De Leon, Jian Paolo C.

IT ELECTIVE 3

BSIT 32E2

**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.Write("Enter the base length 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;

// Display the result

Console.WriteLine("The area of the triangle 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;

class ArrayExample

{

static void Main()

{

// Initialize an array with values based on the formula n^2

int[] numbers = { 1, 4, 9, 16, 25 };

// Find and print the largest element

int max = numbers[0];

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

{

if (numbers[i] > max)

{

max = numbers[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.**

using System;

class SquareRootExample

{

static void Main()

{

Console.WriteLine("Number\tSquare Root");

Console.WriteLine("-------------------");

PrintSquareRoots(1, 10);

}

static void PrintSquareRoots(int start, int end)

{

for (int i = start; i <= end; i++)

{

double squareRoot = Math.Sqrt(i);

Console.WriteLine($"{i}\t{squareRoot}");

}

}

}

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

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

<!DOCTYPE html>

<html>

<head>

<title>My Website</title>

<style>

body {

background-color: lightblue;

}

h1, h2, h3 {

padding: 20px;

}

p {

font-size: 14px;

}

ul {

list-style-type: none;

}

</style>

</head>

<body>

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

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

<img src="path/to/your/image.jpg" alt="Description of the image">

<ul>

<li>Item 1</li>

<li>Item 2</li>

<li>Item 3</li>

</ul>

<p><a href="https://www.external-website.com" target="\_blank">Visit the external website</a></p>

<h3 style="color: red;">Your Heading</h3>

</body>

</html>

**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.**

<!DOCTYPE html>

<html>

<head>

<title>Even or Odd Checker</title>

<style>

body { text-align: center; margin-top: 50px; font-family: Arial, sans-serif; }

button, input { padding: 10px; font-size: 16px; cursor: pointer; }

p { font-size: 18px; margin-top: 20px; }

</style>

</head>

<body>

<h1>Even or Odd Checker</h1>

<label for="numberInput">Enter a number:</label>

<input type="number" id="numberInput">

<button onclick="checkEvenOrOdd()">Check Number</button>

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

<script>

function checkEvenOrOdd() {

var number = document.getElementById("numberInput").value;

var result = (number % 2 === 0) ? "Even" : "Odd";

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

}

</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.**

Object-Oriented Programming (OOP) empowers developers to construct efficient, maintainable, and adaptable software systems. The four essential principles of OOP—encapsulation, inheritance, polymorphism, and abstraction—act as guidelines for structuring code in a way that enhances efficiency, maintainability, and reusability.

1. **Encapsulation** is comparable to a book with its content securely wrapped. It bundles data and methods together, ensuring that internal details are hidden and only necessary information is accessible. Think of a smartphone as an object. Encapsulation ensures that the inner workings (internal hardware, software implementation) are hidden, and users interact with it through well-defined methods (touchscreen, buttons, apps).
2. **Inheritance** allows a new class to inherit properties and behaviors from an existing class, fostering code reuse. Subclasses like Car and Motorcycle can inherit these properties from the base class, adding specific features unique to each type of vehicle.
3. **Polymorphism** enables objects of different classes to be treated uniformly. Circle class and a Square class, both inheriting from Shape, implement their own version of the draw method.
4. **Abstraction** is like browsing titles on a library shelf without delving into the details of each book. It emphasizes essential characteristics while hiding unnecessary complexities. Visualize a coffee maker. From a user's perspective, it abstracts the complex process of brewing coffee. Users interact with high-level functions like "start brewing" without needing to understand the intricate details of heating elements and water pumps.

In software development, OOP offers several advantages:

* Scalability: OOP facilitates the addition of new features or functionalities without disrupting existing code, ensuring adaptability to changing requirements.
* Collaborative Development: OOP's structured approach encourages collaboration among developers, as each can work on different classes independently.

A real-world scenario where OOP excels is in developing video games. Each game entity, such as characters, weapons, and obstacles, can be modeled as objects with specific attributes and behaviors. OOP simplifies game development by providing a clear and scalable framework.