Before reading past below instructions:

1. Create an account in Github using your name in this format: lastname\_firstname\_section
2. Request access to [Lycevm<3Alabang · GitHub](https://github.com/Lycevm-3Alabang)
3. Upload this file ON YOUR GITHUB ACCOUNT with answer under the title / file name : E3\_Assessment\_\_[Section]\_[LastnameFirstName]  
   example: E3\_Assessment\_\_BSCS32E1\_AlamoNinoFrancisco

Help: [Get started with GitHub documentation - GitHub Docs](https://docs.github.com/en/get-started)

**Sample Assessment for Introduction to Programming**

This assessment is designed to evaluate your understanding of basic programming concepts in C#, HTML, CSS, and JavaScript.

Instructions: Read each question carefully and provide complete and clear answers. Avoid multiple-choice format responses. Focus on demonstrating your understanding through code, explanations, and discussions.

**Part 1: C# (30 points)**

(10 points) Write a C# program that calculates the area of a triangle given its base and height. Include user input for both values and display the calculated area.

using System;

class Triangle

{

    static void Main()

    {

        Console.WriteLine("Enter the base of the triangle:");

        double baseLength = double.Parse(Console.ReadLine());

        Console.WriteLine("Enter the height of the triangle:");

        double height = double.Parse(Console.ReadLine());

        double area = 0.5 \* baseLength \* height;

        Console.WriteLine($"The area of the triangle with base {baseLength} and height {height} is: {area}");

    }

}

**(10 points) Declare an array of 5 integers and fill it with values based on a user-defined formula (e.g., n^2). Then, print the largest element in the array.**

using System;

namespace LargestElementInArray

{

    class Program

    {

        static void Main(string[] args)

        {

            int[] numbers = { 12, 6, 51, 23, 9 };

            Console.WriteLine("Numbers within the Array:");

            foreach (int number in numbers)

            {

                Console.WriteLine(number);

            }

            int largest = numbers[0];

            for (int i = 1; i < numbers.Length; i++)

            {

                if (numbers[i] > largest)

                {

                    largest = numbers[i];

                }

            }

            Console.WriteLine("The largest element in the array is: " + largest);

            Console.ReadKey();

        }

    }

}

**(10 points) Implement a simple for loop that iterates from 1 to 10 and prints each number along with its square root.**

using System;

class Loop

{

    static void Main()

    {

        for (int i = 1; i <= 10; i++)

        {

            double squareRoot = Math.Sqrt(i);

            Console.WriteLine($"Number: {i}, Square Root: {squareRoot}");

        }

    }

}

**Part 2: HTML, CSS, and JavaScript (30 points)**

**HTML (10 points):** You are provided with the following incomplete HTML code snippet:

**HTML**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>My Website</title>**

**</head>**

**<body>**

**<h1>Welcome to...</h1>**

**<p>This is a paragraph...</p>**

**<ul>**

**<li>Item 1</li>**

**<li>Item 2</li>**

**</ul>**

**</body>**

**</html>**

Complete the code snippet by adding the following elements:

An image within the <body> tag with a relevant src attribute.

An ordered list (<ol>) with three items.

A hyperlink within a <p> tag that points to an external website.

A CSS styling rule using an inline style attribute to change the font color of the <h3> heading.

CSS (10 points): Create a CSS stylesheet that defines the following styles:

Change the background color of the body element to light blue.

Apply a padding of 20px to all headings (h1, h2, h3).

Set the font size of the <p> tag to 14px.

Make the list items (li) have a bullet point style instead of the default numbers.

**JavaScript (10 points):** Write a JavaScript function that takes a number as input and returns a string indicating whether the number is even or odd. Then, add a button to your HTML page that, when clicked, calls this function and displays the result (even or odd) in a paragraph element below the button.

**HTML:**

<!DOCTYPE html>

<html>

<head>

    <title>My Website</title>

    <link rel="stylesheet" href="styles.css">

    <script src="script.js"></script>

</head>

    <body>

        <h1>Welcome to...</h1>

        <p> This is a paragraph...</p>

        <img src="" alt="This is the image">

        <p> unordered list: </p>

            <ul>

                <li> Peenoise Podcast </li>

                <li> Pampamilya Premium </li>

                <li> PaoLUL </li>

            </ul>

        <p> ordered list: </p>

            <ol>

                <li> YouTube </li>

                <li> FaceBook </li>

                <li> X (Former Twitter) </li>

            </ol>

        <p><a href="https://www.youtube.com">Click This </a> to redirect on YouTube.</p>

        <h3 class="Heading"> This is Heading 3 </h3>

        <button onclick="displayResult()"> Enter a Number </button>

        <p id="result"></p>

    </body>

</html>

**CSS:**

body {

    background-color: lightblue;

}

h1, h3 {

    padding: 20px;

}

p.tags {

    font-size: 14px;

}

ol {

    list-style-type: circle;

}

.Heading {

    color: red;

}

**JavaScripts:**

function checkEvenOrOdd(number) {

    return number % 2 === 0 ? "even" : "odd";

}

function displayResult() {

    var inputNumber = parseInt(prompt("Enter a number: "));

    var result = checkEvenOrOdd(inputNumber);

    document.getElementById("result").innerText = "The Number is " + result + ".";

}

**Part 3: Essay Question (40 points)**

Discuss the importance of object-oriented programming (OOP) concepts in software development. Explain the key principles of OOP (encapsulation, inheritance, polymorphism, abstraction) and provide examples of how they can be used to create more efficient, maintainable, and reusable code. Include real-world scenarios or cases where OOP is particularly valuable.

**Answer:**

Object-oriented programming (OOP) is a programming paradigm that revolves around the concept of “object”, which are instances of classes that encapsulate data (attributes) and behavior (methods).

OOP provides several key concepts that help in designing and building software systems:

**Encapsulation:** Encapsulation refers to the bundling of data (attributes) and methods that operate on the data into a single unit, i.e., a **class**.

Example: In a banking application ‘BankAccount’ class could encapsulate the account balance and provide methods like ‘deposit’ and ‘withdraw’ to modify the balance, ensuring that the balance is always updated correctly.

**Inheritance:** Inheritance is a mechanism where a new class (derived class or subclass) is created from an existing class (base class or superclass), inheriting its attributes and methods. This allows for code reuse and the creation of a hierarchical relationship between classes.

Example: In a vehicle management system, you have a ‘Vehicle’ class as the base class, and then derive classes like ‘Car’, ‘Truck’, and ‘Motorcycle’ from it, inheriting common attributes and methods such as ‘start’ and ‘stop’.

**Polymorphism:** Polymorphism is where the object can take on multiple forms. In OOP, polymorphism allows objects of different classes to be treated as objects of a common superclass. Polymorphism is like having a remote control that can operate different devices.

Example: In a drawing application, the ‘Shape” superclass with a ‘draw’ method. The ‘Circle’ and ‘Rectangle’ classes, which inherit from ‘Shape’, can be override the ‘draw’ method to provide their own implementation, allowing the same ‘draw’ method to be used to draw different shapes.

**Abstraction:** Abstraction is the concept of simplifying complex systems by modeling classes appropriate to the problem and working at the most relevant level of inheritance for a particular aspect of the problem. Meaning by this in abstraction you don’t need to know how it works internally; you just need to know how to use it.

Example: In a simulation game, ‘Animal’ superclass with abstract methods like ‘eat and ‘move’. Concrete classes like ‘Lion’ and ‘Elephant’ would implement these methods according to their specific behavior, abstracting away the details of how each animal behaves.

Points Distribution:

Each part carries equal weight (30 points).

Code clarity, functionality, and explanations will be considered in grading.

The essay question focuses on understanding and application of OOP concepts.