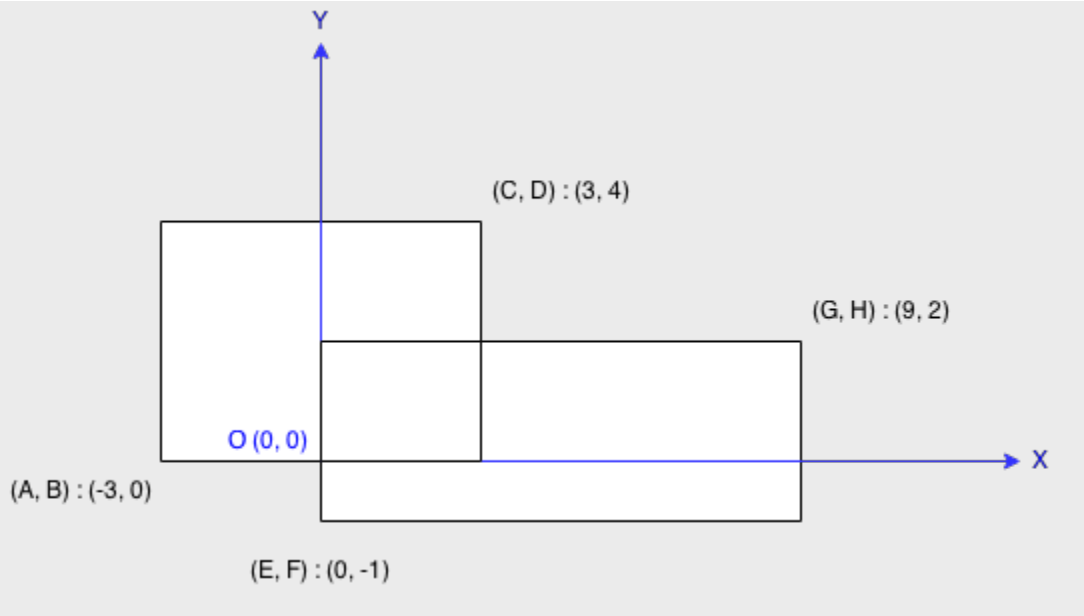
Rectangle Area

Find the total area covered by two **rectilinear** rectangles in a **2D** plane.

Each rectangle is defined by its bottom left corner and top right corner as shown in the figure.



Assume that the total area is never beyond the maximum possible value of **int**.

Credits:
Special thanks to @mithmatt for adding this problem, creating the above image and all test cases.

Solution 1

Instead of checking whether the rectangles overlap, I max `right` with `left` (and `top` with `bottom`). Haven't seen that in other solutions.

```
int computeArea(int A, int B, int C, int D, int E, int F, int G, int H) {
    int left = max(A,E), right = max(min(C,G), left);
    int bottom = max(B,F), top = max(min(D,H), bottom);
    return (C-A)*(D-B) - (right-left)*(top-bottom) + (G-E)*(H-F);
}
```

written by ManuelP original link here

Solution 2

```
public int computeArea(int A, int B, int C, int D, int E, int F, int G, int H) {

        int areaOfSqrA = (C-A) * (D-B);
         int areaOfSqrB = (G-E) * (H-F);

        int left = Math.max(A, E);
        int right = Math.min(G, C);
        int bottom = Math.max(F, B);
        int top = Math.min(D, H);

        //If overlap
        int overlap = 0;
        if(right > left && top > bottom)
             overlap = (right - left) * (top - bottom);

        return areaOfSqrA + areaOfSqrB - overlap;
    }
```

Hello! So, the code should be fairly straightforward. I first calculate the area of each rectangle and then calculate the overlapping area between the two rectangles (if there is one!). At the end, we sum up the individual areas and subtract the overlapping area/o !

Feel free to ask should you have any queries for me OR if my solution can be improved upon! :)

written by waisuan original link here

Solution 3

```
This is utterly ridiculous. As I was writing this I knew I was going about it wro
ng but I wanted to finish it before I thought of a different method or looked at
any other solutions.

  int computeArea(int A, int B, int C, int D, int E, int F, int G, int H) {
    int area1 = (D - B)*(C - A);
    int area2 = (H - F)*(G - E);
    int area3;
    if (area1 == 0) {
        return area2;
    }
    if (area2 == 0) {
        return area1;
    }
    if ((A == D) && (B == F) && (C == G) && (D == H)) {
        return area1;
    }
    if ((E >= C) | (G <= A) | (H <= B) | (D <= F)) {    //not overlapping
        return (area1 + area2);
    }
    if (((G - E) <= (C - A)) && ((H - F) <= (D - B)) && (E >= A) && (F >= B) && (G
<= C) && (D >= H)) {                        //rect2 is inside rect1
        return area1;
    }
    if (((C - A) <= (G - E)) && ((D - B) <= (H - F)) && (E <= A) && (B >= F) && (G
>= C) && (H >= D)) {                        //rect1 is inside rect2
        return area2;
    }
    if ((F >= B) && (E >= A) && (G >= C) && (H >= D)) {                         //ov
erlapping upper right corner
        area3 = (C - E)*(D - F);
    }
    else if ((F >= B) && (E <= A) && (G <= C) && (H >= D)) {
//overlapping upper left corner
        area3 = (G - A)*(D - F);
    }
    else if ((F <= B) && (E <= A) && (G <= C) && (H <= D)) {
//overlapping bottom left corner
        area3 = (G - A)*(H - B);
    }
    else if ((F <= B) && (E >= A) && (G >= C) && (H <= D)) {
//overlapping bottom right corner
        area3 = (H - B)*(C - E);
    }
    else if (((C - A) <= (G - E)) && (H <= D) && (G >= C) && (E <= A) && (F <= B))
{               //overlapping bottom side
        area3 = (C - A)*(H - B);
    }
    else if (((C - A) <= (G - E)) && (H >= D) && (G >= C) && (E <= A) && (F >= B))
{               //overlapping top side
        area3 = (C - A)*(D - F);
    }
    else if (((D - B) <= (H - F)) && (E <= A) && (F <= B) && (H >= D) && (G <= C))
{               //overlapping left side
```

```
        area3 = (G - A)*(D - B);
    }
    else if (((D - B) <= (H - F)) && (E >= A) && (F <= B) && (H >= D) && (G >= C))
{               //overlapping right side
        area3 = (C - E)*(D - B);
    }
    else if (((C - A) >= (G - E)) && (E >= A) && (F >= B) && (C >= G) && (D <= H))
{      //overlapping part of top side
        area3 = (G - E)*(D - F);
    }
    else if (((C - A) >= (G - E)) && (A <= E) && (B >= F) && (G <= C) && (D >= H))
{       //overlapping part of bottom side
        area3 = (G - E)*(H - B);
    }
    else if (((D - B) >= (H - F)) && (E <= A) && (F >= B) && (G <= C) && (H <= D))
{      //overlapping part of left side
        area3 = (G - A)*(H - F);
    }
    else if (((D - B) >= (H - F)) && (E >= A) && (F >= B) && (G >= C) && (H <= D))
{       //overlapping part of right side
        area3 = (C - E)*(H - F);
    }
    else if (((G - E) <= (C - A)) && (E >= A) && (F <= B) && (G <= C) && (H >= D))
{     //overlapping top and bottom
        area3 = (G - E)*(D - B);
    }
    else if (((H - F) <= (D - B)) && (E <= A) && (F >= B) && (C <= G) && (D >= H))
{     //overlapping left and right
        area3 = (C - A)*(H - F);
    }

    return (area1 + area2 - area3);
}
```

written by bigDeeOT original link here

From Leetcoder.