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.

**Answer:**

**using System;**

**class Program**

**{**

**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.

**Answer:**

**using System;**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**int[] array = new int[5];**

**for (int i = 0; i < array.Length; i++)**

**{**

**array[i] = (i + 1) \* (i + 1);**

**}**

**Console.WriteLine("Array:");**

**foreach (int num in array)**

**{**

**Console.Write(num + " ");**

**}**

**Console.WriteLine();**

**int max = array[0];**

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

**{**

**if (array[i] > max)**

**{**

**max = array[i];**

**}**

**}**

**Console.WriteLine("Largest element in the array: " + max);**

**}**

**}**

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

**Answer:**

**using System;**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**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.

**Answer:**

**<!DOCTYPE html>**

**<html>**

**<head>**

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

**</head>**

**<body>**

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

**<!-- A hyperlink within a <p> tag that points to an external website -->**

**<p>This is a paragraph with a <a href="https://www.example.com">link</a> to an external website. </p>**

**<ul>**

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

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

**</ul>**

**<!-- An image within the body tag with a relevant src attribute -->**

**<img src="example.jpg" alt="Example Image">**

**<!-- An ordered list (<ol>) with three items -->**

**<ol>**

**<li>First item</li>**

**<li>Second item</li>**

**<li>Third item</li>**

**</ol>**

**<!-- CSS styling rule using an inline style attribute to change the font color of the <h3> heading -->**

**<h3 style="color: blue;">Heading 3</h3>**

**</body>**

**</html>**

**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.

**Answer:**

/\* CSS stylesheet \*/

body {

background-color: lightblue; /\* Change the background color of the body element to light blue \*/

}

h1, h2, h3 {

padding: 20px; /\* Apply a padding of 20px to all headings (h1, h2, h3) \*/

}

p {

font-size: 14px; /\* Set the font size of the <p> tag to 14px \*/

}

li {

list-style-type: disc; /\* 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.

**Answer:**

function checkEvenOrOdd(number) {

if (number % 2 === 0) {

return "even";

} else {

return "odd";

}

}

function displayResult() {

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

var result = checkEvenOrOdd(userInput);

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

}

<!DOCTYPE html>

<html>

<head>

<title>Even or Odd Checker</title>

</head>

<body>

<button onclick="displayResult()">Check Even/Odd</button>

<p id="result"></p>

<!-- JavaScript code -->

<script>

// Paste JavaScript code here

</script>

</body>

</html>

**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:**

Since they encourage code structure and reusability, object-oriented programming (OOP) concepts are crucial to the software development process. Encapsulation improves data security and lowers code complexity by enabling the bundling of data and operations within a class. While polymorphism allows for flexibility by allowing objects of different kinds to be treated interchangeably, inheritance makes code reuse easier by allowing new classes to be created that inherit characteristics and behavior from older ones. This results in more efficient and flexible code. When creating complicated systems like banking apps, OOP is especially helpful since classes like Account, SavingsAccount, and CheckingAccount may inherit common features and encapsulate unique data and methods, which facilitates maintainability and scalability.

**Key Principles:**

1. **Encapsulation:** To provide a class control over who may access its own data and activity, encapsulation involves grouping methods (behavior) and data (attributes) within the class. Data integrity and security are enhanced by this method of keeping a class's internal information hidden from the public. Take into consideration a class that represents a bank account, for instance. To guarantee that the amount is accessible and updated in a regulated manner, we can encapsulate the account balance and the procedures for making deposits and withdrawals within the class.
2. **Inheritance:** A new class (derived class or subclass) can inherit properties and functions from an already existing class (base class or superclass) through inheritance. This facilitates the development of hierarchical links between classes and encourages the reuse of code. In a university administration system, for example, we might have a base class called Person that has properties like name and age. We could then derive classes from individuals, such as students and professors, that have their own unique attributes and methods added to them.
3. **Polymorphism:** Polymorphism facilitates flexibility and code expansion by allowing objects of distinct classes to be considered as objects of a shared superclass. There are two kinds of polymorphism: runtime polymorphism (which is obtained via method overriding) and compile-time polymorphism (which is accomplished through overloading methods). For instance, we may have a Shape superclass with a draw function in a drawing application. The draw method can be overridden by subclasses such as Circle and Rectangle to offer their own implementations, enabling the usage of the same draw method for several forms.
4. **Abstraction:** The approach of just revealing to users the components that are absolutely essential while hiding the complex technical details of the system is known as abstraction. This helps to make complex large-scale system administration easier and more effective. Consider looking at a simulation program for cars. Users may engage with high-level activities like changing gears or starting the engine without needing to understand the intricate inner workings of the engine or transmission system.

**Real-world scenarios or cases where OOP is particularly valuable:**

1. **Software Development in Large Organizations:** Better code structure and administration are made possible by OOP in large organizations when numerous teams are working on distinct software project modules or components.
2. **Web Development:** To create scalable and maintainable online programs, web developers frequently utilize OOP concepts. Developers can design modular, reusable components like models, views, and controllers by using OOP-based frameworks like Ruby on Rails (Ruby), Django (Python), and Laravel (PHP). Managing intricate web apps with plenty of features and functionalities is made simpler as a result.
3. **Game Development:** Since game developers must oversee several game objects with various interactions and behaviors, object-oriented programming (OOP) is crucial. OOP principles form the foundation of game engines such as Unity and Unreal Engine, which give developers the ability to build and modify game objects, specify their behaviors through polymorphism and inheritance, and arrange game logic in an organized fashion. This makes it easier to create rich and interactive gaming experiences and makes game development more effective.
4. **Financial and Banking Applications:** To model intricate financial instruments, transactions, and business regulations, OOP concepts are used in banking and financial applications. To guarantee data confidentiality, integrity, and regulatory compliance, classes that represent accounts, transactions, loans, and investments can be created with encapsulated data and behavior. Another benefit of OOP is that it makes it easier to create modular, extensible software systems that may change to meet evolving legal and corporate needs.
5. **Embedded Systems Programming:** Code modularity and portability are encouraged by OOP, a crucial tool in embedded systems programming that makes it possible to design and produce firmware for a wide range of electrical devices.

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.