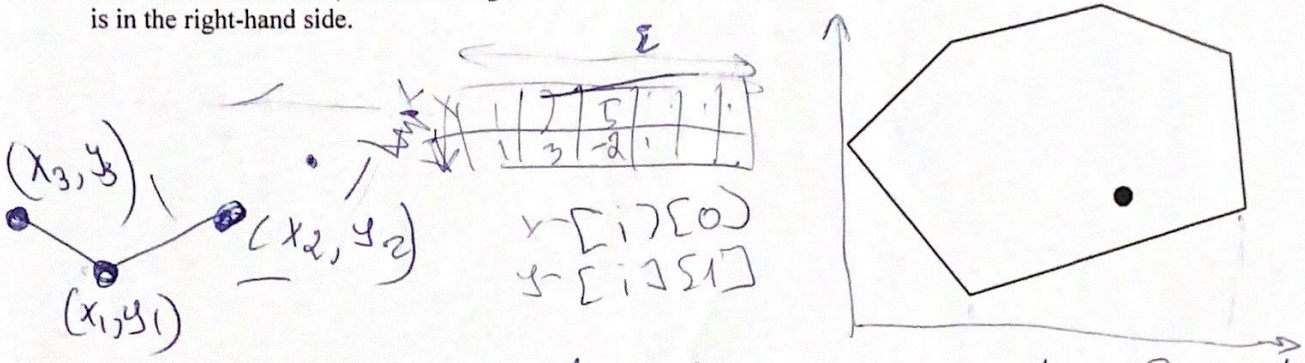
Problem 3: Write a Java function ***public static boolean isInside(double[][] vertex, double x, double y)*** that takes as its argument a ***2-by-n*** array of a convex polygon's vertex coordinates ***double[][] vertex*** – the *x* coordinates in the first row and *y* coordinates in the second row, and ***double x*** and ***double y*** coordinates of a point. It checks, if the point is inside the polygon.

Assume and use a method ***boolean toLeft(double x1, double y1, double x2, double y2, double x0, double y0)*** that takes as its arguments coordinates of three points and returns ***true***, if the third point ***(x0, y0)*** is in the left-hand side, when moving from the first point ***(x1, y1)*** to the second one ***(x2, y2)***; and ***false***, if it is in the right-hand side.




```
public static boolean isInside(double[][] vertex, double x, double y){
    double x1, y1, x2, y2, x3, y3;
    boolean result = true;
    for(int i=0; i < vertex[0].length; i++){
        x1 = vertex[0][i];
        y1 = vertex[1][i];
        if(i+1 == length){ x2 = vertex[0][0]; y2 = vertex[1][0]; }
        else { x2 = [0][i+1]; y2 = [1][i+1]; }
        if(i-1 < 0){ x3 = vertex[0][vertex[0].length-1]; y3 = vertex[1][vertex[0].length-1]; }
        else { x3 = vertex[0][i-1]; y3 = vertex[1][i-1]; }
        angleLeft = toLeft(x1, y1, x2, y2); 6 args needed
        angleRight = !toLeft(x1, y1, x3, y3);
        if(!angleLeft || !angleRight){
            result = false;
            break;
        }
    }
    return result;
}
```

Use the backside, if needed

2

OOP.MT.17031J.H014