**Lab 1**

|  |  |  |
| --- | --- | --- |
| **Category** | **C#** | **Java** |
| **Classes** | PascalCase (e.g., CustomerOrder) | PascalCase (e.g., CustomerOrder) |
| **Interfaces** | PascalCase (e.g., IShape) | PascalCase, often prefixed with "I" (e.g., Shape) |
| **Methods** | PascalCase (e.g., CalculateTotal()) | camelCase (e.g., calculateTotal()) |
| **Variables** | camelCase (e.g., customerName) | camelCase (e.g., customerName) |
| **Constants** | PascalCase or ALL\_UPPERCASE\_WITH\_UNDERSCORES (e.g., MaxValue or MAX\_VALUE) | ALL\_UPPERCASE\_WITH\_UNDERSCORES (e.g., MAX\_VALUE) |
| **Namespaces** | PascalCase (e.g., MyCompany.MyProduct) | all lowercase (e.g., com.mycompany.myproduct) |
| **Braces** | Opening brace on the same line | Opening brace on the same line |
| **Indentation** | Typically 4 spaces | Typically 4 spaces |
| **Single-line Comments** | // | // |
| **Multi-line Comments** | /\* ... \*/ | /\* ... \*/ |
| **XML Documentation** | /// | /\*\* ... \*/ (for Javadoc) |
| **Properties** | Uses PascalCase for property names (e.g., public string Name { get; set; }) | Uses camelCase for getter/setter methods (e.g., public String getName()) |
| **Events** | Uses PascalCase for event names (e.g., public event EventHandler MyEvent;) | Uses PascalCase for event names (e.g., public EventListener myEvent;) |
| **Enums** | Enum names use PascalCase, values use PascalCase (e.g., public enum Color { Red, Green, Blue }) | Enum names use PascalCase, values use ALL\_UPPERCASE\_WITH\_UNDERSCORES (e.g., public enum Color { RED, GREEN, BLUE }) |
| **Exception Handling** | Uses try-catch-finally blocks, with catch typically using the exception type (e.g., catch (Exception ex)) | Uses try-catch-finally blocks, with catch typically using the exception type (e.g., catch (Exception e)) |
| **Namespaces/Packages** | Namespaces can be nested and are often organized by functionality (e.g., MyCompany.MyProduct.Services) | Packages are hierarchical and often reflect the directory structure (e.g., com.mycompany.myproduct.services) |
| **Static Classes** | Can define static classes (e.g., public static class MathUtils { ... }) | Can define utility classes with private constructors to prevent instantiation (e.g., public final class MathUtils { private MathUtils() { } }) |
| **Lambda Expressions** | Uses => syntax (e.g., x => x \* x) | Uses -> syntax (e.g., x -> x \* x) |
| **Attributes/Annotations** | Uses attributes (e.g., [Obsolete]) | Uses annotations (e.g., @Deprecated) |
| **Default Values** | Can specify default values in method parameters (e.g., void MyMethod(int x = 10)) | Does not support default parameter values; method overloading is used instead. |
| **Access Modifiers** | public, private, protected, internal, protected internal | public, private, protected, package-private (default) |