Assignment 2 C#

Question 1.Datatypes in C#?

Answer:

**Primitive Data Types:**

* Boolean: stores true or false values.
* Byte: stores integer values from 0 to 255.
* SByte: stores integer values from -128 to 127.
* Short: stores integer values from -32,768 to 32,767.
* UShort: stores integer values from 0 to 65,535.
* Int: stores integer values from -2,147,483,648 to 2,147,483,647.
* U Int: stores integer values from 0 to 4,294,967,295.
* Long: stores integer values from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807.
* U Long: stores integer values from 0 to 18,446,744,073,709,551,615.
* Float: stores single-precision floating-point values.
* Double: stores double-precision floating-point values.
* Decimal: stores decimal values with high precision and a large range of values.
* Char: stores a single Unicode character.
* Void: represents the absence of a type.

**Non-Primitive Data Types:**

* String: stores a sequence of characters.
* Object: stores any type of value, including value types and reference types.
* Class: defines a blueprint for creating objects that contain data and methods.
* Interface: defines a contract that specifies the properties, methods, and events that a class must implement.
* Delegate: defines a reference to a method with a specific signature.
* Array: stores a fixed-size sequential collection of elements of the same type.
* Enum: defines a set of named constants.
* Struct: defines a value type that can contain data members and methods.

------------------------------------------------------------------------------------------------------------------------

Question 2:What is Statements and condition?

Answer:

**Statements:**

* A statement is a single line of code that performs a specific action.
* Statements are executed sequentially, one after the other, unless control flow statements are used to change the order of execution.
* Examples of statements include variable declarations, assignments, method calls, and control flow statements.

**Conditions:**

* Conditions are used to control the flow of a program based on certain logical conditions.
* C# supports various conditional statements, including if, else if, else, and switch statements.
* The if statement is used to execute a block of code if a specified condition is true.
* The else statement is used to execute a block of code if the same condition is false.
* The else if statement is used to specify a new condition to test if the first condition is false.
* The switch statement is used to specify many alternative blocks of code to be executed based on the value of an expression.

-------------------------------------------------------------------------------------------------------------------------------------

Question3 .Define Operators in C#?

Answer:

**Arithmetic Operators:**

Addition (+): Adds two operands.

Subtraction (-): Subtracts the right operand from the left operand.

Multiplication (\*): Multiplies two operands.

Division (/): Divides the left operand by the right operand.

Modulus (%): Returns the remainder after division.

**Assignment Operators:**

Assignment (=): Assigns a value to a variable.

Addition Assignment (+=): Adds the right operand to the left operand and assigns the result to the left operand.

Subtraction Assignment (-=): Subtracts the right operand from the left operand and assigns the result to the left operand.

Multiplication Assignment (\*=): Multiplies the left operand by the right operand and assigns the result to the left operand.

Division Assignment (/=): Divides the left operand by the right operand and assigns the result to the left operand.

Modulus Assignment (%=): Calculates the modulus of the left operand with the right operand and assigns the result to the left operand.

**Comparison Operators:**

Equal (==): Checks if two operands are equal.

Not Equal (!=): Checks if two operands are not equal.

Greater Than (>): Checks if the left operand is greater than the right operand.

Less Than (<): Checks if the left operand is less than the right operand.

Greater Than or Equal To (>=): Checks if the left operand is greater than or equal to the right operand.

Less Than or Equal To (<=): Checks if the left operand is less than or equal to the right operand.

**Logical Operators:**

Logical AND (&&): Returns true if both operands are true.

Logical OR (||): Returns true if at least one operand is true.

Logical NOT (!): Returns the opposite of the operand's value.

**Bitwise Operators:**

Bitwise AND (&): Performs a bitwise AND operation.

Bitwise OR (|): Performs a bitwise OR operation.

Bitwise XOR (^): Performs a bitwise exclusive OR operation.

Bitwise NOT (~): Performs a bitwise NOT operation.

Left Shift (<<): Shifts bits to the left.

Right Shift (>>): Shifts bits to the right.

**Unary Operators:**

Unary Plus (+): Represents a positive value (rarely used).

Unary Minus (-): Negates a value.

Increment (++): Increments a variable's value by 1.

Decrement (--): Decrements a variable's value by 1.

Logical NOT (!): Inverts the boolean value.

**Ternary Operator (Conditional Operator):**

Conditional ( ? : ): Provides a shorthand way of writing if-else statements.

----------------------------------------------------------------------------------------------------------------------------

Question4: Can +12 and -12 time zone exist. If exist can it meet at same point?

Answer:

Yes, both +12 and -12 time zones exist, and they can meet at the same point on the Earth's surface. These time zones are often referred to as UTC+12 and UTC-12, respectively.

UTC+12 is 12 hours ahead of Coordinated Universal Time (UTC), while UTC-12 is 12 hours behind UTC. When you move 12 hours ahead or behind UTC, you are essentially going around the world in terms of time zones.

-------------------------------------------------------------------------------------------------------

Question 5: Program of Bitwise Operators?

Answer:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BitwiseOperators

{

internal class Program

{

static void Main(string[] args)

{

{

int a = 60;

int b = 13;

int c;

c = a & b;

Console.WriteLine("a & b = " + c);

c = a | b;

Console.WriteLine("a | b = " + c);

c = a ^ b;

Console.WriteLine("a ^ b = " + c);

c = ~a;

Console.WriteLine("~a = " + c);

c = a << 2;

Console.WriteLine("a << 2 = " + c);

c = a >> 2;

Console.WriteLine("a >> 2 = " + c);

Console.ReadLine();

}

}

}

}