# TS: Everyday Types

## Exercise 1: Basic Primitives and Variables

**Objective:** Understand basic TypeScript syntax and type checking with primitives.

**Task:**

1. Create variables to represent a point in a 2D coordinate system using the `number` type.
2. Create a variable to represent a label for the point using the `string` type.
3. Create a `boolean` variable to check if the point is within a specified boundary.

## Exercise 2: Arrays

**Objective:** Learn how to work with arrays in TypeScript.

**Task:**

1. Create an array of numbers to represent a series of x-coordinates.
2. Create an array of strings to label each corresponding point.
3. Create a function that takes an array of x-coordinates and returns the sum of all coordinates.

## Exercise 3: Object Types and Optional Properties

**Objective:** Understand how to handle object types and optional properties in TypeScript.

**Task:**

1. Define an object type for a 2D point that includes optional labels.
2. Create a function that prints the coordinates and labels if available.

## Exercise 4: Union Types and Type Narrowing

**Objective:** Learn how to work with union types and type narrowing in TypeScript.

**Task:**

1. Create a union type that can represent either a 2D point or a label.
2. Write a function that takes this union type and prints either the coordinates or the label based on the type.

## Exercise 5: Type Aliases and Interfaces

**Objective:** Understand how to use type aliases and interfaces in TypeScript.

**Task:**

1. Create a type alias for a 2D point.
2. Create an interface for a labeled point that extends the 2D point.
3. Write a function that accepts either a 2D point or a labeled point and prints the details.

## Exercise 6: Literal Types

**Objective:** Learn how to work with literal types in TypeScript.

**Task:**

1. Create a literal type for the possible labels of points.
2. Write a function that accepts only these specific labels.

## Exercise 7: Enums

**Objective:** Understand how to use enums in TypeScript.

**Task:**

1. Create an enum for the quadrants of a 2D coordinate system.
2. Write a function that takes a point and returns the quadrant it belongs to.

## Exercise 8: Nullable Types

**Objective:** Learn how to handle nullable types in TypeScript.

**Task:**

1. Create a function that takes a point and a nullable label.
2. Ensure the function handles the nullable label safely.