Q1. Make a code using Bitwise operator.

```
using System;

class Program
{
    static void Main()
    {
        int num1 = 5;  // Binary: 0101
        int num2 = 3;  // Binary: 0011

        // Bitwise AND (&) operator
        int resultAnd = num1 & num2;  // Binary: 0001 (Decimal: 1)
        Console.WriteLine("Bitwise AND: " + resultAnd);

        // Bitwise OR (|) operator
        int resultOr = num1 | num2;    // Binary: 0111 (Decimal: 7)
        Console.WriteLine("Bitwise OR: " + resultOr);

        // Bitwise XOR (^) operator
        int resultXor = num1 ^ num2;  // Binary: 0110 (Decimal: 6)
        Console.WriteLine("Bitwise XOR: " + resultXor);

        // Bitwise NOT (~) operator
        int resultNot = ~num1;        // Binary: 11111111111111111111111111111010
(Decimal: -6)
        Console.WriteLine("Bitwise NOT: " + resultNot);
```

```
        // Left Shift (<<) operator
        int resultLeftShift = num1 << 2;  // Binary: 010100 (Decimal: 20)
        Console.WriteLine("Left Shift: " + resultLeftShift);

        // Right Shift (>>) operator
        int resultRightShift = num1 >> 1;  // Binary: 0010 (Decimal: 2)
        Console.WriteLine("Right Shift: " + resultRightShift);
    }
}
```

Q2.

Pointers in C#:

In C#, pointers are used to directly manipulate memory addresses. However, pointer operations are considered unsafe because they can lead to security vulnerabilities and memory corruption if not used carefully. To use pointers in C#, you need to mark a block of code as unsafe using the unsafe keyword.

Here's a brief overview of pointers in C#:

You can declare pointer variables using the asterisk * symbol.

You can use the & operator to obtain the address of a variable.

You can use the * operator to access the value at a memory address.

Example:

```
int number = 42;
unsafe
```

```
{
   int* ptr = &number;
   Console.WriteLine("Value: " + *ptr);
}
```

2. Functions (Methods) in C#:

Functions in C# are blocks of code that perform a specific task.

They help in organizing code into reusable and modular components.

Functions are declared using the static keyword for static methods and void or another

data type for methods that return a value.

Example:

```
static int AddNumbers(int num1, int num2)
{
   return num1 + num2;
}
```

3. Loops in C#:

Loops are used to repeatedly execute a block of code as long as a certain condition is met. C# provides several types of loops:

For Loop: It allows you to specify an initialization, a condition, and an increment or decrement operation in a single line.

Example:

```
for (int i = 0; i < 5; i++)
{
    // Code to be executed in each iteration
}
```

While Loop: It repeatedly executes a block of code as long as a specified condition is true.

```
int i = 0;
while (i < 5)
// Code to be executed in each iteration
    i++;
}
```

Do-While Loop: It is similar to the while loop, but it guarantees that the code block will be executed at least once before checking the condition.

```
int i = 0;
do
{
    // Code to be executed in each iteration
    i++;
} while (i < 5);
```