PW SKILLS ASSIGNMENT

RAJKAMAL YADAV

Question 1:

What are conditional statements? Explain conditional statements with syntax and examples?

Answer:

Conditional statements in JavaScript are used to perform different actions based on different conditions. They help in decision-making processes within the code, allowing the program to execute certain sections of code depending on whether a condition evaluates to true or false.

Common Conditional Statements in JavaScript:

1. **if Statement**
2. **if-else Statement**
3. **if-else-if Statement**
4. **Switch Statement**
5. **Ternary Operator (Conditional Operator)**

1. if Statement

The if statement is used to execute a block of code if a specified condition is true.

Syntax:

```
if (condition) {

// code to be executed if the condition is true

}
```

Example:

```
let age = 20;

if (age >= 18) {

 console.log("You are an adult.");

}
```

Out Put: **“You are an adult”**

2. if-else Statement

The if-else statement provides an alternative code block that executes if the condition in the if statement is false.

Syntax:

```
if (condition) {

 // code to be executed if the condition is true

}

else {

 // code to be executed if the condition is false

}
```

Example:

```
let age = 16;

if (age >= 18) {

 console.log("You are an adult.");

}

else {

console.log("You are not an adult.");

}
```

Out Put: **“You are not an adult”**

3. if-else-if (also known as if-elseif-else or else if) Statement

The if-else-if statement is used when you want to check multiple conditions. As soon as one of the conditions is true, the corresponding block of code is executed, and the rest are ignored.

Syntax:

```
if (condition1) {
   // code to be executed if condition1 is true
} else if (condition2) {
   // code to be executed if condition2 is true
} else {
   // code to be executed if neither condition1 nor condition2 is true
}
```

Example:

```
let score = 75;
if (score >= 90) {
 console.log("Grade: A");
}
else if (score >= 80) {
 console.log("Grade: B");
}
else if (score >= 70) {
console.log("Grade: C");
}
else {
console.log("Grade: F");
}
```

Out Put: **“Grade:C”**

4. switch Statement

The switch statement is used to perform different actions based on the value of a single expression. It is an alternative to using multiple if-else statements when checking the same variable against multiple values.

Syntax:

```
switch (expression) {
   case value1:
      // code to be executed if expression === value1
      break;
   case value2:
      // code to be executed if expression === value2
      break;
   // more cases...
   default:
      // code to be executed if expression does not match any case
}
```

Example:

```
let day = 6;
switch (day) {
   case 1:
      dayName = "Monday";
      break;
   case 2:
      dayName = "Tuesday";
      break;
   case 3:
```

```
        dayName = "Wednesday";
        break;
    case 4:
        dayName = "Thursday";
        break;
    case 5:
        dayName = "Friday";
        break;
    case 6:
        dayName = "Saturday";
        break;
    case 7:
        dayName = "Sunday";
        break;
    default:
        dayName = "Invalid day";
}
```

Out Put: **“Saturday”**

5. Ternary Operator (Conditional Operator)

The ternary operator is a shorthand for the if-else statement. It takes three operands: a condition, a result for true, and a result for false.

Syntax:

```
condition ? expressionIfTrue : expressionIfFalse;
```

Example:

```
let age = 18;

let message = age >= 18 ? "You are an adult." : "You are not an adult.";

console.log(message); // Output: You are an adult.
```

Q2. Write a program that grades students based on their marks?

- **If greater than 90 then A Grade.**
- **If between 70 to 90 then B Grade.**
- **If between 50 to 70 then C Grade.**
- **Below 50 then an F Grade.**

Solution:

```
let score=95;


if (score>90) {
   console.log("A Grade");
}
else if (score<90 && score>70){
console.log("b Grade");
}
else if (score<70 && score>50){
   console.log("C Grade");
}
else {
   console.log("F Grade")
```

```
}
```

Out Put: "A Grade"

Q3. What are loops, and what do we need them? Explain different types of loops with their syntax and examples?

Answer:

Loops in JavaScript are used to repeatedly execute a block of code as long as a specified condition is true. They are essential for automating repetitive tasks, iterating over arrays or objects, and efficiently handling large amounts of data.

Types of Loops in JavaScript:

1. **for Loop**
2. **while Loop**
3. **do-while Loop**
4. **for...in Loop**
5. **for...of Loop**

1. for Loop

The for loop is the most commonly used loop in JavaScript. It repeats a block of code a known number of times. The loop has three parts: initialization, condition, and increment/decrement.

Syntax:

```
for (initialization; condition; increment/decrement) {
   // code to be executed
}
```

Example:

```
for (let i = 0; i < 5; i++) {
   console.log("Iteration number: " + i);
}
```

Output: Iteration number: 0, 1, 2, 3, 4

2. while Loop

The while loop repeats a block of code as long as a specified condition is true. It is useful when the number of iterations is not known beforehand.

Syntax:

```
while (condition) {
  // code to be executed
}
```

Example:

```
let i = 0;
while (i < 5) {
  console.log ("Iteration number: " + i);
  i++;
}
```

Output: Iteration number: 0, 1, 2, 3, 4

3. do-while Loop

The do-while loop is similar to the while loop, but it guarantees that the code block is executed at least once before the condition is tested.

Syntax:

```
do {
  // code to be executed
} while (condition);
```

Example:

```
let i = 0;
do {
  console.log("Iteration number: " + i);
  i++;
} while (i < 5);
```

Output: Iteration number: 0, 1, 2, 3, 4

4. for...in Loop

The for...in loop iterates over the properties of an object (or the indices of an array). It is primarily used for iterating through object properties.

Syntax:

```
for (key in object) {
  // code to be executed
}
```

Example:

```
const person = { name: "Rajkamal", age: 25, city: "Jaunpur" };

for (let key in person) {
  console.log(key + ": " + person[key]);
}
```

Output: name: Rajkamal, age: 25, city: Jaunpur

5. for...of Loop

The for...of loop iterates over the values of an iterable object, such as an array, string, or Set. It is useful for working with arrays and other iterable data structures.

Syntax:

```
for (variable of iterable) {
  // code to be executed
}
```

Example:

```
const numbers = [10, 20, 30];

for (let num of numbers) {
  console.log(num);
}
```

Output: **10, 20, 30**

Q4. Generate numbers between any 2 given numbers.

Ex: const num1 = 10;

const num2 = 25;

Out Put: 11, 12, 13, …., 25

Solution:

```
const num1 = 10;
const num2 = 25;
let numbers = [];

for (let i = num1 + 1; i <= num2; i++) {
   numbers.push(i);
}

console.log(numbers.join(", "));
```

Output: 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25

Q5. Use the while loop to print n9mbers from 1 to 25 in ascending and descending order?

Solution:

```
// Code for ascending order
 let i=1;
while(i<=25) {
   console.log(i);
   i++;
}
```

Output:

1
2
3
4
5
6
7
8
9

```
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
// Code for descending order
 let i=25;
while(i>=1) {
   console.log(i);
   i--;
}
```

Output:

```
25
24
23
22
21
20
19
18
17
16
15
14
```

13

12

11

10

9

8

7

6

5

4

3

2

1