Dot Net Imp

Q1. Short Answer Type Questions.

1. What is CTS?

The Common Type System (CTS) is a standard in the .NET framework that defines how types are declared, used, and managed in the runtime environment. It ensures that objects created in different programming languages can interact with each other. CTS provides a set of rules for data types, allowing for type safety and interoperability among languages.

2. State any two advantages of .NET.

1. **Cross-Platform Development**: With .NET Core and .NET 5+, developers can build applications that run on multiple platforms, including Windows, macOS, and Linux.
2. **Rich Class Library**: .NET provides a comprehensive class library that simplifies development by offering pre-built functionalities for tasks such as file handling, database access, and web services.

3. What is Event Driven Programming?

Event-driven programming is a programming paradigm in which the flow of the program is determined by events such as user actions (mouse clicks, key presses) or messages from other programs. In this model, the program responds to events by executing specific event handlers, making it suitable for graphical user interfaces and interactive applications.

4. Explain the difference between menu and popup menu in .NET.

- **Menu**: A menu is a list of options or commands presented to the user, typically displayed at the top of an application window. Menus can

contain submenus and are usually always visible or accessible through a menu bar.

- **Popup Menu**: A popup menu (or context menu) is a temporary menu that appears when the user right-clicks on an item or performs a specific action. It provides quick access to relevant commands related to the selected item and disappears when the user makes a selection or clicks outside the menu.

5. List any two properties of Radio Button Control.

1. **Checked**: A boolean property that indicates whether the radio button is selected (true) or not (false).
2. **GroupName**: A string property that specifies the group to which the radio button belongs, ensuring that only one radio button in the group can be selected at a time.

6. What do you mean by value type and reference type?

- **Value Type**: Value types store the actual data directly in memory. They are typically stored on the stack and include types such as integers, floats, and structs. When a value type is assigned to another, a copy of the value is made.
- **Reference Type**: Reference types store a reference to the actual data, which is stored on the heap. They include types such as classes, arrays, and strings. When a reference type is assigned to another, both variables point to the same object in memory.

7. What is boxing in C#?

Boxing is the process of converting a value type into a reference type by wrapping it in an object. This allows value types to be treated as objects,

enabling them to be stored in collections that require reference types. Boxing incurs a performance cost due to the additional memory allocation and the need to unbox the value when retrieving it.

8. What is meant by ADO.NET?

ADO.NET is a set of classes in the .NET framework that provides data access services for .NET applications. It allows developers to interact with various data sources, such as databases and XML files, using a disconnected architecture. ADO.NET supports both connected and disconnected data access models, enabling efficient data manipulation and retrieval.

9. Enlist any four data types used in .NET.

1. **int**: Represents a 32-bit signed integer.
2. **string**: Represents a sequence of characters.
3. **bool**: Represents a boolean value (true or false).
4. **double**: Represents a double-precision floating-point number.

10. What is the use of the 'this' keyword in C#?

The **this** keyword in C# refers to the current instance of the class. It is used to differentiate between class members and parameters with the same name, to pass the current instance to another method, or to call another constructor within the same class.

11. What is the use of CLR?

The Common Language Runtime (CLR) is the virtual machine component of the .NET framework that manages the execution of .NET programs. It provides

services such as memory management, type safety, exception handling, and garbage collection, ensuring that applications run efficiently and securely.

12. Enlist any two operators in VB.NET.

1. **Arithmetic Operators**: Used for performing mathematical operations (e.g., +, -, *, /).
2. **Comparison Operators**: Used for comparing two values (e.g., =, <>, <, >).

13. Explain the following functions:

- **i. MessageBox()**: Displays a modal dialog box that can contain a message, title, and buttons for user interaction. It is commonly used to show alerts or prompts to the user.
- **ii. InputBox()**: Displays a dialog box that prompts the user for input. It allows the user to enter a string value, which can be used in the application.

14. Explain constructor and destructors in C#.

- **Constructor**: A special method that is called when an instance of a class is created. It is used to initialize the object's properties and allocate resources. Constructors can be overloaded to provide different ways of initializing an object.
- **Destructor**: A special method that is called when an object is destroyed. It is used to perform cleanup operations, such as releasing unmanaged resources. In C#, destructors are defined using the **~ClassName** syntax.

15. Explain server object.

Server objects are objects that are created and managed on the server side in web applications. They provide functionality for handling requests, managing sessions, and accessing resources. Examples of server objects include Application, Session, and Request objects in ASP.NET, which allow developers to manage application state and user sessions.

16. Explain the types of menu control.

Menu controls in .NET can be categorized into several types:

- **Main Menu**: A menu that is displayed at the top of the application window, typically containing top-level options.
- **Context Menu (Popup Menu)**: A menu that appears upon right-clicking an item, providing quick access to relevant commands.
- **MenuStrip**: A control that provides a way to create a menu bar in Windows Forms applications, allowing for hierarchical menus.

17. Explain connected and disconnected architecture in ADO.NET.

- **Connected Architecture**: In this model, a continuous connection to the database is maintained while performing operations. It allows for real-time data manipulation but can be resource-intensive.
- **Disconnected Architecture**: In this model, a connection to the database is established only when needed. Data is retrieved into a dataset, and the connection is closed. This approach is more efficient for applications that do not require constant database access.

18. Explain Timer control in VB.NET.

The Timer control in VB.NET is used to execute a block of code at specified intervals. It raises the **Tick** event at regular intervals, allowing developers to perform actions such as updating the user interface or executing background tasks. The Timer control can be configured by setting its **Interval** property (in milliseconds) and enabling it with the **Enabled** property.

19. What is IDE?

An Integrated Development Environment (IDE) is a software application that provides comprehensive facilities to programmers for software development. It typically includes a code editor, debugger, compiler, and tools for building and managing projects. Examples of IDEs for .NET development include Visual Studio and Visual Studio Code.

20. What is CLS?

The Common Language Specification (CLS) is a set of rules and guidelines that define a subset of the Common Type System (CTS) to ensure interoperability between different .NET languages. It specifies which features of the .NET framework are available to all languages, promoting consistency and compatibility across languages.

21. Explain JIT compilers.

Just-In-Time (JIT) compilers are part of the CLR that convert Intermediate Language (IL) code into native machine code at runtime. This process allows for optimizations specific to the current execution environment, improving performance. JIT compilation occurs the first time a method is called, and the compiled code is cached for subsequent calls.

22. Explain class and object in C#.

- **Class**: A class is a blueprint for creating objects. It defines properties, methods, and events that the objects created from the class will have. Classes encapsulate data and behavior, promoting code reusability and organization.
- **Object**: An object is an instance of a class. It represents a specific entity with its own state and behavior defined by the class. Objects can interact with each other and can be manipulated through their methods and properties.

23. What is method overloading in C#?

Method overloading is a feature in C# that allows multiple methods in the same class to have the same name but different parameters (different type, number, or order of parameters). This enables developers to create methods that perform similar functions but with different input types or quantities, enhancing code readability and usability.

24. Explain the Range validator.

The Range validator is a control in ASP.NET that checks whether a user input value falls within a specified range. It can be used to validate numeric, date, or string values. The Range validator requires setting minimum and maximum values, and it provides feedback to the user if the input is outside the defined range.

25. What is ASP.NET?

ASP.NET is a web application framework developed by Microsoft for building dynamic web applications, services, and websites. It allows developers to create web applications using .NET languages such as C# and VB.NET.

ASP.NET provides a rich set of controls, libraries, and tools for building scalable and secure web applications.

26. Enlist any two form controls in VB.NET.

1. **TextBox**: A control that allows users to input text data.
2. **Button**: A control that users can click to perform an action or trigger an event.

27. Enlist concatenation operators in VB.NET.

1. **&**: The primary operator for concatenating strings.
2. **+**: Can also be used for concatenation, but it may lead to ambiguity when used with numeric types.

28. List properties of Array in C#.

1. **Length**: Gets the total number of elements in the array.
2. **Rank**: Gets the number of dimensions of the array.

29. What do you mean by constructor?

A constructor is a special method in a class that is called when an instance of the class is created. It is used to initialize the object's properties and allocate resources. Constructors can have parameters to allow for different initialization scenarios and can be overloaded to provide multiple ways to create an object.

30. What is ADO.NET Dataset?

An ADO.NET Dataset is an in-memory representation of data that can hold multiple tables, relationships, and constraints. It is a disconnected data structure that allows for data manipulation and retrieval without maintaining a constant connection to the database. Datasets can be filled with data from a database and can be used to update the database when changes are made.

31. Write any two string functions in C#.

1. **Substring(int startIndex, int length)**: Returns a substring from the specified start index with the specified length.
2. **Replace(string oldValue, string newValue)**: Replaces all occurrences of a specified string with another string.

32. Enlist any four data types used in VB.NET.

1. **Integer**: Represents a 32-bit signed integer.
2. **String** : Represents a sequence of characters.
3. **Boolean**: Represents a true or false value.
4. **Double**: Represents a double-precision floating-point number.

33. What is the use of the virtual keyword?

The **virtual** keyword in C# is used to declare a method or property in a base class that can be overridden in derived classes. It allows for polymorphism, enabling derived classes to provide specific implementations of the method or property while maintaining the same interface.

34. List any four common web controls.

1. **Button**: A control that triggers an event when clicked.

2. **Label**: A control used to display text on a web page.
3. **TextBox**: A control that allows users to input text.
4. **DropDownList**: A control that allows users to select an item from a list of options.

35. What is the use of SiteMapPath control?

The SiteMapPath control in ASP.NET is used to display a breadcrumb navigation trail for users. It shows the current page's position within the site hierarchy, allowing users to easily navigate back to previous pages. This enhances user experience by providing context and improving site navigation.

36. List any four properties of ComboBox control.

1. **Items**: A collection that holds the items displayed in the ComboBox.
2. **SelectedItem**: Gets or sets the currently selected item in the ComboBox.
3. **DropDownStyle**: Determines how the ComboBox displays its items (e.g., dropdown or simple).
4. **Text**: Gets or sets the text associated with the ComboBox, which can be the selected item or user input.

Q2. Long Answer Type Questions.

1. Explain DataGridView control.

The DataGridView control in Windows Forms is a versatile control used to display and manipulate tabular data. It allows users to view, edit, and navigate through data in a grid format. Key features include:

- **Data Binding**: Supports binding to various data sources, such as databases, collections, and arrays.

- **Customization**: Allows customization of columns, rows, and cell styles, including formatting and colors.
- **Editing**: Supports in-place editing of data, allowing users to modify values directly in the grid.
- **Sorting and Filtering**: Provides built-in functionality for sorting and filtering data.

2. Explain the architecture of ASP.NET.

The architecture of ASP.NET consists of several key components:

- **ASP.NET Application**: The web application that contains the code and resources.
- **Web Server**: Hosts the ASP.NET application and processes incoming requests (e.g., IIS).
- **ASP.NET Runtime**: Manages the execution of ASP.NET applications, including request processing, session management, and security.
- **Common Language Runtime (CLR)**: Executes the .NET code and provides services such as memory management and exception handling.
- **Framework Class Library (FCL)**: A collection of reusable classes and libraries that provide functionality for building web applications.

3. Explain classes in ADO.NET.

ADO.NET provides several classes for data access and manipulation:

- **Connection**: Represents a connection to a specific data source. It is used to establish a connection to the database.

- **Command**: Represents a SQL statement or stored procedure to be executed against a database. It is used to perform operations such as SELECT, INSERT, UPDATE, and DELETE.
- **DataReader**: Provides a fast, forward-only way to read data from a database. It is used for retrieving data in a read-only manner.
- **DataSet**: An in-memory representation of data that can hold multiple tables and relationships. It allows for disconnected data access.
- **DataAdapter**: Acts as a bridge between a DataSet and the database, allowing for data retrieval and updates.

4. What are the properties and methods of the server object?

The server object in ASP.NET provides methods and properties for server-side operations. Key properties and methods include:

- **Properties**:
    - **ScriptTimeout**: Gets or sets the time in seconds before a script times out.
    - **MapPath**: Maps a virtual path to a physical path on the server.
- **Methods**:
    - **Transfer()**: Transfers execution to another page on the server without making a round trip to the client.
    - **Execute()**: Executes a specified page and returns the output to the calling page.

5. Explain MessageBox function in detail.

The MessageBox function in .NET is used to display a modal dialog box that can contain a message, title, and buttons for user interaction. It is commonly

used to show alerts, warnings, or prompts to the user. The function can be customized with various parameters, including:

- **Text**: The message to be displayed.
- **Caption**: The title of the message box.
- **Buttons**: Specifies which buttons to display (e.g., OK, Cancel).
- **Icon**: Specifies the icon to be displayed (e.g., Information, Warning). Example:

```
MessageBox.Show("This is a message.", "Message Box Title",
MessageBoxButtons.OK, MessageBoxIcon.Information);
```

6. Explain advantages of .NET.

1. **Cross-Platform Development**: With .NET Core and .NET 5+, developers can build applications that run on multiple platforms, including Windows, macOS, and Linux.
2. **Rich Class Library**: .NET provides a comprehensive class library that simplifies development by offering pre-built functionalities for tasks such as file handling, database access, and web services.
3. **Language Interoperability**: .NET supports multiple programming languages, allowing developers to use the best language for their needs while still being able to work together.
4. **Security**: .NET provides built-in security features, including code access security and role-based security, to protect applications from unauthorized access.

7. Explain any four properties of TextBox control.

1. **Text**: Gets or sets the text displayed in the TextBox.

2. **MaxLength**: Gets or sets the maximum number of characters that can be entered in the TextBox.
3. **ReadOnly**: Gets or sets a value indicating whether the TextBox is read-only, preventing user input.
4. **Multiline**: Gets or sets a value indicating whether the TextBox can accept multiple lines of text, allowing for larger text input.

8. Explain pop menus in VB.NET.

Popup menus, also known as context menus, are menus that appear when a user right-clicks on an item or performs a specific action. In VB.NET, popup menus can be created using the ContextMenuStrip control. Key features include:

- **Dynamic Display**: Popup menus can be displayed dynamically based on user actions or application state.
- **Customization**: Developers can add various menu items, including submenus, and assign event handlers to respond to user selections.
- **Integration**: Popup menus can be integrated with other controls, allowing for context-sensitive options based on the selected item.

9. What is control structures? Discuss any two control structures.

Control structures are constructs that dictate the flow of execution in a program. They allow developers to control the order in which statements are executed. Two common control structures are:

- **Conditional Statements**: These include **if**, **else if**, and **else** statements that execute different blocks of code based on specified conditions.
- **Loops**: These include **for**, **while**, and **do-while** loops that repeat a block of code multiple times until a specified condition is met.

10. Explain components of .NET framework.

The .NET framework consists of several key components:

- **Common Language Runtime (CLR)**: The execution engine that manages the execution of .NET applications, providing services such as memory management and security.
- **Framework Class Library (FCL)**: A comprehensive collection of reusable classes, interfaces, and value types that provide functionality for various programming tasks.
- **ASP.NET**: A framework for building web applications and services.
- **Windows Forms**: A set of classes for creating rich desktop applications.
- **Entity Framework**: An object-relational mapping (ORM) framework for data access.

11. Explain Inheritance with example.

Inheritance is a fundamental object-oriented programming concept that allows a class (derived class) to inherit properties and methods from another class (base class). This promotes code reusability and establishes a hierarchical relationship between classes.
Example:

```
public class Animal // Base class
{
   public void Eat()
   {
      Console.WriteLine("Eating...");
   }
}
```

```
public class Dog : Animal // Derived class
{
   public void Bark()
   {
      Console.WriteLine("Barking...");
   }
}
Dog myDog = new Dog();
myDog.Eat(); // Inherited method
myDog.Bark(); // Own method
```

12. Explain Server control.

Server controls are components in ASP.NET that run on the server and generate HTML content sent to the client. They provide a way to create dynamic web pages with rich functionality. Key features include:

- **State Management**: Server controls maintain their state across postbacks, allowing for persistent data.
- **Event Handling**: They support events, enabling developers to respond to user actions (e.g., button clicks).
- **Rich Functionality**: Server controls include various built-in controls such as GridView, Button, and TextBox, which simplify web development.

13. Explain command object.

The Command object in ADO.NET is used to execute SQL statements or stored procedures against a database. It encapsulates the command text, connection, and parameters needed for execution. Key properties and methods include:

- **CommandText**: The SQL statement or stored procedure to execute.
- **Connection**: The Connection object that represents the database connection.

- **ExecuteReader()**: Executes the command and returns a DataReader for reading data.
- **ExecuteNonQuery()**: Executes the command and returns the number of rows affected, typically used for INSERT, UPDATE, or DELETE operations.

14. Explain ASP.NET page life cycle in detail.

The ASP.NET page life cycle consists of a series of stages that a page goes through from the time a request is made until the response is sent back to the client. Key stages include:

- **Page Request**: The page is requested by the client.
- **Start**: The page's properties are initialized, and the request is processed.
- **Initialization**: Controls on the page are initialized, and their properties are set.
- **Load**: The page and its controls are loaded with data.
- **Postback Event Handling**: If the page is a postback, events are raised for controls that have been changed.
- **Rendering**: The page is rendered into HTML and sent to the client.
- **Unload**: Cleanup operations are performed, and resources are released.

15. What are the classes in ADO.NET? Explain in detail.

ADO.NET provides several key classes for data access and manipulation, each serving a specific purpose in the data handling process:

- **Connection**: This class represents a connection to a specific data source, such as a SQL Server database. It is used to establish a connection to the database and manage the connection state.
  The **Connection** class has properties like **ConnectionString**, which specifies the details needed to connect to the database.
- **Command**: The **Command** class is used to execute SQL statements or stored procedures against a database. It encapsulates the command

text, connection, and parameters needed for execution. Key properties include **CommandText** (the SQL statement) and **Connection** (the associated database connection). Methods like **ExecuteReader()**, **ExecuteNonQuery()**, and **ExecuteScalar()** are used to execute commands and retrieve results.

- **DataReader**: The **DataReader** class provides a fast, forward-only way to read data from a database. It is used for retrieving data in a read-only manner, making it efficient for scenarios where data is only read and not modified. The **DataReader** is connected to the database, meaning it requires an active connection to retrieve data.
- **DataSet**: The **DataSet** class is an in-memory representation of data that can hold multiple tables, relationships, and constraints. It allows for disconnected data access, meaning that data can be retrieved from the database, manipulated in memory, and then updated back to the database without maintaining a constant connection.
- **DataAdapter**: The **DataAdapter** class acts as a bridge between a **DataSet** and the database. It is responsible for filling the **DataSet** with data from the database and updating the database with changes made to the **DataSet**. The **DataAdapter** uses the **SelectCommand**, **InsertCommand**, **UpdateCommand**, and **DeleteCommand** properties to manage data operations.

16. How to create menus in VB.NET?

Creating menus in VB.NET can be done using the MenuStrip control, which allows developers to create a menu bar for Windows Forms applications. Here are the steps to create menus:

1. **Add a MenuStrip Control**: Drag and drop the MenuStrip control from the toolbox onto the form.
2. **Add Menu Items**: Click on the MenuStrip to add top-level menu items. You can add submenus by right-clicking on a menu item and selecting "Add" to create a dropdown menu.

3. **Set Properties**: Customize the properties of the menu items, such as **Text** (the display name) and **Name** (the identifier used in code).
4. **Handle Events**: Double-click on a menu item to create an event handler for the **Click** event. In the event handler, you can define the actions to be performed when the menu item is clicked.

Example:

```
Private Sub FileMenuItem_Click(sender As Object, e As EventArgs) Handles
FileMenuItem.Click
  MessageBox.Show("File menu clicked!")
End Sub
```

17. Enlist and explain various objectives of .NET frameworks.

The objectives of the .NET framework include:

1. **Interoperability**: To allow different programming languages to work together seamlessly, enabling developers to use the best language for their needs while still being able to integrate with other languages.
2. **Code Reusability**: To promote the reuse of code through the use of libraries and components, reducing development time and effort.
3. **Security**: To provide a secure environment for applications through built-in security features, including code access security and role-based security.
4. **Performance**: To enhance application performance through features like Just-In-Time (JIT) compilation, which optimizes code execution at runtime.
5. **Scalability**: To support the development of scalable applications that can handle increased loads and user demands without significant changes to the codebase.
6. **Ease of Development**: To simplify the development process with a rich set of libraries, tools, and frameworks that streamline common tasks and reduce complexity.

18. State and explain various statements used in VB.NET.

VB.NET supports several types of statements for controlling program flow and performing operations. Key statements include:

1. **Dim Statement**: Used to declare variables. Example:

```
Dim count As Integer
```

2. **If...Then...Else Statement**: Used for conditional execution. Example:

```
If count > 0 Then
  Console.WriteLine("Count is positive.")
Else
  Console.WriteLine("Count is not positive.")
End If
```

3. **For...Next Statement**: Used for looping a specific number of times. Example:

```
For i As Integer = 1 To 10
Console.WriteLine(i) Next
```

4. **While...End While Statement**: Used for looping while a condition is true. Example:

```
While count < 10
  count += 1
End While
```

5. **Select Case Statement**: Used for multi-way branching based on the value of an expression. Example:

```
Select Case dayOfWeek
  Case 1
    Console.WriteLine("Monday")
  Case 2
    Console.WriteLine("Tuesday")
  Case Else
    Console.WriteLine("Other day")
End Select
```

19. What are the HTML controls? Explain.

HTML controls in ASP.NET are server-side controls that render as HTML elements in the browser. They allow developers to create interactive web forms and applications. Key HTML controls include:

1. **TextBox**: Allows users to input text. It renders as an **<input type="text">** element in HTML.

```
<asp:TextBox ID="txtName" runat="server" />
```

2. **Button**: A clickable button that can trigger server-side events. It renders as an **<input type="button">** element.

```
<asp:Button ID="btnSubmit" runat="server" Text="Submit" />
```

3. **Label**: Displays text on the web page. It renders as a **** or **<label>** element.

```
<asp:Label ID="lblMessage" runat="server" Text="Hello, World!" />
```

4. **DropDownList**: A list of options from which users can select one. It renders as a **<select>** element.

```
<asp:DropDownList ID="ddlOptions" runat="server">
  <asp:ListItem Text="Option 1" Value="1" />
  <asp:ListItem Text="Option 2" Value="2" />
</asp:DropDownList>
```

5. **CheckBox**: Allows users to select or deselect an option. It renders as an **<input type="checkbox">** element.

```
<asp:CheckBox ID="chkAgree" runat="server" Text="I agree" />
```

20. Write steps to connect to a database using ADO.NET.

To connect to a database using ADO.NET, follow these steps:

1. **Import the Namespace**: Include the necessary ADO.NET namespaces in your code.

```
using System.Data;
using System.Data.SqlClient;
```

2. **Create a Connection String**: Define the connection string that specifies the database server, database name, and authentication details.

```
string connectionString =
"Server=myServerAddress;Database=myDataBase;User
Id=myUsername;Password=myPassword;";
```

3. **Create a Connection Object**: Instantiate a **SqlConnection** object using the connection string.

```
using (SqlConnection connection = new SqlConnection(connectionString))
{
  connection.Open(); // Open the connection
  // Perform database operations here
}
```

4. **Create a Command Object**: Create a **SqlCommand** object to execute SQL queries or stored procedures.

```
SqlCommand command = new SqlCommand("SELECT * FROM Users",
connection);
```

5. **Execute the Command**: Use the command object to execute the query and retrieve data.

```
SqlDataReader reader = command.ExecuteReader();
while (reader.Read())
{
  Console.WriteLine(reader["User Name"].ToString());
}
reader.Close(); // Close the reader
```

6. **Close the Connection**: Ensure the connection is closed after operations are complete, either explicitly or by using a **using** statement.

21. Explain Common Type Systems (CTS) and Common Language Specification (CLS).

- **Common Type System (CTS)**: The CTS is a standard that defines how types are declared, used, and managed in the .NET framework. It ensures that types created in different programming languages can interact with each other. The CTS provides a set of rules for data types, allowing for type safety and interoperability among languages.

- **Common Language Specification (CLS)**: The CLS is a subset of the CTS that defines a set of rules and guidelines to ensure that different .NET languages can work together. It specifies which features of the .NET framework are available to all languages, promoting consistency and compatibility. The CLS allows developers to create libraries and components that can be used across different .NET languages, enhancing code reusability and interoperability.

Q3. Programs.

1. Write a program to show a list of doctors visiting "Sahyadri Multispecialty Hospital."

Ans:-

```
using System;
using System.Collections.Generic;

class Program
{
   static void Main()
   {
      List<string> doctors = new List<string>
      {
         "Dr. John Smith - Cardiologist",
         "Dr. Emily Johnson - Neurologist",
         "Dr. Michael Brown - Orthopedic Surgeon",
         "Dr. Sarah Davis - Pediatrician"
      };

      Console.WriteLine("Doctors visiting Sahyadri Multispecialty Hospital:");
      foreach (var doctor in doctors)
      {
         Console.WriteLine(doctor);
      }
```

```
    }
}
```

2. Write a program to find prime numbers between 2 to 20.

Ans:-

```
using System;

class Program
{
    static void Main()
    {
        Console.WriteLine("Prime numbers between 2 and 20:");
        for (int num = 2; num <= 20; num++)
        {
            bool isPrime = true;
            for (int i = 2; i <= Math.Sqrt(num); i++)
            {
                if (num % i == 0)
                {
                    isPrime = false;
                    break;
                }
            }
            if (isPrime)
            {
                Console.WriteLine(num);
            }
        }
    }
}
```

3. Write a program in C# for multiplication of two numbers.

Ans:-

```
using System;
```

```
class Program
{
   static void Main()
   {
      Console.Write("Enter first number: ");
      int num1 = Convert.ToInt32(Console.ReadLine());
      Console.Write("Enter second number: ");
      int num2 = Convert.ToInt32(Console.ReadLine());
      int product = num1 * num2;
      Console.WriteLine($"The product of {num1} and {num2} is {product}.");
   }
}
```

4. Write a program in C# to calculate the area of a square.

Ans:-

```
using System;

class Program
{
   static void Main()
   {
      Console.Write("Enter the side length of the square: ");
      double side = Convert.ToDouble(Console.ReadLine());
      double area = side * side;
      Console.WriteLine($"The area of the square is {area}.");
   }
}
```

5. Write a VB.NET program to check if a given number is a palindrome or not.

Ans:-

```
Module Module1
   Sub Main()
```

```
    Console.Write("Enter a number: ")
    Dim number As String = Console.ReadLine()
    Dim reversed As String = StrReverse(number)

    If number = reversed Then
      Console.WriteLine("The number is a palindrome.")
    Else
      Console.WriteLine("The number is not a palindrome.")
    End If
  End Sub
End Module
```

6. Write a VB.NET program to print Yesterday's date on screen.

Ans:-

```
Module Module1
  Sub Main()
    Dim yesterday As DateTime = DateTime.Now.AddDays(-1)
    Console.WriteLine("Yesterday's date: " & yesterday.ToString("d"))
  End Sub
End Module
```

7. Write a VB.NET program to display Student ID, Student Name, Student Course, and Student Fees.

Ans:-

```
Module Module1
  Sub Main()
    Dim studentID As Integer = 1
    Dim studentName As String = "John Doe"
    Dim studentCourse As String = "Computer Science"
    Dim studentFees As Decimal = 1500.0D

    Console.WriteLine("Student ID: " & studentID)
    Console.WriteLine("Student Name: " & studentName)
    Console.WriteLine("Student Course: " & studentCourse)
```

```
    Console.WriteLine("Student Fees: " & studentFees)
  End Sub
End Module
```

8. Write a C# program to swap two numbers.

Ans:-

```
using System;

class Program
{
  static void Main()
  {
    Console.Write("Enter first number: ");
    int num1 = Convert.ToInt32(Console.ReadLine());
    Console.Write("Enter second number: ");
    int num2 = Convert.ToInt32(Console.ReadLine());

    // Swapping
    int temp = num1;
    num1 = num2;
    num2 = temp;

    Console.WriteLine($"After swapping: First number = {num1}, Second
number = {num2}");
  }
}
```

9. Write a VB.NET program to move the text “Pune University” continuously from left to right.

Ans:-

```
Imports System.Windows.Forms

 Public Class Form1
```

```
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        Timer1.Start()
    End Sub

    Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles
Timer1.Tick
        Label1.Left += 5
        If Label1.Left > Me.Width Then
            Label1.Left = -Label1.Width
        End If
    End Sub
End Class
```

10. Write a C# program for multiplication of a matrix.

Ans:-

```
using System;

class Program
{
    static void Main()
    {
        int[,] matrixA = { { 1, 2 }, { 3, 4 } };
        int[,] matrixB = { { 5, 6 }, { 7, 8 } };
        int[,] result = new int[2, 2];

        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 2; j++)
            {
                result[i, j] = 0;
                for (int k = 0; k < 2; k++)
                {
```

```
                result[i, j] += matrixA[i, k] * matrixB[k, j];
            }
        }
    }

    Console.WriteLine("Result of matrix multiplication:");
    for (int i = 0; i < 2; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            Console.Write(result[i, j] + " ");
        }
        Console.WriteLine();
    }
  }
}
```

11. Write a VB.NET program to find the max number among entered two numbers.

Ans:-

```
Module Module1
  Sub Main()
    Console.Write("Enter first number: ")
    Dim num1 As Integer = Convert.ToInt32(Console.ReadLine())
    Console.Write("Enter second number: ")
    Dim num2 As Integer = Convert.ToInt32(Console.ReadLine())

    Dim max As Integer = If(num1 > num2, num1, num2)
    Console.WriteLine("The maximum number is: " & max)
  End Sub
End Module
```

12. Write a VB.NET program to check whether the entered string is a palindrome or not.

Ans:-

```
Module Module1
  Sub Main()
    Console.Write("Enter a string: ")
    Dim input As String = Console.ReadLine()
    Dim reversed As String = StrReverse(input)

    If input.Equals(reversed, StringComparison.OrdinalIgnoreCase) Then
      Console.WriteLine("The string is a palindrome.")
    Else
      Console.WriteLine("The string is not a palindrome.")
    End If
  End Sub
End Module
```

13. Write a VB.NET program to accept a number from the user through an input box and display its multiplication table into a list box.

Ans:-

```
Module Module1
  Sub Main()
    Dim number As Integer = Convert.ToInt32(InputBox("Enter a number:"))
    Dim result As String = ""

    For i As Integer = 1 To 10
      result &= number & " x " & i & " = " & (number * i) & vbCrLf
    Next

    MessageBox.Show(result, "Multiplication Table")
  End Sub
End Module
```

14. Write a program in C# for the sum of two numbers.

Ans:-

```
using System;
```

```
class Program
{
   static void Main()
   {
      Console.Write("Enter first number: ");
      int num1 = Convert.ToInt32(Console.ReadLine());
      Console.Write("Enter second number: ");
      int num2 = Convert.ToInt32(Console.ReadLine());
      int sum = num1 + num2;
      Console.WriteLine($"The sum of {num1} and {num2} is {sum}.");
   }
}
```

15. Write a program in C# to reverse a given number.

Ans:-

```
using System;

class Program
{
   static void Main()
   {
      Console.Write("Enter a number: ");
      int number = Convert.ToInt32(Console.ReadLine());
      int reversed = 0;

      while (number > 0)
      {
         int digit = number % 10;
         reversed = reversed * 10 + digit;
         number /= 10;
      }
```

```
        Console.WriteLine($"Reversed number is: {reversed}");
    }
}
```

16. Write a C# program to find the area of a circle.

Ans:-

```
using System;

class Program
{
    static void Main()
    {
        Console.Write("Enter the radius of the circle: ");
        double radius = Convert.ToDouble(Console.ReadLine());
 double area = Math.PI * radius * radius;
        Console.WriteLine($"The area of the circle is {area}.");
    }
}
```

17. Write a VB.NET program to accept a character from the user and check whether it is a vowel or not.

Ans:-

```
Module Module1
    Sub Main()
        Console.Write("Enter a character: ")
        Dim ch As Char = Console.ReadLine()(0)

        If "aeiouAEIOU".Contains(ch) Then
            Console.WriteLine(ch & " is a vowel.")
        Else
            Console.WriteLine(ch & " is not a vowel.")
        End If
    End Sub
End Module
```

18. Write a program in C# to create a function to find the factorial of a given number.

Ans:-

```
using System;

class Program
{
   static void Main()
   {
      Console.Write("Enter a number: ");
      int number = Convert.ToInt32(Console.ReadLine());
      long factorial = Factorial(number);
      Console.WriteLine($"The factorial of {number} is {factorial}.");
   }

   static long Factorial(int n)
   {
      if (n == 0) return 1;
      return n * Factorial(n - 1);
   }
}
```

19. Write a VB.NET program to accept the details of an Employee (Eno, Ename, salary).

Ans:-

```
Module Module1
   Sub Main()
      Console.Write("Enter Employee Number: ")
      Dim eno As Integer = Convert.ToInt32(Console.ReadLine())
      Console.Write("Enter Employee Name: ")
      Dim ename As String = Console.ReadLine()
      Console.Write("Enter Salary: ")
      Dim salary As Decimal = Convert.ToDecimal(Console.ReadLine())
```

```
        Console.WriteLine("Employee Details:")
        Console.WriteLine("Employee Number: " & eno)
        Console.WriteLine("Employee Name: " & ename)
        Console.WriteLine("Salary: " & salary)
    End Sub
End Module
```

20. Write a C# program to find the Armstrong number of a given number.

Ans:-

```
using System;

class Program
{
    static void Main()
    {
        Console.Write("Enter a number: ");
        int number = Convert.ToInt32(Console.ReadLine());
        int sum = 0, temp = number, digits = number.ToString().Length;

        while (temp > 0)
        {
            int digit = temp % 10;
            sum += (int)Math.Pow(digit, digits);
            temp /= 10;
        }

        if (sum == number)
        {
            Console.WriteLine($"{number} is an Armstrong number.");
        }
        else
        {
```

```
        Console.WriteLine($"{number} is not an Armstrong number.");
    }
  }
}
```

21. Write a VB.NET program to display today's date on the screen.

Ans:-

```
Module Module1
  Sub Main()
    Console.WriteLine("Today's date: " & DateTime.Now.ToString("d"))
  End Sub
End Module
```

Q4. Short Notes.

1. Event Driven Programming

Event-driven programming is a programming paradigm in which the flow of the program is determined by events such as user actions (mouse clicks, key presses) or messages from other programs. In this model, the program responds to events by executing specific event handlers. This approach is commonly used in graphical user interfaces (GUIs) and interactive applications, allowing for a more responsive user experience.

2. JIT Compilers

Just-In-Time (JIT) compilers are part of the Common Language Runtime (CLR) in the .NET framework that convert Intermediate Language (IL) code into native machine code at runtime. This process allows for optimizations specific to the current execution environment, improving performance. JIT compilation occurs the first time a method is called, and the compiled code is cached for subsequent calls, enhancing execution speed.

3. Method Overloading

Method overloading is a feature in C# that allows multiple methods in the same class to have the same name but different parameters (different type,

number, or order of parameters). This enables developers to create methods that perform similar functions but with different input types or quantities, enhancing code readability and usability.

Example:

```
public class MathOperations
{
   public int Add(int a, int b) { return a + b; }
   public double Add(double a, double b) { return a + b; }
   public int Add(int a, int b, int c) { return a + b + c; }
}
```

4. Object-Oriented Concept in C#

C# is an object-oriented programming language that supports the four main principles of OOP:

1. **Encapsulation**: Bundling data (attributes) and methods (functions) that operate on the data into a single unit called a class. Access to the data is restricted through access modifiers (public, private, protected).
2. **Inheritance**: A mechanism that allows one class (derived class) to inherit properties and methods from another class (base class), promoting code reusability.
3. **Polymorphism**: The ability to present the same interface for different underlying data types. It allows methods to be defined in multiple forms, typically through method overriding and overloading.
4. **Abstraction**: Hiding complex implementation details and showing only the essential features of an object. This is achieved through abstract classes and interfaces.

5. Validation Control in ASP.NET

Validation controls in ASP.NET are used to ensure that user input meets specific criteria before it is processed. They provide a way to validate data entered into web forms, enhancing data integrity and user experience. Common validation controls include:

- **RequiredFieldValidator**: Ensures that a user does not skip a required input field.
- **RangeValidator**: Checks that a value falls within a specified range.
- **RegularExpressionValidator**: Validates input against a specified regular expression pattern.
- **CustomValidator**: Allows for custom validation logic defined by the developer.

6. Data Types in VB.NET

VB.NET supports several data types that can be categorized into two main groups:

1. **Value Types**: Store the actual data directly. Examples include:
    - **Integer**: Represents a 32-bit signed integer.
    - **Double**: Represents a double-precision floating-point number.
    - **Boolean**: Represents a true or false value.
2. **Reference Types**: Store a reference to the actual data. Examples include:
    - **String**: Represents a sequence of characters.
    - **Object**: The base type from which all other types derive.

7. Inheritance

Inheritance is a fundamental object-oriented programming concept that allows a class (derived class) to inherit properties and methods from another class (base class). This promotes code reusability and establishes a hierarchical relationship between classes. In C#, inheritance is implemented using the **:** symbol.

Example:

```
public class Animal // Base class
{
   public void Eat()
   {
      Console.WriteLine("Eating...");
```

```
    }
}

public class Dog : Animal // Derived class
{
    public void Bark()
    {
        Console.WriteLine("Barking...");
    }
}

// Usage
Dog myDog = new Dog();
myDog.Eat(); // Inherited method
myDog.Bark(); // Own method
```

8. ASP.NET Server Controls

ASP.NET server controls are components that run on the server and generate HTML content sent to the client. They provide a way to create dynamic web pages with rich functionality. Key features include:

- **State Management**: Server controls maintain their state across postbacks, allowing for persistent data.
- **Event Handling**: Server controls can handle events like button clicks, enabling interactive web applications.
- **Rich Controls**: They include various types such as GridView, ListView, and FormView, which simplify data binding and display.

9. Constructor and Destructor

In C#, a constructor is a special method that is called when an instance of a class is created. It initializes the object's properties and allocates resources. Constructors can be overloaded to provide different ways of initializing an object. A destructor, on the other hand, is called when an object is destroyed,

allowing for cleanup of resources. In C#, destructors are defined using the **~** symbol.

Example:

```
public class SampleClass
{
   public SampleClass() // Constructor
   {
      Console.WriteLine("Constructor called");
   }

   ~SampleClass() // Destructor
   {
      Console.WriteLine("Destructor called");
   }
}
```