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.

Console.Write("Enter Base of Triangle: ");

int baseT = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter Height of Triangle: ");

int heightT = Convert.ToInt32(Console.ReadLine());

int areaT = (baseT \* heightT) / 2;

Console.Write("Area of Triangle is " + areaT);

**(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.**

int[] num = {1, 2, 3, 4, 5};

int max = num[0];

for (int i = 0; i < num.Length; i++){

    num[i] \*= 10;

    max = Math.Max(max, num[i]);

}

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

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

for (int i = 1; i <= 10; i++){

    double sqrRt = Math.Sqrt(i);

    Console.WriteLine("Number " + i + " with square root of " + sqrRt);

}

**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.

<!DOCTYPE html>

<html>

<head>

  <title>My Website</title>

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

</head>

<body>

  <h1>Welcome to...</h1>

  <p>This is a paragraph <a href="https://youtube.com">pointing to an external website.</a></p>

  <ul>

    <li>Item 1</li>

    <li>Item 2</li>

  </ul>

  <h3 style="color: red;">YouTube</h3>

  <img src="https://clipartcraft.com/images/youtube-icon-clipart-video-8.png" style="height: 100px; width: 100px;">

  <ol>

    <li>Item 1</li>

    <li>Item 2</li>

    <li>Item 3</li>

  </ol>

  </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.

body {

  background-color: lightblue;

}

h1, h2, h3 {

  padding: 20px;

}

p {

  font-size: 14px;

}

li {

  list-style-type: disc;

}

**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.

  <p>Enter a number: </p>

  <input type="number" id="number">

  <button onclick="checkEvenOdd()">Check</button>

  <p id="result"></p>

  <script>

    function checkEvenOdd(){

      const n = document.getElementById("number").value;

      if (n % 2 === 0) {

        document.getElementById("result").textContent = "The number is Even!";

        return;

      } else {

        document.getElementById("result").textContent = "The number is Odd!";

        return;

      }

    }

  </script>

**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 important in Software Development since it offer several benefits and advantages:

1. OOP breaks down problems and projects into smaller, self-contained objects which encapsulates both attributes and their methods. This modularity promotes reusability which, in other, newer projects, saves time and effort.
2. OOP is inherently scalable; by adding new objects, developers can expand existing functionalities without compromising the main design.
3. OOP provides a standardized approach to software development, making it easier for multiple developers to work on the same project simultaneously. This leads to more efficient development processes since it facilitates clearer communication and collaboration.
4. OOP promotes individuality between objects which results in flexibility for modifications and adaptations to changing requirements. This also gives way to inheriting properties and behaviours from existing ones, further enhancing reusability and adaptability.

Key Principles of OOP:

1. Encapsulation
   1. This principle groups attributes and methods within objects, creating self-contained units. This also results in protection of data integrity and promoting data hiding.
   2. **Example**: A StudentAccount object encapsulates student number (data) and methods like adding and dropping subjects (methods).
2. Inheritance
   1. This allows creating subclasses that inherit properties and behaviours from existing ones. This promotes reusability and reduces redundancy.
   2. **Example:** A Student class with attributes like name and subclasses Subject and Grade which inherits these but adds specific behaviours.
3. Polymorphism
   1. Objects of different classes can respond to the same method call but in different ways based on their specific implementations.
   2. **Example:** A StudentInfo method called on a Student object gives the student’s subjects, while being called on a Professor gives the student’s grades.
4. Abstraction
   1. Focuses on essential properties and behaviours of objects, hiding unnecessary details which promotes code understandability and reduces complexity.
   2. **Example:** A StudentInfo class with an abstract Grade method. It may provide a common interface for student information but it can be used by different objects such as StudentGrade and SubjectGrade.

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.