Sagre, Hazel Joy 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 TriangleAreaCalculator

{

static void Main()

{

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;

class ArrayExample

{

static void Main()

{

int[] numbers = new int[5];

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

{

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

}

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("-------------------");

for (int i = 1; i <= 10; 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:**

Complete the code snippet by adding the following elements:

<!DOCTYPE html>

<html>

<head>

<title>My Website</title>

</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> <!-- Additional item in the ordered list -->

</ul>

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

<h3 style="color: red;">Your Heading</h3> <!-- Change the color to your desired color -->

</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.**

<html>

<head>

<title>Even or Odd Checker</title>

<style>

body {

text-align: center;

margin-top: 50px;

}

button {

padding: 10px;

font-size: 16px;

cursor: pointer;

}

p {

font-size: 18px;

margin-top: 20px;

}

</style>

</head>

<body>

<h1>Even or Odd Checker</h1>

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

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

<script>

function checkEvenOrOdd() {

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

var number = parseInt(userInput);

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) is important as it helps developers build software efficiently, maintain it effectively, and adapt it to evolving needs of modern programming. It consists of four key principles: encapsulation, inheritance, polymorphism, and abstraction.

1. **Encapsulation** is like putting tools in separate compartments. It keeps the inner workings of each tool hidden and only shows what's needed for others to use it. For example, a Person object might hide details like age and name, only letting other parts of the program interact with it through specific methods.
2. **Inheritance** is similar to passing down skills from one generation to the next. It allows new tools (classes) to inherit features from existing ones. For instance, a Car inherits properties like speed and fuel efficiency from a Vehicle class, saving time and effort in programming.
3. **Polymorphism** is akin to a single tool being used in different ways depending on the situation. It allows different objects to be treated similarly even if they have different behaviors. For instance, a Shape class might have different subclasses like Circle and Rectangle, each implementing their own version of a calculateArea method.
4. **Abstraction** is like looking at a tool and understanding what it does without needing to know its inner workings. It focuses on the essential features and hides unnecessary details. For example, a BankAccount class might only show methods like deposit and withdraw without revealing how it's implemented behind the scenes.

In software development, OOP brings several benefits:

* Code Reusability: OOP lets us reuse existing code, making development faster and reducing errors.
* Maintainability: It helps keep code organized and understandable, making it easier to update and fix.
* Scalability: OOP allows for easy expansion of programs, adding new features without breaking existing ones.

A real-world example where OOP shines is in building graphical user interfaces (GUIs). Each element in a GUI, like buttons and text fields, can be treated as objects with specific behaviors and properties. OOP makes it easy to create, modify, and maintain these elements, ensuring smooth interaction for users.