Arithmetic operation of Anonymous methods

Anonymous methods provide a technique to pass a code block as a delegate parameter. Anonymous methods are the methods without a name, just the body.

Delegate declaration determines the methods that can be referenced by the delegate. A delegate can refer to a method, which has the same signature as that of the delegate.

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

delegate void AnonymousArithmetic(int n);
namespace Anonymous
{
    class TestDelegate
    {
        static int num = 20;

        public static void additional(int a)
        {
            num += a;
            Console.WriteLine("Named addition: {0}", num);
        }
        public static void subtraction(int s)
        {
            num -= s;
            Console.WriteLine("Named subtraction: {0}", num);
        }
        public static void multiplication(int m)
        {
            num *= m;
            Console.WriteLine("Named multiplication: {0}", num);
        }
        public static void division(int d)
        {
            num /= d;
            Console.WriteLine("Named division: {0}", num);
        }

        public static int getNum()
        {
            return num;
        }

        static void Main(string[] args)
        {
            //create delegate instances using anonymous method
            AnonymousArithmetic Anonymousvalue = delegate (int x) {
                Console.WriteLine("Anonymous Method: {0}", x);
            };

            //calling the delegate using the anonymous method
```

```
            Anonymousvalue(20);

            //instantiating the delegate using the additional methods
            Anonymousvalue = new AnonymousArithmetic(additional);
            Anonymousvalue(10);

            //instantiating the delegate using the subtraction methods
            Anonymousvalue = new AnonymousArithmetic(subtraction);
            Anonymousvalue(10);

            //instantiating the delegate using the multiplication methods
            Anonymousvalue = new AnonymousArithmetic(multiplication);
            Anonymousvalue(10);

            //instantiating the delegate using another division methods
            Anonymousvalue = new AnonymousArithmetic(division);
            Anonymousvalue(2);

            Console.ReadKey();
        }
    }
}
```