TS: Type Guards and Narrowing Exercises

Exercise 1: Calculating the Area of Geometric Shapes

Objective: Practice type guards to calculate the area of various geometric shapes.

Task:

1. Define an interface `Shape` for `Circle`, `Square`, and `Rectangle`. Each shape should have a `kind` property and respective properties for their dimensions.
2. Implement a function `getArea` that takes a `Shape` and returns its area. Use type guards to narrow down the shape type and calculate the area accordingly.

Exercise 2: Validating Coordinates in 2D Space

Objective: Use type guards to validate points in 2D space.

Task:

1. Create an interface `Point` that represents a point in 2D space with `x` and `y` coordinates.
2. Implement a function `isPoint` that checks if a given object is a `Point`.
3. Use this function to filter out valid points from an array of objects.

Exercise 3: Determining Shape Perimeters

Objective: Calculate the perimeter of various geometric shapes using type guards.

Task:

1. Extend the `Shape` interface from Exercise 1 to include `Triangle`.
2. Implement a function `getPerimeter` that calculates the perimeter of each shape using type guards.

Exercise 4: Classifying Spatial Objects

Objective: Classify spatial objects using the `in` operator for type narrowing.

Task:

1. Define interfaces `Point`, `Line`, and `Polygon`.
2. Implement a function `classifyObject` that takes an object and returns a string describing whether it's a point, line, or polygon using the `in` operator for type narrowing.

Exercise 5[1]: Finding Intersections in 3D Space

Objective: Find intersections in 3D space using `instanceof` for type narrowing.

Task:

1. Create interfaces `Point3D` and `Line3D`.
2. Implement a function `findIntersections` that takes two spatial objects and returns an array of intersection points if they intersect. Use `instanceof` to narrow down the types.

[1] Requires linear algebra