

```
const student = {  
  name: "shradha",  
  age: 23,  
  eng: 95,  
  math: 93,  
  phy: 97,  I  
  getAvg() {  
    let avg = (eng + math + phy) / 3;  
    console.log(avg);  
  }  
}
```

```
> student.getAvg();
```

✖ ▶ Uncaught ReferenceError: eng is not defined
at Object.getAvg (app.js:8:19)
at <anonymous>:1:9

JS app.js > [🔗] student

```
1  const student = {  
2      name: "shradha",  
3      age: 23,  
4      eng: 95,  
5      math: 93,  
6      phy: 97,    }  
7      getAvg() {  
8          let avg = (this.eng + this.math + this.phy) / 3;  
9          console.log(avg);  
10     }  
11 }
```


```
> student.getAvg();
```

```
95
```

```
< undefined
```

this Keyword

“This” keyword refers to an object that is executing the current piece of code.

JS app.js >  student >  getAvg

```
1  const student = {  
2      name: "shradha",  
3      age: 23,  
4      eng: 95,  
5      math: 93,  
6      phy: 97,  
7      getAvg() {  
8          let avg = (this.eng + this.math + this.phy) / 3;  
9          console.log(`${this.name} got avg marks = ${avg}`);  
10     }  
11 }
```

try & catch

The try statement allows you to define a block of code to be tested for errors while it is being executed.

The catch statement allows you to define a block of code to be executed, if an error occurs in the try block.

```
try {  
    console.log(a);  
} catch {  
    console.log("variable a doesn't  
}
```

```
const student = {  
  name: "shradha",  
  age: 23,  
  eng: 95,  
  math: 93,  
  phy: 97,  
  getAvg() {  
    console.log(this);  
    let avg = (this.eng + this.math + this.phy) / 3;  
    console.log(`${this.name} got avg marks = ${avg}`);  
  }  
}
```


JS app.js

```
1 console.log("hello");  
2 console.log("hello");  
3 console.log(a);  
4 console.log("hello2");  
5 console.log("hello2");  
6 console.log("hello2");
```

hello





hello

✖ ▶ Uncaught ReferenceError: a is not defined
at app.js:3:13

>

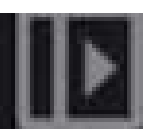
JS app.js

```
1  console.log("hello");
2  console.log("hello");
3  try {
4      console.log(a);
5  } catch {
6      console.log("caught an error.. a is not defined")
7  }
8
9  console.log("hello2");
10 console.log("hello2");
11 console.log("hello2");
```

		top ▼		Filter	Default levels ▼	No Issues	
hello						app.js:1	
hello						app.js:2	
caught an error.. a is not defined						app.js:6	
hello2						app.js:9	
hello2						app.js:10	
hello2						app.js:11	

JS app.js > ...

```
1 console.log("hello");
2 console.log("hello");
3 let a = 5;
4 try {
5     console.log(a);
6 } catch {
7     console.log("caught an error.. a is not defined");
8 }
9
10 console.log("hello2");
11 console.log("hello2");
12 console.log("hello2");
```



top ▼



File

hello

hello

5

hello2

hello2

hello2

> |

JS app.js > ...

```
1  console.log("hello");
2  console.log("hello");
3  // let a = 5;
4  try {
5      console.log(a);
6  } catch(err) {
7      console.log("caught an error.. a is not defined");
8      console.log(err);
9  }
10
11 console.log("hello2");
12 console.log("hello2");
13 console.log("hello2");
```

 top ▼  Filter

hello

hello

caught an error.. a is not defined

ReferenceError: a is not defined
at app.js:5:17

hello2

hello2

hello2

> |

Arrow Functions

const func = (arg1, arg2 ..) => { function definition }

```
const sum = (a, b) => {  
  console.log(a+b);  
}
```

```
> sum(2, 3);
```

```
5
```

```
< undefined
```

```
> sum
```

```
< (a, b) => {  
  console.log(a + b);  
}
```

```
>
```

this Keyword

“This” keyword refers to an object that is executing the current piece of code.

```
> cube(2);
```

```
< 8
```

```
> let c = cube(2);
```

```
< undefined
```

```
> c
```

```
< 8
```

```
const pow = (a, b) => {  
  return a**b  
}
```

```
const hello = () => {  
  console.log("hello world")  
}
```

Arrow Functions

Implicit return

const func = (arg1, arg2 ..) => { value }

```
const mul = (a, b) => (  
  a * b  
);
```

```
const mul = (a,b) => (a*b);|
```



```
const sum = (a, b) => a + b;
```

```
> mul(2, 3);
```

```
< 6
```

```
> mul(2, 19);
```

```
< 38
```

```
> sum(4, 5);
```

```
< 9
```

```
> cube(2);
```

```
< 8
```

Set Timeout

setTimeout(function, timeout)

```
console.log("hi there!");  
  
setTimeout( ()=> {  
  |   console.log("Apna College");  
}, 4000);  
  
console.log("welcome to");
```

Set Timeout

`setTimeout(function, timeout)`

↑ ↑
callback time (ms)

Hi there!

Welcome to

Apna College

Set Interval

setInterval(function, timeout)

```
setInterval( () => {  
  console.log("Apna College");  
}, 2000);
```

clearInterval(id)


```
setInterval(() => {  
  console.log("Apna College");  
}, 2000);
```

Hi there!

3 Apna College

this with Arrow Functions

Arrow
1) lexical scope
 ↓
parent → call
 scope

function
1) scope → this → calling object

this with Arrow Functions

func
↓
obj. func
Student. get Name
this = student

arrow func
↓
obj. func
↓
window
this

```
const student = {  
  name: "aman",  
  marks: 95,  
  prop: this, //global scope  
  getName: function () {  
    console.log(this);  
    return this.name;  
  },  
  getMarks: () => {  
    ⚡ console.log(this); //parent's scope -> window  
    return this.marks;  
  },  
};
```

snehagupta7385@gmail.com

```
getMarks: () => {  
  console.log(this); //parent's scope -> window  
  return this.marks;  
},
```

```
getInfo1: function () {  
  setTimeout(() => {  
    console.log(this);  
  }, 2000);  
},
```


snehagupta7385@gmail.com

```
getInfo2: function () {  
  setTimeout(function () {  
    console.log(this);  
  }, 2000);  
},  
};
```

```
const cube = (n) => {  
  |💡 return n*n*n;|  
}
```

JS app.js > ...

1 const square = (n) => n * n;

2 

3 console.log(square(4));

4



Practice Qs

Write an arrow function that returns the square of a number 'n'.

$(n) \Rightarrow \{$ $\text{return } n * n;$ $\}$		$(n) \Rightarrow (n * n);$
---	--	----------------------------

Write a function that prints "Hello World" 5 times at intervals of 2s each.

$\left[\begin{array}{l} \text{setInterval}(() \Rightarrow \{ \\ \quad \text{print('HW')}; \\ \}, 2000); \end{array} \right.$	$\begin{array}{c} \text{10 seconds} \\ \hline \downarrow \\ \text{setTimeout} \end{array}$
--	--

```
let id = setInterval(() => {  
  console.log("Hello World");  
}, 2000);
```

```
setTimeout(() => {  
  clearInterval(id);  
}, 10000);
```

```
let id = setInterval(() => {  
  console.log("Hello World");  
}, 2000);
```

```
setTimeout(() => {  
  clearInterval(id);  
  console.log("clear interval ran");  
}, 10000);
```

```
getInfo1: function () {  
    setTimeout(() => {  
        console.log(this); //student  
    }, 2000);  
},  
getInfo2: function () {  
    ⚡ setTimeout(function () {  
        console.log(this); //window  
    }, 2000);  
},  
};
```