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 AreaOfTriangle

{

    static void Main(string[] args)

    {

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

        double baseLength = Convert.ToDouble(Console.ReadLine());

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

        double height = Convert.ToDouble(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 = { 4, 20, 18, 45, 76 };

            Console.WriteLine("Array Numbers:");

            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 NumberLoop

{

    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="numscript.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> Coco Code </li>

                <li> Pixel Art  </li>

                <li> Godot 2D Pixel Game </li>

            </ul>

        <p> ordered list: </p>

            <ol>

                <li> YouTube </li>

                <li> FaceBook </li>

                <li> Youtube </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>

**JavaScript**

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 + ".";

}

**CSS**

body {

    background-color: peachpuff;

}

h1, h3 {

    padding: 20px;

}

p.tags {

    font-size: 14px;

}

ol {

    list-style-type: circle;

}

.Heading {

    color: rgba(223, 202, 85, 0.918);

}

**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.

OOP is very important in software development for the cause of providing clear and efficient way of organizing and managing the codes. Doing this, your project leads to more maintainable, reusable, and scalable software solutions.

Key principles of OOP:

Encapsulation: this involves bundling data and methods that operate on the data into a single unit called an object.

Example: Consider an email system where an **Email** class encapsulates the details of an email message, including sender, recipient, subject, and body. Methods like **send()** and **read()** provide controlled access to these details, ensuring that the email's internal state is properly managed and hidden from external manipulation.

Inheritance: this enables a class to inherit properties and behaviors from another class. This key allows you to use code reusability and facilitates the creation of a hierarchy of classes, where subclasses can extend or specialize in the function of their superclass.

Example: Employee Management System - In an employee management system, a base class **Employee** can define attributes like **name**, **employee\_id**, and methods like **calculate\_salary()**. Subclasses like **Manager** and **Developer** can inherit from **Employee** and override methods or add additional functionalities specific to each role.

Polymorphism: It allows objects of different classes to be treated as objects of a common superclass. It enables flexibility in code design by allowing functions or methods to work with objects of multiple types and to behave differently based on the type of object they are operating on.

Example: Consider a program that calculates the area of various shapes. We can have a common interface **Shape** with a method **calculate\_area()**. Different shape classes like **Circle**, **Rectangle**, and **Triangle** can implement this interface and provide their own implementations of **calculate\_area()**. Polymorphism allows the program to call **calculate\_area()** on any shape object, and the appropriate method will be invoked based on the shape type.

Abstraction: It 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: Consider a universal remote control that can operate various electronic devices like TVs, DVD players, and sound systems. The remote-control abstracts away the complexities of each device's operation by providing a simplified interface with buttons like **power**, **volume up**, **volume down**, and **channel change**. Users interact with the remote control without needing to understand the inner workings of each device.

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.