Questions:

Question #1

```
function logThis(){
  console.log(this);
}

const myObj = {
  logThis
}

myObj.logThis()
```

Question #2

```
function logThis(){
  console.log(this);
}

const myObj = {
  foo: function(){
    logThis();
  }
}

myObj.foo()
```

Question #3

```
const logThis = () => {
  console.log(this);
}

const myObj = {
  foo: logThis
}

myObj.foo()
```

Question #4

```
function logThis() {
  console.log(this);
}

const myObj = { name: "sag1v" }

logThis.apply(myObj)
```

Question #5

```
const logThis = () => {
  console.log(this);
}

const myObj = { name: "sag1v" }

logThis.apply(myObj)
```

Question #6

```
function logThis(){
  console.log(this);
}

const someObj = new logThis()
```

Question #7

```
function logThis(){
  'use strict'
  console.log(this);
}

function myFunc(){
  logThis();
}

const someObj = new myFunc()
```

Question #8

```
function logThis(){
  console.log(this);
}

class myClass {
  logThat(){
    logThis()
  }
}
```

```
const myClassInstance = new myClass()
myClassInstance.logThat()
```

Question #9

```
function logThis(){
  console.log(this);
}

class myClass {
  logThat(){
    logThis.call(this)
  }
}

const myClassInstance = new myClass()
myClassInstance.logThat()
```

Question #10

```
class myClass {
  logThis = () => {
    console.log(this);
  }
}

const myObj = { name: 'sagiv' };

const myClassInstance = new myClass()
myClassInstance.logThis.call(myObj)
```

Bonus questions

Question #11

```
function logThis() {
  console.log(this);
}

const btn = document.getElementById('btn');
btn.addEventListener('click', logThis);
```

Question #12

```
const logThis = () => {
  console.log(this);
}

const btn = document.getElementById('btn');
```

```
btn.addEventListener('click', logThis);
```

Challenge #1

```
const call = {
  caller: "mom",
  says: function() {
    console.log(`Hey, ${this.caller} just called.`);
  }
};

call.says();
```

What will the code above log to the console?
(A) Hey, undefined just called.
(B) Hey, mom just called.
(C) Hey, caller just called.

Challenge #2

```
const call = {
  caller: "mom",
  says: () => {
    console.log(`Hey, ${this.caller} just called.`);
  }
};

call.says();
```

What will the code above log to the console?
(A) Hey, undefined just called.
(B) Hey, mom just called.
(C) Hey, caller just called.

Challenge #3

```
const call = {
  caller: "mom",
  says: function() {
    console.log(`Hey, ${this.caller} just called.`);
  }
};
```

```
let newCall = call.says;

newCall();
```

What will the code above log to the console?

(A) Hey, undefined just called.

(B) Hey, mom just called.

Challenge #4

```
function anotherCaller() {
  console.log(`${this.caller} called, too!`);
}

const call = {
  caller: "mom",
  anotherCaller: anotherCaller,
  says: function() {
    console.log(`Hey, ${this.caller} just called.`);
  }
};

let newCall = call.anotherCaller;

newCall();
```

What will the code above log in the console?

(A) mom called, too!

(B) Hey, mom just called.

(C) undefined called, too!

```
const call = {
  caller: "mom",
  anotherCaller: function() {
        console.log(`${this.caller} called, too!`)
      },
  says: function() {
    console.log(`Hey, ${this.caller} just called.`);
  }
};

let newCall = call.anotherCaller;
newCall();
```