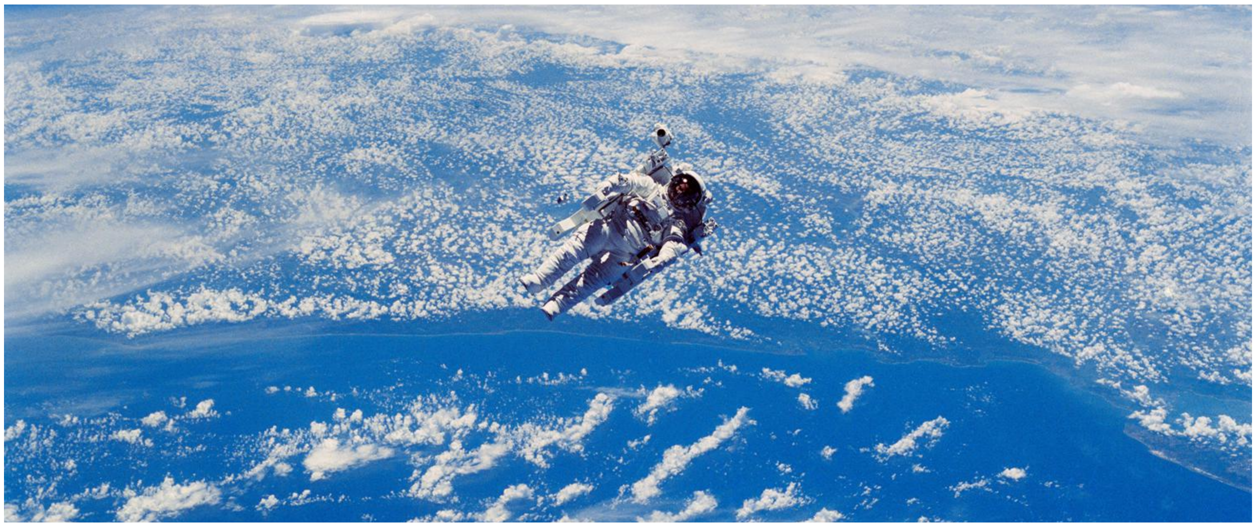

JS Class 4 Notes and Assignments

1. Control Structures: Conditional Statements

`if` / `else`

- `if`: Executes a block of code if the condition is true.
- `else`: Executes a block of code if the condition in the `if` statement is false.

Examples:

```
// Example 1: Simple if-else statement
let age = 20;
if (age >= 18) {
  console.log("You are eligible to vote.");
} else {
  console.log("You are not eligible to vote.");
}
```

```
// Example 2: Real-life temperature check
let temperature = 30;
if (temperature > 35) {
  console.log("It's too hot.");
} else if (temperature > 20) {
  console.log("It's a nice day.");
} else {
  console.log("It's cold.");
}
```

`switch`

- `switch`: Selects one of many code blocks to be executed, based on the value of an expression.

Examples:

```
// Example 1: Using switch to decide based on day of the week
let day = 3;
switch (day) {
  case 1:
    console.log("Monday");
    break;
  case 2:
    console.log("Tuesday");
    break;
  case 3:
    console.log("Wednesday");
    break;
  default:
    console.log("Invalid day");
}
```

```
// Example 2: Real-life fruit price based on fruit name
let fruit = "Apple";
switch (fruit) {
  case "Banana":
    console.log("Banana is $1");
    break;
  case "Apple":
    console.log("Apple is $2");
    break;
  default:
    console.log("Fruit not available.");
}
```

---

2. Loops

`for` Loop

- `for`: Repeats a block of code a certain number of times.

Examples:

```
// Example 1: Simple for loop
for (let i = 0; i < 5; i++) {
  console.log(i);  // Output: 0, 1, 2, 3, 4
}
```

```
// Example 2: Real-life total cart value calculation
let prices = [100, 200, 300];
let total = 0;
for (let i = 0; i < prices.length; i++) {
  total += prices[i];
}
console.log(`Total cart value: $${total}`);
```

`while` Loop

- `while`: Executes a block of code as long as a specified condition is true.

Examples:

```
// Example 1: Basic while loop
let i = 0;
```

```
while (i < 3) {
  console.log(i);
  i++;
}
```

```
// Example 2: Countdown timer simulation
let time = 10;
while (time > 0) {
  console.log(`${time} seconds left`);
  time--;
}
```

`do-while` Loop

- `do-while`: Executes the code block once before checking the condition, then repeats the loop as long as the condition is true.

Examples:

```
// Example 1: Basic do-while loop
let j = 0;
do {
  console.log(j);
  j++;
} while (j < 3);
```

```
// Example 2: ATM Pin attempt simulation
let pin;
let attempt = 0;
do {
  pin = prompt("Enter your pin:");
  attempt++;
} while (pin !== "1234" && attempt < 3);
```

`for...in` Loop

- `for...in`: Iterates over the properties of an object.

Examples:

```
// Example 1: Iterating over an object
let user = { name: "John", age: 25, city: "New York" };
for (let key in user) {
  console.log(key, user[key]);
}
```

```
// Example 2: Real-life user profile display
let profile = { username: "rohitdev", email: "rohit@domain.co
m", age: 30 };
for (let field in profile) {
  console.log(`${field}: ${profile[field]}`);
}
```

`forEach` Loop

- `forEach`: Executes a provided function once for each array element.

Examples:

```
// Example 1: Simple forEach loop
let numbers = [1, 2, 3];
numbers.forEach(function(number) {
  console.log(number);
});
```

```
// Example 2: Real-life list of tasks
let tasks = ["Clean the house", "Wash the dishes", "Buy groce
ries"];
tasks.forEach((task, index) => {
  console.log(`${index + 1}. ${task}`);
});
```

---

3. Break and Continue Statements

`break`

- `break`: Exits the current loop or switch statement.

Examples:

```
// Example 1: Using break to exit a loop
for (let i = 0; i < 5; i++) {
  if (i === 3) break;
  console.log(i);  // Output: 0, 1, 2
}
```

```
// Example 2: Real-life ticket selling scenario
let tickets = 5;
for (let i = 1; i <= 10; i++) {
  if (i > tickets) break;
  console.log(`Ticket ${i} sold`);
}
```

`continue`

- `continue`: Skips the current iteration and moves to the next one.

Examples:

```
// Example 1: Using continue to skip a value
for (let i = 0; i < 5; i++) {
  if (i === 2) continue;
  console.log(i);  // Output: 0, 1, 3, 4
}
```

```
// Example 2: Real-life skip out-of-stock items
let items = ["Phone", "Laptop", "Out-of-stock", "Tablet"];
items.forEach(item => {
  if (item === "Out-of-stock") {
    console.log("Skipping unavailable item");
    return;
  }
  console.log(`Selling ${item}`);
});
```

Assignment Questions

Conditional Statements: `if` / `else` and `switch`

1. Create a weather app that suggests what to wear based on the temperature (cold, moderate, or hot).
2. Develop a student grading system using `if` / `else` to determine if a student has passed or failed based on their marks.
3. Write a program to suggest meals based on the time of day using `if...else` (morning, afternoon, evening).
4. Create a vehicle type decision program where a user can input the number of wheels and get a suggestion for the type of vehicle (bike, car, etc.).
5. Use a `switch` statement to build a simple calculator that can perform addition, subtraction, multiplication, or division based on user input.

Loops (`for` , `while` , `do-while` , `forEach` , `for...in`)

1. Create a program that calculates the factorial of a given number using a `for` loop.
2. Create a countdown timer that logs every second until it reaches zero using a `while` loop.
3. Create a shopping cart that calculates the total cost of items using a `forEach` loop.

4. Write a program that asks the user to input a pin number up to 3 times. If the correct pin is entered, the loop stops (`do-while` loop).
5. Use a `for...in` loop to display all the details of a product (name, price, category) from an object.

Break and Continue Statements

1. Create a loop that counts from 1 to 10, but stops when it reaches 7 using the `break` statement.
2. Write a program that loops through numbers 1 to 10, but skips the numbers that are divisible by 3 using the `continue` statement.
3. Build a vending machine simulator that stops vending once the stock of an item is depleted using `break`.
4. Use a `continue` statement to loop through an array of products and skip over the ones that are out of stock.
5. Create a program that asks for user input for up to 5 attempts but breaks the loop when a valid input is received.