# INTCDE21ID008

# STAGE-3

## 916483 – Naveen S

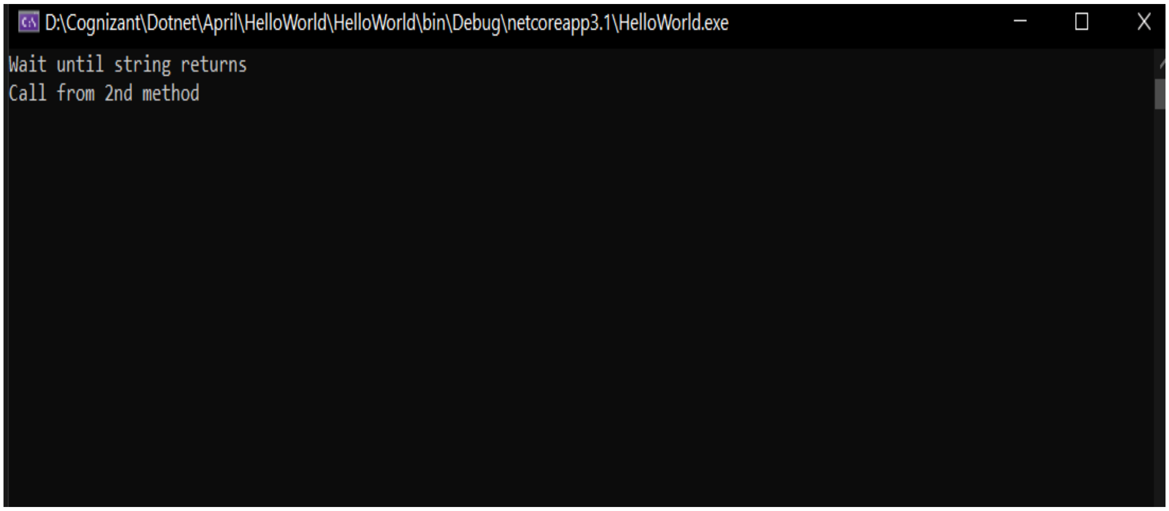
## Day 3 – C# Additional Topics Async Programming, Multithreading

### Hands-On 1:

Async Await usage – 1

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

namespace AsyncHandsOn
{
    class Program
    {
        public static async Task Method1()
        {
            await Task.Run(() =>
            {
                string msg = Method2();
                Console.WriteLine(msg);
            });
        }

        public static string Method2()
        {
            Console.WriteLine("Wait until string returns");
            Thread.Sleep(3340);
            return "Call from 2nd method";
        }
        static void Main(string[] args)
        {
            Method1();
            Console.ReadKey();
        }
    }
}
```

### OUTPUT:


D:\Cognizant\Dotnet\April\HelloWorld\HelloWorld\bin\Debug\netcoreapp3.1\HelloWorld.exe

```
Wait until string returns
Call from 2nd method
```

## Async Await usage – 2

### Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApp
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        public int CountChars()
        {
            int count = 0;
            using (StreamReader streamReader = new StreamReader("D:\\Day3_Handson\\Kuttralam.txt"))
            {
                string content = streamReader.ReadToEnd();
                count = content.Length;
                Thread.Sleep(2000);
            }
            return count;
        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }

        private async void button1_Click_1(object sender, EventArgs e)
        {
            Task<int> task = new Task<int>(CountChars);
            task.Start();

            label1.Text = "File is processing";
            int count = await task;
            label1.Text = count.ToString() + " characters";
        }
    }
}
```

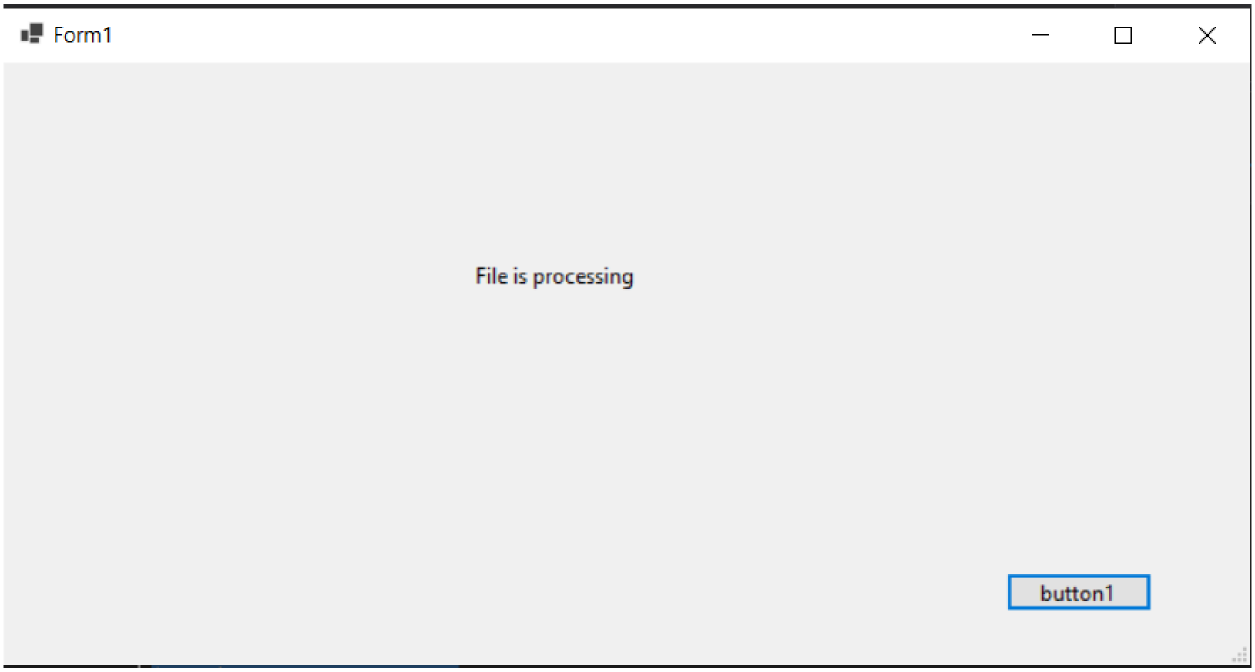
## Form1.Designer.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApp
{
    partial class Form1
    {
        /// <summary>
        ///  Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        ///  Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }
        #region Windows Form Designer generated code

        /// <summary>
        ///  Required method for Designer support - do not modify
        ///  the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.label1 = new System.Windows.Forms.Label();
            this.button1 = new System.Windows.Forms.Button();
            this.SuspendLayout();
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Location = new System.Drawing.Point(300, 150);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(0, 20);
            this.label1.TabIndex = 1;
            //
            // button1
            //
            this.button1.Location = new System.Drawing.Point(643, 381);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(94, 29);
            this.button1.TabIndex = 2;
            this.button1.Text = "button1";
            this.button1.UseVisualStyleBackColor = true;
            this.button1.Click += new System.EventHandler(this.button1_Click_1);
            //
            // Form1
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(8F, 20F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(800, 450);
            this.Controls.Add(this.button1);
            this.Controls.Add(this.label1);
            this.Name = "Form1";
            this.SizeGripStyle = System.Windows.Forms.SizeGripStyle.Show;
            this.Text = "Form1";
            this.Load += new System.EventHandler(this.Form1_Load);
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Button button1;
    }
}
```

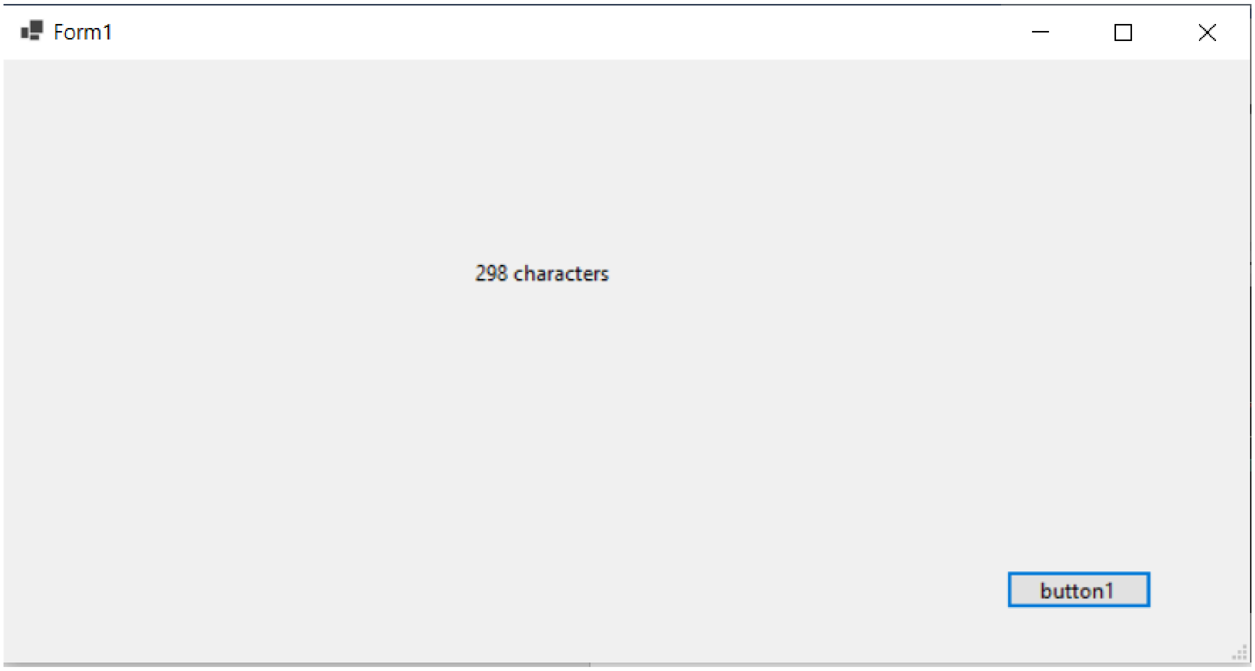
## Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApp
{
    static class Program
    {

        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

## OUTPUT:



File is processing



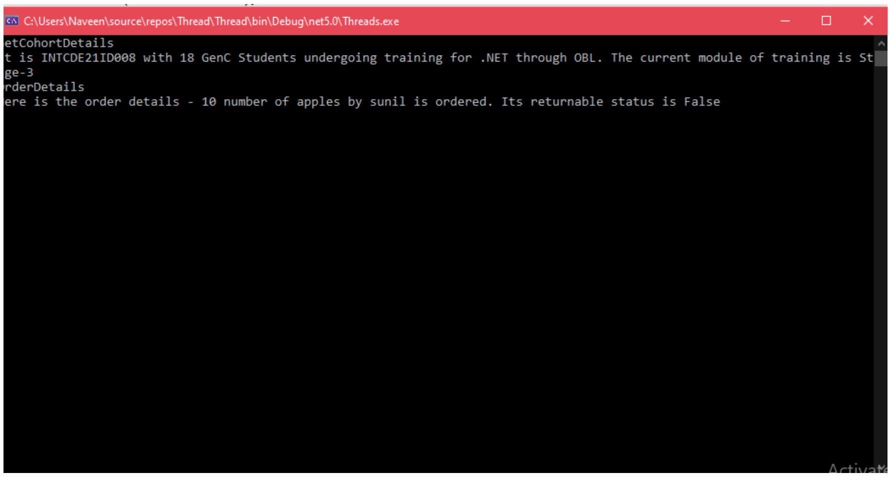
298 characters

## NAMED PARAMETERS – ORDER OF ARGUMENTS AS PER THE FUNCTION AND MODIFY

### Program.cs

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

namespace named_arguments
{
    class Program
    {
        public static void GetCohortDetails(string Cohortname, int GenCcount, string Mode, string Track, string
CurrentModule)
        {
            Console.WriteLine("It is {0} with {1} GenC Students undergoing training for {2} through {3}. The current
module of training is {4}", Cohortname, GenCcount, Track, Mode, CurrentModule);
        }
        public static void OrderDetails(string Productname, string Sellername, int Orderquantity = 1, bool returnable = true)
        {
            Console.WriteLine("Here is the order details – {0} number of {1} by {2} is ordered. Its returnable status is {3}",
Orderquantity, Productname, Sellername, returnable);
        }

        static void Main(string[] args)
        {
            Console.WriteLine("GetCohortDetails");
            GetCohortDetails(Cohortname: "INTCDE21008", GenCcount: 18, Track: ".NET", Mode: "OBL", CurrentModule:
"Stage-3");
            Console.WriteLine("OrderDetails");
            OrderDetails(Sellername: "sunil", Productname: "apples", Orderquantity: 10, returnable: false);


            Console.Read();

        }
    }
}
```

# OUTPUT:


C:\Users\Naveen\source\repos\Thread\Thread\bin\Debug\net5.0\Threads.exe

```
etCohortDetails
t is INTCDE21ID008 with 18 GenC Students undergoing training for .NET through OBL. The current module of training is St
ge-3
rderDetails
ere is the order details - 10 number of apples by sunil is ordered. Its returnable status is False
```