Sure, I can provide you with 10 basic TypeScript programs along with instructions on how to run them. To run TypeScript programs, you need to have Node.js and TypeScript installed on your machine.

Setting Up Your Environment:

1. **Install Node.js**:
   - Download and install Node.js from nodejs.org.

2. **Install TypeScript**:
   - Open a terminal (command prompt, terminal, etc.) and run the following command to install TypeScript globally:
   ```shell
   npm install -g typescript
   ```

3. **Create a Directory**:
   - Create a new directory for your TypeScript projects:
   ```shell
   mkdir typescript-projects
   cd typescript-projects
   ```

4. **Create a TypeScript File**:
   - Create a new file with a `.ts` extension for each of the programs.

5. **Compile and Run TypeScript Programs**:
   - To compile a TypeScript file to JavaScript, run:
   ```shell
   tsc filename.ts
   ```
   - This will generate a JavaScript file (`filename.js`) in the same directory.
   - To run the compiled JavaScript file, use Node.js:
   ```shell
   node filename.js
   ```

Programs:

1. **Hello World**:
   ```typescript
   console.log("Hello, TypeScript!");
   ```

2. **Type Annotations**:
   ```typescript
   let name: string = "Alice";
   let age: number = 30;
   console.log(`Name: ${name}, Age: ${age}`);
   ```

3. **Function**:
   ```typescript
   function add(x: number, y: number): number {
       return x + y;
   }

   console.log(add(5, 3)); // Output: 8
   ```

4. **Interfaces**:
   ```typescript
   interface Person {
       name: string;
       age: number;
   }

   const person: Person = { name: "Alice", age: 30 };
   console.log(`Name: ${person.name}, Age: ${person.age}`);
   ```

5. **Classes**:
   ```typescript
   class Car {
       model: string;

       constructor(model: string) {
           this.model = model;
       }

       drive() {
           console.log(`Driving a ${this.model}`);
       }
   }

   const myCar = new Car("Toyota");
   myCar.drive(); // Output: Driving a Toyota
   ```

6. **Generics**:
   ```typescript
   function reverseArray<T>(arr: T[]): T[] {
     return arr.reverse();
   }

   const numbers = [1, 2, 3, 4];
   const reversedNumbers = reverseArray(numbers);
   console.log(reversedNumbers);
   ```

7. **Union Types**:
   ```typescript
   let value: string | number;

   value = "Hello";
   console.log(value); // Output: Hello

   value = 42;
   console.log(value); // Output: 42
   ```

8. **Intersection Types**:
   ```typescript
   interface Bird {
     fly(): void;
   }

   interface Fish {
     swim(): void;
   }

   type FlyingFish = Bird & Fish;

   const flyingFish: FlyingFish = {
     fly: () => console.log("Flying"),
     swim: () => console.log("Swimming")
   };

   flyingFish.fly();
   flyingFish.swim();
   ```

9. **Type Assertions**:
   ```typescript
   let value: any = "This is a string";
   let strLength: number = (value as string).length;
   console.log(`Length of string: ${strLength}`);
   ```

10. **Enum**:
   ```typescript
   enum Color {
       Red,
       Green,
       Blue
   }

   let favoriteColor: Color = Color.Green;
   console.log(`My favorite color is: ${Color[favoriteColor]}`);
   ```

How to Run the Programs:

1. **Create a new file**:
   - Create a new file with a `.ts` extension (e.g., `hello-world.ts`) for each program.

2. **Copy and Paste the Code**:
   - Copy and paste the code for each program into the corresponding `.ts` file.

3. **Compile the TypeScript File**:
   - In your terminal, navigate to the directory where the `.ts` files are located.
   - Run the following command to compile the TypeScript file to JavaScript:
   ```shell
   tsc hello-world.ts
   ```

4. **Run the Compiled JavaScript File**:
   - After compilation, a `.js` file (e.g., `hello-world.js`) will be generated in the same directory.
   - Run the compiled JavaScript file using Node.js:
   ```shell
   node hello-world.js
   ```

You can repeat these steps for each of the programs provided. This way, you can practice writing TypeScript code and running it to see the output.