📌 Difference between Heap and Stack

Both Heap and Stack are memory areas in **RAM** (not on hard disk).

Stack:

- Stores **local variables**, method calls, and function parameters.
- Memory is allocated and deallocated automatically (LIFO – Last In, First Out).
- Very fast access.
- Size is usually much smaller than the heap and limited by the OS.
- Example:

```
void Foo() {
   int x = 10;  // stored in stack
}
```

Heap:

- Stores **objects, arrays, and dynamic data**.
- Managed manually (in C++) or by **Garbage Collector** (in C#, Java).
- Slower than stack, but much larger in size.
- Memory remains until garbage-collected or explicitly freed.
- Example:
- var obj = new MyClass(); // object stored in heap

✅ Both reside in **RAM**, not in hard disk. The hard disk is only used for paging/swapping if RAM is full.

📌 ArrayList.TrimToSize()

This method sets the **Capacity** of the ArrayList to be equal to its **Count** (the number of elements actually in the list).

Example:

```
ArrayList list = new ArrayList();
list.Add(1);
list.Add(2);
list.Add(3);

Console.WriteLine("Count: " + list.Count);     // 3
Console.WriteLine("Capacity: " + list.Capacity); // 4 (default grow size)

list.TrimToSize(); // reduces capacity to match count
Console.WriteLine("Capacity after trim: " + list.Capacity); // 3
```

🔑 **Key Points:**

- It **frees unused memory** by cutting capacity down to the exact number of elements.
- After trimming, if you add a new element, the ArrayList will **increase its capacity again (usually doubling)**.
- Useful when you know the collection won't grow further → optimizes memory usage.

✅ So in short:
TrimToSize() = shrink extra reserved space, keep only what's needed.