Providing Controlled Access: Getters and Setters

- **getters** and **setters** are what we name methods, which are used to access and mutate attributes (instead of direct access)
- they are also known as **accessors** and **mutators**
- often the getter is simply **passing** the underlying attribute (Please: No side effects!), and the setter is ensuring that all required invariants hold true
- it is important to think about whether it's worth exposing a setter in the first place – is it really required?
- for instance, in our example, setting **noSides** directly would seem like a fundamentally bad idea

this getter method can be used as an alternative to access the underlying attribute points – in this basic case it simply passes on the attribute itself

```
class Polygon {
  Position[] points;
  int noSides;

  Polygon(Position[] points) {
    this.points = points;
    noSides = points.length;
  }

  Position[] getPoints() {
    return points;
  }

  void setPoints(Position[] points) {
    this.points = points;
    noSides = points.length;
} }
```

this setter method allows the mutation of the points field and, most importantly, make sure that the overall state is consistent and our invariant 'points.length == noSides' holds true after the manipulation