Assignment : Conditions and Loops

Q1. What are conditional statements? Explain conditional statements with syntax and examples.

Ans: Conditional statements are used in programming to make decisions.
They allow a program to execute different blocks of code based on whether a condition is true or false.

Common Conditional Statements:
if Statement :

Purpose:

Executes a block of code only if the condition is true.

Syntax:

```
if (condition) {
    // code to execute
}
```

Example:

```
let age = 18;

if (age >= 18) {
    console.log("You are eligible to vote");
}
```

if–else Statement

Purpose:

Executes one block if condition is true, another if false.

Syntax:

```
if (condition) {
    // true block
```

```
} else {
    // false block
}
```

Example:

```
let marks = 35;

if (marks >= 40) {
    console.log("Pass");
} else {
    console.log("Fail");
}
```

switch Statement

Purpose:

Used when checking one value against many cases.

Syntax:

```
switch (expression) {
    case value1:
        // code
        break;
    case value2:
        // code
        break;
    default:
        // code
}
```

Example:

```
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");
}
```

/*Q2. Write a program that grades students based on their marks^
If greater than 90 then A GradE
If between 70 and 90 then a B gradE
If between 50 and 70 then a C gradE
Below 50 then an F grade */

```
//* Ans:

function StudentsGrade(marks){
    if(marks > 90){
        console.log("Grade : A");
    }
    else if(marks<90 && marks>70){
      console.log("Grade : B");
    }
    else if(marks<70 && marks>50){
        console.log("Grade : C");
    }else{
        console.log("Grade : F");
    }
}

StudentsGrade(43);
```

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

Ans:

Loops are control statements used to repeat a block of code multiple times until a given condition is satisfied.

Instead of writing the same code again and again, we use loops to make programs:

- Shorter
- Easier to understand
- Easier to maintain

need for loops:

Without loops:

```
console.log("Hello");
console.log("Hello");
console.log("Hello");
```

With loops:

```
for (let i = 1; i <= 3; i++) {
    console.log("Hello");
}
```

- Saves time
- Reduces errors
- Improves readability

Types of Loops (with Syntax & Examples):

for Loop

- Use:

When the number of iterations is known.

- Syntax:

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

- Example:

```
for (let i = 1; i <= 5; i++) {
    console.log(i);
}
```

- Explanation:

Initialization → let i = 1

Condition → i <= 5

Increment → i++

while Loop

- Use:

When the number of iterations is not fixed and depends on a condition.

- Syntax:

```
while (condition) {
    // code to execute
}
```

- Example:

```
let i = 1;

while (i <= 5) {
    console.log(i);
    i++;
}
```

- Explanation:

Loop runs as long as the condition is true.

do–while Loop

- Use:

When the loop must execute at least once, even if the condition is false.

- Syntax:

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

- Example:

```
let i = 6;

do {
    console.log(i);
```

```
    i++;
} while (i <= 5);
```

- Explanation:

Output is 6 because the code runs once before checking the condition.

for...of Loop

- Use:

To loop through values of arrays or strings.

- Syntax:

```
for (let variable of iterable) {
    // code
}
```

- Example:

```
let fruits = ["Apple", "Banana", "Mango"];

for (let fruit of fruits) {
    console.log(fruit);
}
```

for...in Loop

- Use:

To loop through keys or indexes of an object or array.

- Syntax:

```
for (let key in object) {
    // code
```

```
}
```

- Example:

```
let student = {
    name: "Rahul",
    age: 20
};

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

Difference Between Loops

Loop Best Used When

- for: Known number of repetitions
- while Condition-based: repetition
- do–while: Must run at least once
- for...of : Array or string values
- for...in : Object properties

```
/*Q4. Generate numbers between any 2 given numbers.

Ex
 const num1 = 10
 const num2 = 25;

Output: 11, 12, 13, …., 25
*/
//* Ans:

function PrintBetween(num1,num2){
    for(let i = 1; i + Math.min(num1,num2)<Math.max(num1,num2); i++){
        console.log(i+Math.min(num1,num2));
    }
```

```
}
PrintBetween(10,25);
```

//* Q5. Use the while loop to print numbers from 1 to 25 in ascending and descending order.

//* Ans:

```
function PrintAcc(){
    let i = 1;
    while(i<=25){
        console.log(i);
        i++;
    }
}
PrintAcc();

function PrintDcc(){
    let i = 25;
    while(i>=1){
        console.log(i);
        i--;
    }
}
PrintDcc();
```