Malta

C# Code dag 1

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

public enum Days
{
    Monday,
    Tuesday,
    Wednesday
}

namespace RPG
{
    class Program
    {
        /// <summary>
        /// Main entry point for the application.
        /// </summary>
        static void Main()
        {
            // Declare variables
            int playerHealth = 100;
            float playerAttack = 10.0f;
            char playerClass = 'W';
            string playerName = "Hero";
            bool isPlayerAlive = true;
            string playerClassName = null;

            //Write Line Example
            Console.WriteLine("Hello World");
            Console.Write("Helo");
```

```
Console.WriteLine("World");

// Concatenation
string conctenatedString = "Hello" + " " + "World";
Console.WriteLine(conctenatedString);

// String interpolation
string interpolatedString = $"Hello {playerName}, your class is {playerClass}";
Console.WriteLine(interpolatedString);

//Console Input
Console.Write("Enter your name:");
playerName = Console.ReadLine();

//Console Input with formatting
Console.WriteLine($"Player Name: {playerName}");



//Arithmetic Operations
int a = 5, b = 10;

// Addition
int sum = a + b;
Console.WriteLine($"Sum: {sum}");

// Subtraction
int difference = b - a;
Console.WriteLine($"Difference: {difference}");

// Multiplication
int product = a * b;
Console.WriteLine($"Product: {product}");

// Division
int division = b / a;
Console.WriteLine($"Quotient: {division}");

// Modulus -- Remainder of division
int modulus = b % a;
Console.WriteLine($"Remainder: {modulus}");
```

```
            //Comparison Operators
            // Equal to
            bool isEqual = (a == b);
            Console.WriteLine($"Is Equal: {isEqual}");

            // Not equal to
            bool isNotEqual = (a != b);
            Console.WriteLine($"Is Not Equal: {isNotEqual}");

            // Greater than
            bool isGreater = (a > b);
            Console.WriteLine($"Is Greater: {isGreater}");

            // Less than
            bool isLess = (a < b);
            Console.WriteLine($"Is Less: {isLess}");

            // Greater than or equal to
            bool greaterThanOrEqual = (a >= b);
            Console.WriteLine($"Is Greater or equal too: {greaterThanOrEqual}");

            // Less than or equal to
            bool lessThanOrEqual = (a <= b);
            Console.WriteLine($"Is Less or equal too: {lessThanOrEqual}");

            //Logical Operators
            bool ACondition = true, BCondition = false;
            // AND -- Both conditions must be true
            bool andCondition = ACondition && BCondition;
            Console.WriteLine($"AND Condition: {andCondition}");

            // OR -- At least one condition must be true
            bool orCondition = ACondition || BCondition;
            Console.WriteLine($"OR Condition: {orCondition}");

            // NOT -- Inverts the value of the condition if true it becomes false and vice
versa
            bool notCondition = !ACondition;
            Console.WriteLine($"NOT Condition: {notCondition}");
```

```
            // Ternary Operator
            // A shorthand for if-else statements
            string result = (a > b) ? "A is greater than B" : "B is greater than A";
            Console.WriteLine(result);

            //Control Flow
            // If-Else Statement
            if (playerHealth > 0)
            {
                Console.WriteLine($"{playerName} is alive.");
            }
            else
            {
                Console.WriteLine($"{playerName} is dead.");
            }

            //if else if else statement
            if (playerHealth > 50)
            {
                Console.WriteLine($"{playerName} is alive.");
            }
            else if (playerHealth == 0)
            {
                Console.WriteLine($"{playerName} is dead.");
            }
            else
            {
                Console.WriteLine($"{playerName} is injured.");
            }

            //Alternative way to write if, if the if statement is 1 line long ; else with
braces
            if (playerHealth > 0) Console.WriteLine(result);

            // Switch Statement
            int day = 3;
            switch(day)
            {
                case 0: Console.WriteLine("Day is Sunday"); break;
```

```
            case 1: Console.WriteLine("Day is Monday"); break;
            case 2: //Adding break statement to prevent fall through
            case 3: Console.WriteLine("Day is Wednesday"); break;
            case 4: Console.WriteLine("Day is Thursday"); break;
            case 5: Console.WriteLine("Day is Friday"); break;
            default: Console.WriteLine("No Days"); break;
        }

        //Looping Constructs
        // For Loop

        //i++ will increment the value of i by 1 after each iteration
        //++i will increment the value of i by 1 before each iteration

        for (int i = 0; i < 5; i++)
        {
            Console.WriteLine($"For Loop Iteration: {i}");
        }

        // While Loop
        int j = 0;

        // The while loop will continue to execute as long as the condition is true
        while (j < 5)
        {
            Console.WriteLine($"While Loop Iteration: {j}"); // Print the current value of
j
            j++; // Increment j by 1
        }

        // Do-While Loop
        int k = 0;
        // The do-while loop will execute at least once before checking the condition
        do
        {
            Console.WriteLine($"Do-While Loop Iteration: {k}"); // Print the current value
of k
            k++; // Increment k by 1
        } while (k < 5); // Continue as long as k is less than 5
```

```
            // Foreach Loop
            // The foreach loop is used to iterate over collections, such as arrays or lists
            string[] names = { "Alice", "Bob", "Charlie" };
            foreach (string name in names)
            {
                Console.WriteLine($"Foreach Loop Name: {name}"); // Print each name in the
array
            }

            //Arrays and List
            // Array Declaration
            int[] numbers = new int[5]; // Declare an array of integers with a size of 5
            numbers[0] = 1; // Assign value to the first element
            numbers[1] = 2; // Assign value to the second element
            numbers[2] = 3; // Assign value to the third element
            numbers[3] = 4; // Assign value to the fourth element
            numbers[4] = 5; // Assign value to the fifth element

            // List Initialization
            List<int> numberList = new List<int>(); // Declare a list of integers
            numberList.Add(1); // Add value to the list
            numberList.Add(2); // Add value to the list
            numberList.Add(3); // Add value to the list

            //Remove
            numberList.Remove(2); // Remove value from the list

            //Remove From Index
            numberList.RemoveAt(0); // Remove value from the list at index 0

            //Length of Array -- Count of elements in the array
            Console.WriteLine($"Length of Array: {numbers.Length}");

            //Length of List -- Count of elements in the list
            Console.WriteLine($"Length of List: {numberList.Count}");

            //Dictionary
            // Dictionary Initialization
            Dictionary<string, int> playerStats = new Dictionary<string, int>();
```

```
            // Adding key-value pairs to the dictionary
            playerStats.Add("Health", 100);

            //Remove key-value pair from dictionary
            playerStats.Remove("Health");

            //Loop Through dictionary
            foreach (KeyValuePair<string, int> stat in playerStats)
            {
                Console.WriteLine($"Key: {stat.Key}, Value: {stat.Value}");
            }

            //Enum
            // Enum Declaration
            Days dayOfWeek = Days.Monday;

            //print as string
            Console.WriteLine($"Day of Week: {dayOfWeek}");
            // Enum to int
            int dayValue = (int)dayOfWeek;

            Console.WriteLine($"Day Value: {dayValue}"); //0

            //Try Catch
            try
            {
                // Code that may throw an exception
                // This will throw a DivideByZeroException
                int resultTry = 10 / 1; // Division by zero
                Console.WriteLine($"Result: {resultTry}"); // This line will not execute if an
exception is thrown
            }
            catch (DivideByZeroException ex)
            {
                // Handle the exception
                Console.WriteLine($"Error: {ex.Message}");
            }
            catch (Exception ex)
            {
                // Handle any other exceptions
```

```
                Console.WriteLine($"An error occurred: {ex.Message}");
            }
            finally
            {
                // Code that will always execute, regardless of whether an exception occurred
                Console.WriteLine("Finally block executed.");
            }

        }
    }
}
```

Revision #1
Created 2025-05-06 17:01:12 UTC by Max
Updated 2025-05-06 17:02:26 UTC by Max