Série de Exercícios 4

1. Na 1ª linha os bytes alocados são relativos ao objecto base que está a ser executado.

Na 2ª linha é o espaço ocupado pelos vários objectos A alocados no array.

Na 3ª linha a seguir a ser executado o GC.Collect() o valor é muito próximo porque o array “v” é destruído mas os objectos “A” contidos no array não são destruídos visto a classe A ter um destructor vazio.

Na 4ª linha o método GC.WaitForPendingFinalizers() suspende as threads em execução e marca os vários objectos “A” para remoção pelo GC. Quando o método GC.Collect() é executado, vai finalizar todos os objectos marcados para remoção anteriormente.

2.a) No *Test1-A* é mais rápido a criar objectos visto a classe *Test1* não redefinir o *destructor* usando assim o *destructor* herdado, ao contrário da classe *Test2* a qual redefine o *destructor* sendo necessário apontar para o novo método na tabela de métodos.

No Test1-B leva algum tempo visto o *GC* estar a fazer a recolha dos objectos que estavam alocados no *array objects*, e ao passar ao *Test1-C* já não tem nada a recolher, logo o tempo perdido entre o *Test1-B* e *Test1-C* é 0.

No *Test2-B* na primeira chamada ao *GC.Collect* são marcados para remoção os objectos pertencentes ao *array objects* que são depois finalizados na chamada ao *GC.Collect* do *Test2-C*.

b) Ao alterar o código de criação dos objectos, passa-se a usar reflexão o que consome muito mais recursos fazendo com que o algoritmo torne-se muito mais lento

3.A excepção na execução do código é devido aos blocos *using* que chama o método Dispose*()* no fim do bloco.

A correção do código passa por remover a utilização dos blocos *using* apresentada abaixo:

class Program

{

public static void Main()

{

var resp = GetResponseBody("http://users.isr.ist.utl.pt/~cfb/VdS/v317.txt");

string line = null;

while ((line = resp.Invoke()) != null)

Console.WriteLine(line);

}

public static Func<string> GetResponseBody(string path)

{

WebRequest request = WebRequest.Create(path);

WebResponse response = request.GetResponse();

StreamReader reader = new StreamReader(response.GetResponseStream());

return () => reader.ReadLine();

}

}