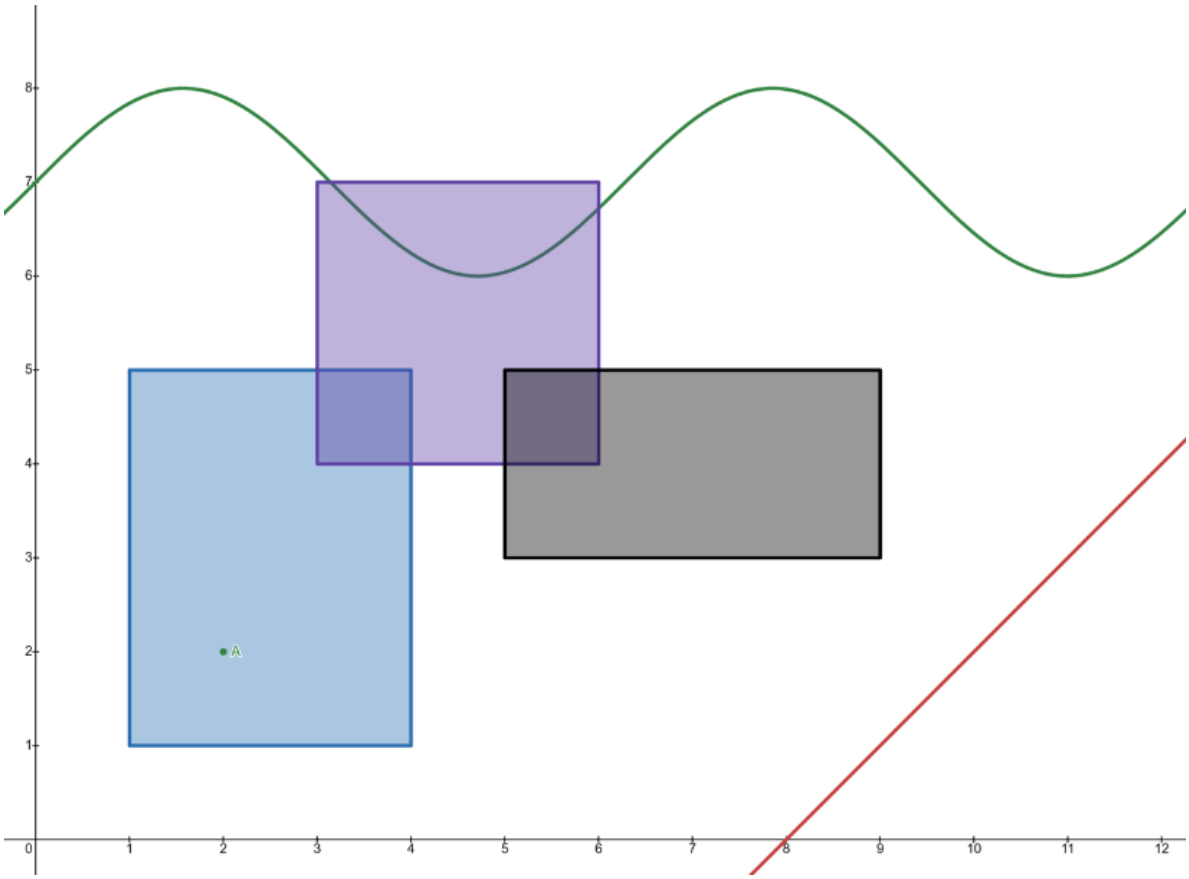
CSE 1322L - Assignment 1 (Spring 2026)

Introduction

Nowadays, whenever someone uses a computer, they do so either through their keyboard, mouse, or by tapping on a touchscreen. In doing so yourself, you may have wondered: "how does the computer know if I've hit or missed a specific button?". The answer is simple: through the use of geometry.

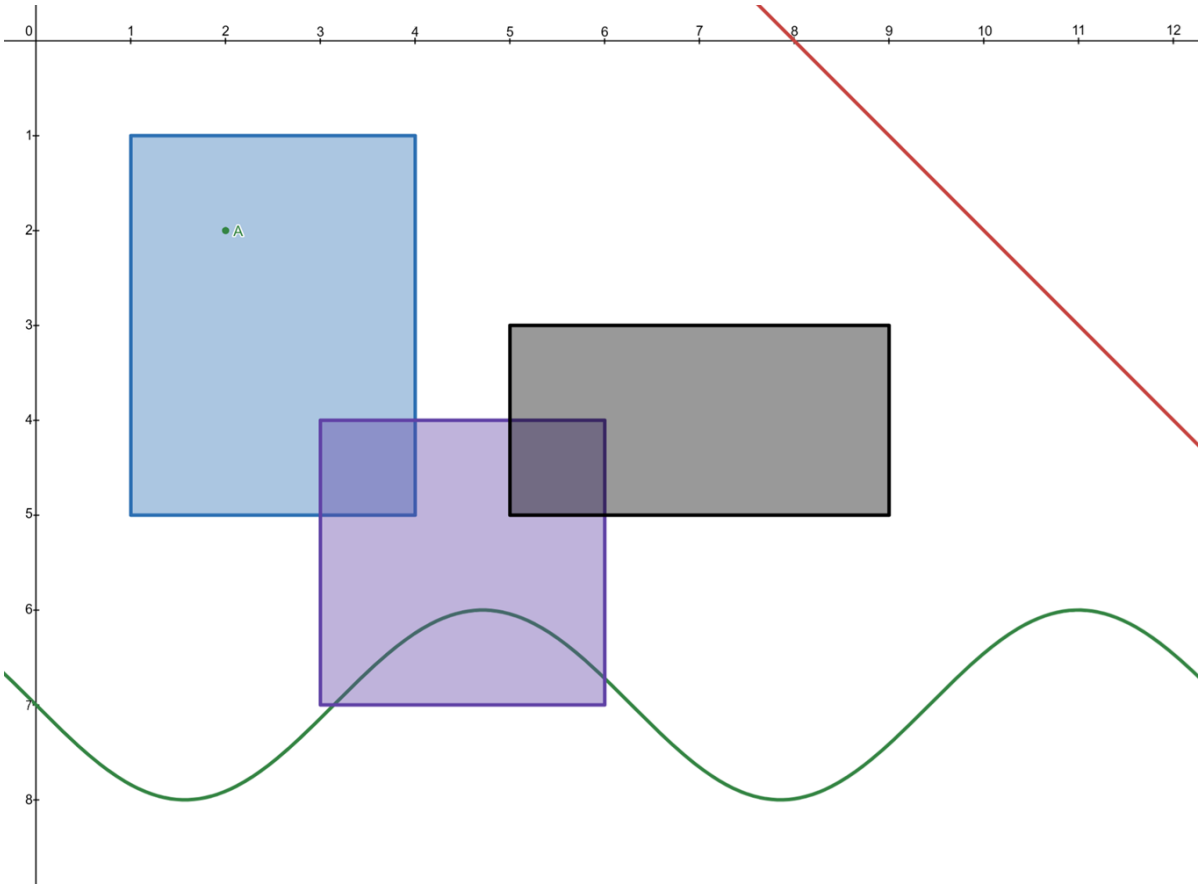
You may remember the cartesian plane from your geometry classes. In this plane, you can specify points, lines, segments, areas, and many other concepts using formulas or coordinates.



Your device's screen works the same way, with some key differences:

- The origin (0,0) is usually at the **top-left**
- The y-axis grows **downwards** instead of upwards

Mathematically speaking, your computer screen's coordinate system is like the Cartesian plane's if its y-axis were inverted along the x-axis. We are also usually only interested in the 1st quadrant:



Of interest to us are points and areas:

- **Points**: a pair of coordinates (x, y). Your mouse is a point on the screen
- **Areas**: every control you see on your screen (e.g.: buttons) has an area associated with it. Clicking in that area interacts with the control. These areas are usually rectangular, their left and right edges (sides) are parallel to the y-axis, and their top and bottom edges are parallel to the x-axis.

While you may think that an area is best described by specifying the 4 points on its corners (vertices), it is more expedient for the computer to describe an area by specifying the coordinates of its origin, its width, and its height. This way, the computer only has to remember 4 numbers (x, y, width, height) instead of 8 (the x and y coordinates of its 4 corners). This also allows us to determine the edges of a specific area:

- An area's top edge is its y-coordinate
- An area's left edge is its x-coordinate
- An area's bottom edge is the sum of its y-coordinate and its height
- An area's right edge is the sum of its x-coordinate and its width

For example: in the previous picture, the black area has coordinates (5,3), width 4, and height 2. Its top edge is at 3, left edge is at 5, right edge is at 9 (5 + 4), and bottom edge at 5 (3 + 2).

To determine whether a point is inside an area, **all** of the following must be true:

- The point must be at or below the area's top edge
- The point must be at or above the area's bottom edge
- The point must be at or to the right of the area's left edge
- The point must be at or to the left of the area's right edge

You can check this by looking at Point A in the previous picture. All of the above bullet points are satisfied relative to the blue area, but not to the purple or black areas.

Lastly, it is sometimes necessary to determine if two areas have any overlap between them. To do this, **all** the following must be true:

- Area A's left edge must be at or to the left of Area B's right edge
- Area A's right edge must be at or to the right of Area B's left edge
- Area A's top edge must be at or above Area B's bottom edge
- Area A's bottom edge must be at or below Area B's top edge

You can, once again, verify that all of the above is true between the blue and purple areas, and the purple and black areas, but not between the blue and black areas.

Requirements

Write a program which allows the user to store the coordinates of a single point, as well as the coordinates, width, and height of 2 rectangular areas. The program must also allow the user to check if the point is contained inside either area, if the two areas intersect, and to view the currently stored information on the point and areas.

At the start of your program, create enough variables to store the following:

- The x and y coordinates of a point
- The x and y coordinates of the top-leftmost point of 2 different areas
- The width and the height of 2 areas
- All the above variables must be **whole numbers initialized at 0**

In a loop, implement the menu options below:

1. **Enter Point**: prompts the user for the x and y coordinate of the point, updating the appropriate variables above
2. **Enter Area 1**: prompts the user for the x and y coordinates of one the areas, as well as its width and height. Update the appropriate variables above.
3. **Enter Area 2**: prompts the user for the x and y coordinates of the other area, as well as its width and height. Update the appropriate variables above.

4. **Point-Area Intersection**: Ask the user if they want to use Area 1 or 2, then use that area to determine if the point is inside it or not. Print a message stating whether the point is in the selected area or not.
5. **Area-Area Intersection**: If the two areas have an area in common, print "The two areas intersect.". Otherwise, print "The two areas do NOT intersect.".
6. **View info**: Prints the following, replacing the curly braces and its contents with the appropriate variable:

```
Point x: {point_x}, y: {point_y}

Area 1 x: {area_1_x}, y: {area_1_y}, width: {area_1_width}, height: {area_1_height}

Area 2 x: {area_2_x}, y: {area_2_y}, width: {area_2_width}, height: {area_2_height}
```

7. **Quit**: Terminates the program

You can make the following assumptions about your program:

- The user will never enter a negative number
- The user will not try to select a menu option that does not exist
- When selecting menu option 4, the user will not try to use a non-existing area

Deliverables

- Assignment1.java (driver)

Considerations

- You will get partial credit for partial work, as long as the rubric permits it. Even if you can't do everything, try to do as much as possible.
- Remember to name your variables sensibly. You will need 10 variables to store the information specified at the start of "Requirements". Using a single letter to name your variables will just make it harder for you to write your program and for the grader to read it. Instead, prefer to use a descriptive name such as "area1_width"
- Remember that you are dealing with screen coordinates, not the cartesian plane
    - Your origin is at the top left
    - Your x-axis grows to the right
    - Your y-axis grown downwards
- If Area A and Area B intersect, that means there is a third area which both areas share, Area C. How could you determine the coordinates, width, and height of Area C?

Sample Output (user input in red)

```
[Intersection Detector]

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 6

Point x: 0, y: 0
Area 1 x: 0, y: 0 width: 0, height: 0
Area 2 x: 0, y: 0 width: 0, height: 0

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 1
Enter x-coordinate of Point: 3
Enter y-coordinate of Point: 4

Point coordinates updated.

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 2
Enter x-coordinate of Area 1: 1
Enter y-coordinate of Area 1: 2
Enter width of Area 1: 2
Enter height of Area 1: 5
```

```
Area 1 updated

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 3
Enter x-coordinate of Area 2: 4
Enter y-coordinate of Area 2: 9
Enter width of Area 2: 5
Enter height of Area 2: 3

Area 2 updated

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 6

Point x: 3, y: 4
Area 1 x: 1, y: 2 width: 2, height: 5
Area 2 x: 4, y: 9 width: 5, height: 3

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 4
Use Area 1 or Area 2? 1

The point is in Area 1.
```

```
1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 4
Use Area 1 or Area 2? 2

The point is NOT in Area 2.

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 5
The two areas do NOT intersect.

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 3
Enter x-coordinate of Area 2: 2
Enter y-coordinate of Area 2: 3
Enter width of Area 2: 5
Enter height of Area 2: 3

Area 2 updated

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
```

```
6. View info
7. Quit
Select option: 5
The two areas intersect.

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 4
Use Area 1 or Area 2? 2

The point is in Area 2.

1. Enter Point
2. Enter Area 1
3. Enter Area 2
4. Point-Area Intersection
5. Area-Area Intersection
6. View info
7. Quit
Select option: 7
Shutting off...
```