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

{

// Write a program to find the area of a triangle

static double TriangleArea(double a, double b)

{

double Area;

Area = (a \* b) / 2; // A is a for the Base and B is for the height when the system multiply it 3x4 we got 12 divided it into 2 it will be 6

return Area;

}

public static void Main(string[] args)

{

double a, b; // double means that for every root there is a decimal same thing as float for the other languages int we use it for fixed number such as 1,2,3,4,5etc without the decimal numbers

Console.WriteLine("Enter Base Of Triangle");

// In here the user will type in a numbers they would like for the base of the triangle

a = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("Enter Height Of Triangle");

// now here the user inputs the height of the triangle

b = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("Area of Triangle: {0}", Math.Round(TriangleArea(a, b), 2));

Console.ReadLine(); //Now here is the final process when the system or computer reads the equation it shows us the answer by multiplying and dividing them into 2 saying Area of Triangle is = Answer

}

}

**(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:**

**# Declare an array of 5 integers and fill it with values based on the formula n^2**

**array = [n\*\*2 for n in range(5)] # This will create [0, 1, 4, 9, 16]**

**# Print the array**

**print("Array:", array)**

**# Find and print the largest element in the array**

**largest\_element = max(array)**

**print("Largest element:", largest\_element)**

**In this example, the array is [0, 1, 4, 9, 16], and the largest element, as printed by the script, is 16.**

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

**Answer:**

**import math**

**for i in range(1, 11): # Iterates from 1 to 10 inclusive**

**print(f"Number: {i}, Square Root: {math.sqrt(i):.2f}")**

**In this code:**

**range(1, 11) generates numbers from 1 to 10.**

**math.sqrt(i) calculates the square root of the current number i.**

**The print statement uses an f-string to format the output, where {i} is replaced by the current number and {math.sqrt(i):.2f} is replaced by its square root rounded to two decimal places for better readability.**

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

**Answer:**

**<!DOCTYPE html>**

**<html>**

**<head>**

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

**<link rel="stylesheet" href="style.css"> <!-- Link to the CSS file -->**

**</head>**

**<body>**

**<h1>Welcome to My Website</h1>**

**<p>This is a paragraph introducing the content of the site.</p>**

**<!-- Adding an image -->**

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

**<ul>**

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

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

**</ul>**

**<!-- Adding an ordered list -->**

**<ol>**

**<li>First Ordered Item</li>**

**<li>Second Ordered Item</li>**

**<li>Third Ordered Item</li>**

**</ol>**

**<!-- Adding a hyperlink -->**

**<p>Visit <a href="https://www.example.com" target="\_blank">Example.com</a> for more information.</p>**

**<!-- Adding an h3 heading with inline CSS for font color -->**

**<h3 style="color: red;">An Important Announcement</h3>**

**</body>**

**</html>**

**CSS (10 points)**

**Next, we'll create a CSS stylesheet (style.css) to define the specified styles.**

**css**

**Copy code**

**/\* Change the background color of the body \*/**

**body {**

**background-color: lightblue;**

**}**

**/\* Apply padding to all headings \*/**

**h1, h2, h3 {**

**padding: 20px;**

**}**

**/\* Set the font size of the paragraph tags \*/**

**p {**

**font-size: 14px;**

**}**

**/\* Make list items have a bullet point style \*/**

**ul li {**

**list-style-type: disc; /\* This is actually the default for unordered lists \*/**

**}**

**ol li {**

**list-style-type: decimal; /\* This is the default for ordered lists, ensuring they are numbered \*/**

**}**

**JavaScript (Optional)**

**Though JavaScript wasn't explicitly requested, here's a simple example demonstrating how to add a basic interactive feature using JavaScript. Suppose we want to show an alert when the user clicks on the "An Important Announcement" heading.**

**Add this script tag just before the closing </body> tag in your HTML file:**

**html**

**Copy code**

**<script>**

**document.querySelector('h3').addEventListener('click', function() {**

**alert('Thank you for clicking the announcement!');**

**});**

**</script>**

**Putting It All Together**

**Ensure the HTML file is saved (e.g., index.html) and the CSS content is saved in a file named style.css in the same directory. The HTML file includes a link to the CSS file for styling and a simple JavaScript snippet for interactivity. This setup demonstrates basic web development practices by combining HTML for structure, CSS for styling, and JavaScript for interactivity.**

**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.

Answer:

<!DOCTYPE html>

<html>

<head>

<title>My Website</title>

<link rel="stylesheet" href="styles.css"> <!-- Linking the external CSS stylesheet -->

</head>

<body>

<h1>Welcome to My Website</h1>

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

<img src="image.jpg" alt="A relevant alt text"> <!-- Add your image source -->

<ul>

<li>Item 1</li>

<li>Item 2</li>

</ul>

<!-- Adding an ordered list -->

<ol>

<li>Ordered Item 1</li>

<li>Ordered Item 2</li>

<li>Ordered Item 3</li>

</ol>

<!-- Adding a heading with inline CSS for font color -->

<h3 style="color: deeppink;">This is a colorful heading</h3>

</body>

</html>

In this snippet:

An image is added with a placeholder src="image.jpg"—you'll need to replace "image.jpg" with the actual path to your image.

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

A hyperlink (<a href="...">) is embedded within a paragraph (<p>).

An <h3>

**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:

First, let's write the JavaScript function. You can include this script directly in your HTML file within <script> tags or in a separate .js file that you link to your HTML.

javascript

Copy code

function checkEvenOdd(number) {

if (number % 2 === 0) {

return "even";

} else {

return "odd";

}

}

Step 2: HTML Modification

Now, modify your HTML to include a button and a paragraph. You'll also need an input field to allow the user to enter a number. Here's how you can integrate these elements into your HTML:

html

Copy code

<!DOCTYPE html>

<html>

<head>

<title>My Website</title>

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

</head>

<body>

<!-- Your existing HTML content -->

<!-- Adding an input field for the number -->

<input type="number" id="numberInput" placeholder="Enter a number">

<!-- Adding a button -->

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

<!-- Adding a paragraph to display the result -->

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

<script>

function displayResult() {

const number = document.getElementById('numberInput').value;

const result = checkEvenOdd(number);

document.getElementById('result').innerText = `The number is ${result}.`;

}

function checkEvenOdd(number) {

if (number % 2 === 0) {

return "even";

} else {

return "odd";

}

}

</script>

</body>

</html>

In this code:

An <input> element allows the user to enter a number.

A <button> is added, and its onclick attribute is set to call the displayResult() function when clicked.

A <p> element with an id of "result" is where the function's output will be displayed.

The displayResult() function reads the value from the input field, calls checkEvenOdd() with that value, and updates the content of the paragraph to show whether the number is even or odd.

This setup provides a simple user interface for checking if a number is even or odd, integrating HTML, CSS, and JavaScript to achieve the desired functionality.

**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 uses "objects" – data structures consisting of data fields and methods together with their interactions – to design applications and computer programs. It offers a robust and clear modular structure for programs which makes it good for defining abstract data types where implementation details are hidden and the unit of code is modulated around the data or objects.

Key Principles of OOP

1. Encapsulation

Encapsulation is the mechanism of hiding the internal details or mechanics of how an object does something for security reasons. It involves bundling the data (attributes) and methods (functions) that operate on the data into a single unit called a class and restricting access to some of the object's components. This can be achieved using access modifiers like private, protected, and public.

Example: In a banking system, an account class encapsulates the balance attribute (data) and deposit, withdraw methods (behaviors) necessary to manage an account, while preventing direct access to the balance from the outside.

2. Inheritance

Inheritance allows a class to inherit properties and methods from another class. The class which inherits the properties is known as the subclass (or derived class), and the class whose properties are inherited is known as the superclass (or base class). This helps in code reusability and method overriding.

Example: In a vehicle management system, a 'Car' class can inherit from a 'Vehicle' class. If 'Vehicle' has attributes like weight and speed, 'Car' will inherit these attributes and can also have additional features specific to it, such as trunk size.

3. Polymorphism

Polymorphism allows objects of different classes to be treated as objects of a common superclass. It's the ability of a single function or method to work in different ways based on the input or the class that invokes it. There are two types: compile-time (method overloading) and runtime (method overriding).

Example: In a graphic system, a shape class could have a method draw(). The subclasses such as circle, square, triangle can override the draw method to implement their version of draw, but a user can call draw on any shape instance similarly.

4. Abstraction

Abstraction involves hiding complex implementation details and showing only the necessary features of the object. It helps in reducing programming complexity and effort. It is achieved using abstract classes and interfaces.

Example: In a software for managing employees, an Employee class can provide an abstract base with common attributes like name, id, and methods like calculateSalary(). Specific types of employees can extend the Employee class with concrete implementations.

Importance and Real-World Scenarios

Software Maintenance and Evolution: With OOP, software is more modular, easier to understand, and modify. Since changes are localized, it reduces the risk of side effects affecting other parts of the system. This is crucial in large-scale enterprise systems where continuous software evolution and maintenance are required.

Reusability: Through inheritance and polymorphism, OOP facilitates the reuse of code that has already been tested and implemented, reducing redundancy and effort. Frameworks and libraries are prime examples where OOP's reusability aspect shines, allowing developers to build applications faster by using existing modules.

Scalability: OOP principles make it easier to scale applications and add new functionalities on top of the existing ones without breaking the codebase. This is essential in developing games, enterprise applications, and systems where new requirements are continuously integrated.

Development Efficiency: Encapsulation and abstraction make complex systems more manageable by breaking them down into smaller, more digestible objects. This modularity enables teams to work on different components in parallel, significantly improving development speed and efficiency.

Real-World Modeling: OOP allows developers to model real-world scenarios more effectively. Objects can represent real-life entities, making the code more intuitive and aligned with how humans perceive the world. This is particularly valuable in simulations, game development, and systems that require a close representation of real-world entities.

In conclusion, the principles of OOP provide a framework for structuring programs that are clear, reusable, and scalable. These principles are not just academic; they have practical implications in software development, significantly impacting the quality, maintenance, and lifecycle of applications.

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.