typescript generics

TypeScript Generics is a tool that provides a way to create **reusable** components. It creates a component that can work with a **variety of data types** rather than a single data type. It allows users to consume these components and use their own types. Generics ensures that the program is flexible as well as scalable in the long term.

Advantage of Generics

1. **Type-safety:** We can hold only a single type of object in generics. It doesn't allow to store of other objects.
2. **Typecasting is not required:** There is no need to typecast the object.
3. **Compile-Time Checking:** It is checked at compile time so the problem will not occur at runtime.

Possible Generics:

- Functions
- Classes
- Methods
- Variables

Example

The below example helps us to understand the generics clearly.

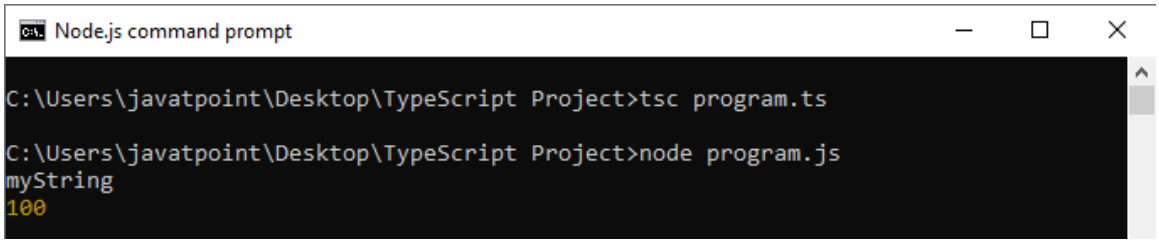
```
function identity<T>(arg: T): T {
    return arg;
}
let output1 = identity<string>("myString");
let output2 = identity<number>( 100 );
console.log(output1);
```

```
7. console.log(output2);
```

When we compile the above file, it returns the corresponding JavaScript file as below.

```
1. function identity(arg) {
2.     return arg;
3. }
4. var output1 = identity("myString");
5. var output2 = identity(100);
6. console.log(output1);
7. console.log(output2);
```

Output:



References:

https://www.javatpoint.com/typescript-generics