TW-011 STUDENT VERSION

TEAMWORK

CLARUSWAY
WAY TO REINVENT YOURSELF

Meeting Agenda

- ▶ Icebreaking
- ▶ Questions
- ▶ Interview Questions
- ▶ Coding Challenge
- ▶ Video of the week
- ▶ Retro meeting
- ▶ Case study / project

Teamwork Schedule

Ice-breaking 5m

- Personal Questions (Study Environment, Kids etc.)
- Any challenges (Classes, Coding, studying, etc.)
- Ask how they're studying, give personal advice.
- Remind that practice makes perfect.

Team work 5m

- Ask what exactly each student does for the team, if they know each other, if they care for each other, if they follow and talk with each other etc.

Ask Questions 15m

1. Which statement creates a new object using the Person constructor?

A. let student = construct Person;
B. let student = new Person();
C. let student = construct Person();
D. let student = Person();

2. When would 'results shown' be logged to the console?

```
let modal = document.querySelector('#result');
setTimeout(function () {
  modal.classList.remove('hidden');
}, 10000);
console.log('Results shown');
```

A. immediately
B. after 10 second
C. after 10000 seconds
D. after results are received from the HTTP request

3. What is the result in the console of running the code shown?

```
let Storm = function () {};
Storm.prototype.precip = 'rain';

let WinterStorm = function () {};
WinterStorm.prototype = new Storm();
WinterStorm.prototype.precip = 'snow';

let bob = new WinterStorm();
console.log(bob.precip);
```

A. Storm()
B. is not defined
C. 'snow'
D. 'rain'

4. What is the result in the console of running this code?

```
function logThis() {
  this.desc = 'logger';
  console.log(this);
}
new logThis();
```

A. {desc: "logger"}
B. undefined
C. window
D. function

5. For the following class, how do you get the value of 42 from an instance of X?

```
class X {
  get Y() {
    return 42;
  }
}
var x = new X();
```

A. x.get('Y')
B. x.Y
C. x.Y()
D. x.get().Y

6. Your code is producing the error: `TypeError: Cannot read property 'reduce' of undefined`. What does that mean?

A. You are calling a method named reduce on an object that's has a null value.
B. You are calling a method named reduce on an empty array.
C. You are calling a method named reduce on an object that's declared but has no value.
D. You are calling a method named reduce on an object that does not exist.

7. What is the result in the console of running the code shown?

```
var start = 1;
function setEnd() {
  var end = 10;
}
setEnd();
console.log(end);
```

A. 0
B. 1
C. 10
D. ReferenceError

8. What will this code log in the console?

```
function sayHello() {
  console.log('hello');
}

console.log(sayHello.prototype);
```

A. an object with a constructor property
B. undefined
C. 'hello'
D. an error message

9. Which method cancels event default behavior?

A. `stop()`
B. `cancel()`
C. `prevent()`
D. `preventDefault()`

10. Which method is called automatically when an object is initialized?

A. `create()`
B. `new()`
C `constructor()`
D. `init()`

Interview Questions 15m

1. What is the difference between a class and an object in JavaScript?

In JavaScript, a class is a blueprint for creating objects with similar properties and methods. An object is an instance of a class and is created using the new operator.

2.2. Why are classes important in OOP? How do they help developers write better code?

Classes are important in Object-Oriented Programming (OOP) as they provide a blueprint for creating objects (instances of the class), and encapsulate data and behavior within those objects. This abstraction and encapsulation helps developers to write more modular, maintainable, and reusable code.

3. Can you provide some examples of using inheritance in JavaScript?

Using prototype method.

4. What do you understand by polymorphism?

Polymorphism in JavaScript refers to the concept of an object or function having the ability to take on multiple forms. In the context of OOP (Object-Oriented Programming), polymorphism refers to the ability of an object to respond to the same method call differently based on its underlying type. This is achieved through method overloading and method overriding.

5. What is encapsulation?

Encapsulation in JavaScript refers to the concept of wrapping data and functions within an object, making it a self-contained unit and hiding its internal state and behavior from the outside world. This helps in reducing the coupling between objects and provides a way to maintain the state and behavior of an object.

Coding Challenge 20m

- Coding Challenge: Random Password Generator (JS-08)

☕

Coffee Break 10m

☕

Video of the Week 5m

- What are Classes, Objects, and Constructors?

Retro Meeting on a personal and team level 5m

Ask the questions below:

- What went well?
- What went wrong?
- What is the improvement areas?

Case study/Project 15m

Digital Clock JS-05

Closing 5m

-Next week's plan

-QA Session
