**– Lab 1.2 –**

**3.d.i.2.a.**

.method private hidebysig static int32 Add(int32 a,

int32 b) cil managed

{

// Code size 9 (0x9)

.maxstack 2 //JIT should reserve 2 stack cells for this method.

.locals init (int32 V\_0) //Local variable declaration.

IL\_0000: nop //Do nothing (Can be used for debugging

//or code optimization).

IL\_0001: ldarg.0 //Load the 1st argument on the evaluation stack.

IL\_0002: ldarg.1 //Load the 2nd argument on the evaluation stack.

IL\_0003: add //Add the top 2 values on the stack and push their sum

//back on the top of the stack.

IL\_0004: stloc.0 //Pop a value from the stack into the local variable.

IL\_0005: br.s IL\_0007 //Branch to target (what happened to IL\_0006?)

IL\_0007: ldloc.0 //Load the local variable on the evaluation stack.

IL\_0008: ret //Return from method.

} // end of method Calc::Add

//NOTE: Since the instruction “add” loads the method’s return value (the sum) on the stack’s top,

// the local variable is probably used (in IL\_0004 - IL\_0007) for debugging purposes.

**3.e.ii.**

The file “calc.il” contains the manifest (metadata) and the IL code from the file “add.netmodule”.  
The file “calc.res” is a Win32 resource file, which was generated from the file “add.netmodule”.

**3.g.iii.1.**

The class “Calc” is not recognized by the class “Program”, because it was compiled separately, as a module. Therefore, this error can be fixed by explicitly adding the module file “calc.netmodule” during the compilation of the source file “program.cs”.

**3.g.iv.1.**

Since “Add” and “Subtract” are **private** methods of the class “Calc”, they cannot be accessed outside of it. To allow the class “Program” access to these methods, they have to be set as **public**.

**3.g.x.3.**

We see the metadata of the executable file “calc.exe”.