# Hands-on lab

## Lab: Language execution basics

September 2020

## Exercise 1

Create a class *Person*, with an *int* property *Age* and a *string* property Name. Now create a list of 3 *Person*-objects and prove (by means of a Unit Test) they are different objects.

Use .NET Core.

## Exercise 2

Now create a *PersonValue* **struct** instead of a *Person* class and has the same properties. Prove by means of a unit test a different behavior between value and reference types.

## Exercise 3

Consider the following class *Grade*:

**C#**

```
class Grade
{
	public float Score { get; set; } // Between 0.0f and 100.0f
	public string CourseName { get; set; }
}
```

And the following program:

**C#**

```
class Program
{
	static void Main(string[] args)
	{
		GradeBook book = new GradeBook();

		var grades = new List<Grade>
		{
			new Grade {CourseName = "Java", Score = 80.0f },
			new Grade {CourseName = "PHP", Score = 20.0f }
		};

		grades[0].Score = 85.0f;
	}
}
```

Now change the class *Grade* from **class** to **struct** and recompile. What do you notice? Explain! Fix it, while keeping the **struct**.

## Exercise 4

Study Benchmark.NET. Compare the creation of Person objects vs PersonValue objects. Prove that it is faster to allocate value objects than it is to allocate reference objects

## Exercise 5

What is the fastest way to concatenate strings? Evaluate at least the following approaches:

- + operator
- StringBuilder
- Interpolated strings ($)

Compare your code with the pitfalls in the following article. Did you make any mistakes in this regard?

https://www.meziantou.net/stringbuilder-performance-pitfalls.htm

## Exercise 6

Study and reflect upon:

https://dev.to/tyrrrz/interview-question-heap-vs-stack-c-5aae