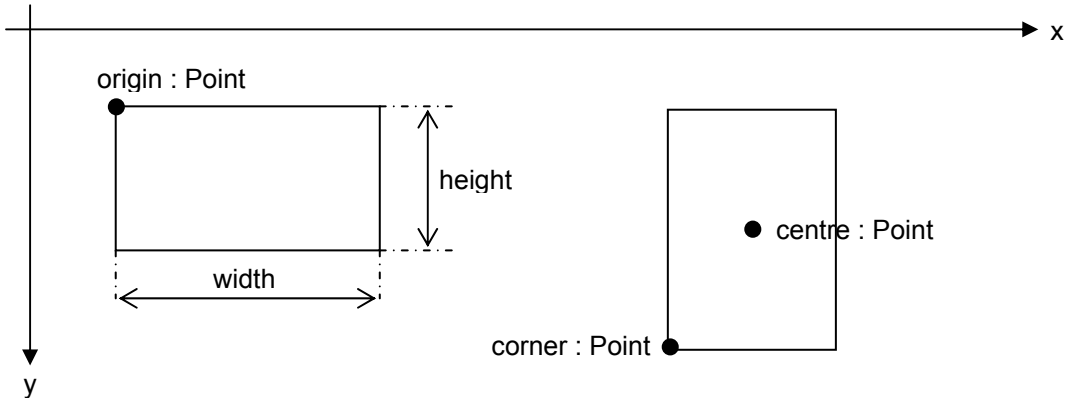
Arbeitsblatt 3, infT

Enwerfen von Klassen in C++
Die Klassen **`Point`** und **`Rectangle`** sind zu entwerfen.

`Point` modelliere einen Punkt in der zwei-dimensionalen Ebene. Die Daten-Member sind **`private`** vorzusehen. **`Point`** gewähre der Klasse **`Rectangle`** via **`friend`** direkten Zugriff auf seine Daten-Member.

`Rectangle` modelliere ein Rechteck in der zwei-dimensionalen Ebene. Die Kanten eines Rechteck seien stets parallel zu den Koordinatensystem-Achsen ausgerichtet.

Situation und Konstruktoren:



Stellen Sie für **`Point`** einen Konstruktor bereit
- mit Defaultargumenten für einen Punkt im Ursprung des Koordinatensystems,
- mit zwei Argumenten für **`x`** und **`y`**.

Ein **`Point`** soll mit **`print`** seine Koordinaten an **`cout`** bekannt geben.

Stellen Sie für **`Rectangle`** zwei Konstruktoren bereit mit Default-Argumenten. Ohne Argumente beim Instantiieren soll ein Einheits-Rechteck erzeugt werden: Kantenlängen 1, Origin im Ursprung des Koordinatensystems. Obiges Bild zeigt, welche Argumente die Konstruktoren zu berücksichtigen haben. Stellen Sie auch hier eine Funktion **`print`** bereit.

Testen Sie das Erstellen (und Zerstören) einiger Rechtecke auf dem Stack, dann auch auf dem Heap. Sehen Sie dazu in den Konstruktoren und den Destruktoren kleine Meldungen für **`cout`** vor. Formulieren Sie die Funktionen unter korrektem Einsatz von **`const`**, möglichst mit Parameter-Übergabe *by (const) reference* und unter Verwendung von Konstruktor-Initialisierlisten.

Testen Sie zum Schluss auch folgenden stand-alone-Block in **`main`**:

```
{
  cout << "Aggregate demo" << endl;
  cout << "==============" << endl;
  const int len = 6;
  Rectangle rarr[len] = {Rectangle()};
}
```