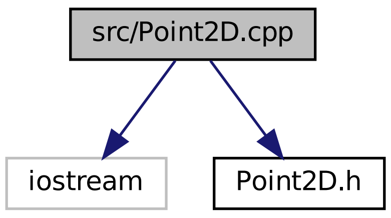
Here is the call graph for this function:



5.4 src/Point2D.cpp File Reference

```
#include <iostream>
#include "Point2D.h"
```

Include dependency graph for Point2D.cpp:



5.4.1 Detailed Description

Author

lcv

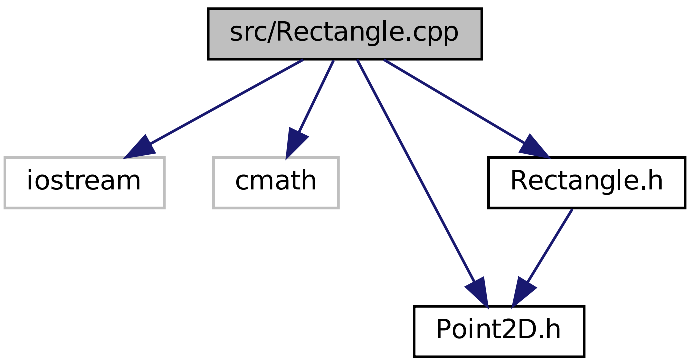
Date

16 de enero de 2020, 20:06

5.5 src/Rectangle.cpp File Reference

```
#include <iostream>
#include <cmath>
#include "Point2D.h"
#include "Rectangle.h"
```

Include dependency graph for Rectangle.cpp:



Functions

- Rectangle doOverlap (const Rectangle &r1, const Rectangle &r2)
- bool isInside (const Point2D &p, const Rectangle &r)

  Calculates whether a point is internal to a rectangle.

5.5.1 Detailed Description

Author

lcv

Date

16 de enero de 2020, 20:06

5.5.2 Function Documentation

5.5.2.1 doOverlap()

```
Rectangle doOverlap (
            const Rectangle & r1,
            const Rectangle & r2 )
```

Parameters

r1	One rectangle
r2	Other rectangle

Returns

The rectangle given by the intersection of `r1` and `r2`

Note

This is an external function to the class Rectangle but since it is also friend, this function is allowed access to private data/methods

Definition at line 59 of file Rectangle.cpp.

```
                                                                    {
       Rectangle result;
       Point2D rTL, rBR;
       /* NO FRIEND
           rTL.setX(max(r1.getTopLeft().getX(),r2.getTopLeft().getX()));
           rTL.setY(max(r1.getTopLeft().getY(),r2.getTopLeft().getY()));
           rBR.setX(min(r1.getBottomRight().getX(),r2.getBottomRight().getX()));
           rBR.setY(min(r1.getBottomRight().getY(),r2.getBottomRight().getY()));
        */
       rTL.setX(max(r1.topleft.getX(),r2.topleft.getX()));
       rTL.setY(min(r1.topleft.getY(),r2.topleft.getY()));
       rBR.setX(min(r1.bottomright.getX(),r2.bottomright.getX()));
       rBR.setY(max(r1.bottomright.getY(),r2.bottomright.getY()));
       result.setGeometry(rTL,rBR);
       return result; // Read more
   }
```

5.5.2.2 isInside()

```
bool isInside (
            const Point2D & p,
            const Rectangle & r )
```

Calculates whether a point is internal to a rectangle.

Parameters

p	The point
r	The rectangle

Returns

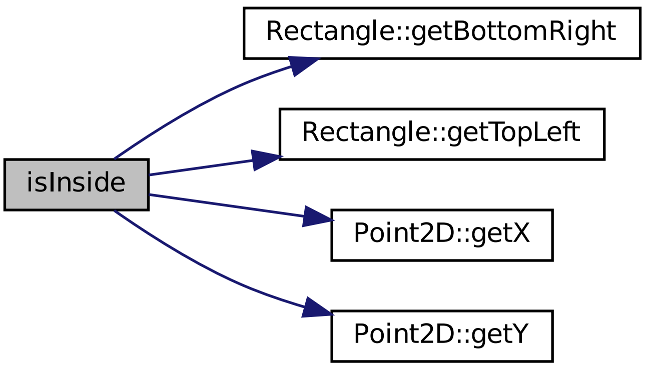
Return values

true	if `p` is inscribed within `r`,
false	otherwise

Definition at line 76 of file Rectangle.cpp.

```
                                                                        {
          return r.getTopLeft().getX() <= p.getX() && p.getX()<=r.getBottomRight().getX() &&
                    r.getTopLeft().getY() >= p.getY() && p.getY()>=r.getBottomRight().getY();
}
```

Here is the call graph for this function:



Index