Chatper 2 – Functions

1. Introduction
   1. type flexibility often allows functions to be invoked with unexpected argument types
   2. Even when this doesn’t result in thrown errors, there can be negative consequences

|  |
| --- |
| function printLengthOfText(text) {  console.log(text.length);  }    printLengthOfText(3); // Prints: undefined |

Exercise

|  |
| --- |
| function printOperations(a, b) {  if (typeof a !== 'number' || typeof b !== 'number') {  throw new Error('Both arguments must be numbers!');  }  console.log(a + b, a / b);  }  // The function call below should print: 12 1  printOperations(6, 6);  function exclaim(name, count) {  for (let i = 0; i < count; i += 1) {  console.log(`${name}!`);  }  }  // Exclaim 'Muriel!' six times  exclaim('Muriel',6); |

1. Parameter Type Annotations
   1. parameters may be given type annotations with the same syntax as variable declarations: a colon next to the name

|  |
| --- |
| function greet(name: string) {  console.log(`Hello, ${name}!`);  }    greet('Katz'); // Prints: Hello, Katz    greet(1337); // Error: argument '1337' is not assignable to parameter of type 'string' |

* 1. Parameters that we do not provide type annotations for are assumed to be of type any—the same way variables are.

|  |
| --- |
| function printKeyValue(key: string, value) {  console.log(`${key}: ${value}`);  }    printKeyValue('Courage', 1337); // Prints: Courage: 1337  printKeyValue('Mood', 'scared'); // Prints: Mood: scared |

Exercise

|  |
| --- |
| function triple(value: number) {  return value \* 3;  }  function greetTripled(greeting: string, value: number) {  console.log(`${greeting}, ${triple(value)}`!);  }  greetTripled('Hiya', 5); |