Aula Prática 10

1. Analise o seguinte código. Apresente na forma que considere conveniente o estado do *heap* (relativo aos objectos criados durante a execução do método Main e organizado por gerações) nos pontos identificados pelos comentários.
   Experimente e utilize os métodos da classe GC para verificar as gerações dos objectos.

```
class MyReference : IDisposable {
  object strongRef;
  WeakReference weakRef;

 public MyReference(object o, bool isStrong)
  {
   if (isStrong) strongRef = o;
   else weakRef = new WeakReference(o);
  }

  private void Dispose(bool disposing) {
   if (strongRef != null) strongRef = null;
   else weakRef.Target = null;
  }

  public void Dispose() {
   Dispose(true);
   GC.SuppressFinalize(this);
  }

  public object Target {
    get {
    if (strongRef != null) return strongRef;
    return weakRef.Target;
    }
  }
  ~MyReference() { Dispose(false); }
```

```
class A {
  public readonly int v;
  public A(int v) { this.v=v; }
  public override string ToString() {
    return v.ToString();
  }
}

class Program {
  static void Main() {
    A a1 = new A(3), a2 = new A(4);
    MyReference m1;
    m1 = new MyReference(a1, false);
    Console.WriteLine(m1.Target);
    //1
    GC.Collect();
    //2
   using (MyReference m2 = new
                  MyReference(a2,true)){
     Console.WriteLine(m2.Target);
   }
   //3
    GC.WaitForPendingFinalizers();
    GC.Collect();
  //4
 }
}
```

2. Defina um namespace Example e defina nesse namespace a seguinte classe Aluno:

```
    class Aluno{
        public int number;
        public String name;
        public Aluno(){ name="AVE";}
        public Aluno(int nr, String name) { number = nr; this.name = name; }
        public String GetName(){ return name; }
        public int GetNumber(){ return number; }
        }
```

3. Defina no mesmo **`namespace`** a seguinte classe **`Program`**

```
Using System.Reflection;
//…
public class Program{
   static void Main(string[] args){
        try{
        Type myTypeObj = Type.GetType("Example.Aluno");
        object a2 = Activator.CreateInstance(myTypeObj);
        Console.WriteLine(a2.GetType());
             }
}
```

   a. O que é que observou?
   b. Acrescente ao método **`Main`** a instrução **`Type t=a2.GetType();`** Utilize o método **`GetMethod`** da classe **`Type`** para obter o **`MethodInfo`** de **`GetName`**, isto é, os

atributos do método e acesso à sua metadata. Teste a sua aplicação, utilizando a *property* **IsPublic** para confirmar que o método é público

c. Seja **m** o **MethodInfo** da alínea anterior, acrescente ao seu código as instruções:

```
String s=(String)m.Invoke(a2, null);
Console.WriteLine(s);
```

Qual foi o constructor utilizado da classe aluno?

d. Utilize outro método da classe **Activator**, para que possa ser utilizado o outro construtor da classe Aluno.

3 Acrescente ao namespace o método **ListMethods(Type t)** que lista todos os métodos do tipo **t**.

Faça uma aplicação que teste o seu método com o tipo **Aluno**.