**Sadaf Jalil**

**Roll No: 80**

**BSCS**

**Semester: 7th (B)**

**Lab**

**Task# 01**

Write a C# program that prints out the name of a fruit at timed intervals.

* For this create and start 3 threads.
* The run () method for the fruit class has a loop that repeats 3 times.
* The sleep () method causes the threads to sleep for the number of a seconds specified when the thread is created.
* Once the thread is done sleeping, it should output a time value, the name of the fruit specified, and the current loop number.
* After 3 iteration the thread terminated.

using System;

using System.Collections.Generic;

using System.Threading;

using System.Web;

class Fruit

{

public string Name;

public int sleep;

public Fruit(string n,int s)

{

Name = n;

sleep = s;

}

public void Run()

{

for (int i = 0; i < 5; i++)

{

Thread.Sleep(sleep \* 1000);

Console.WriteLine ($"slept for {sleep} seconds: Fruit :{ Name}, Iteration :{ i}");

}

}

Static void Main (string [] ages)

{

Fruit apple = new Fruit ("Apple", 1);

Fruit Mango = new Fruit ("Mango", 2);

Fruit Grapes = new Fruit ("Grapes", 3);

Thread t1 = new Thread (apple. Run);

Thread t2 = new Thread (Mango. Run);

Thread t3 = new Thread (Grapes. Run);

t1.Start ();

t2.Start ();

t3.Start ();

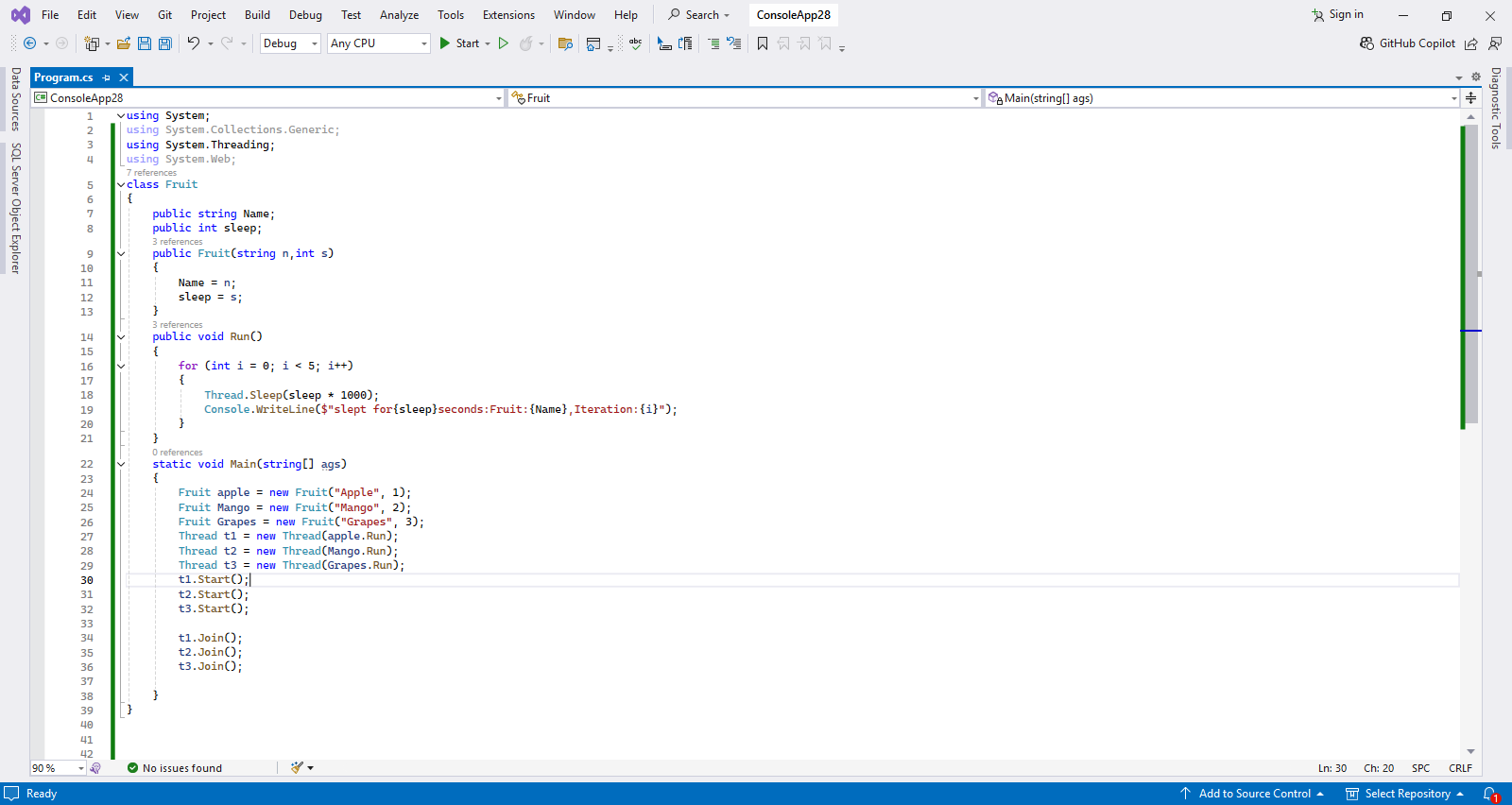
t1.Join ();

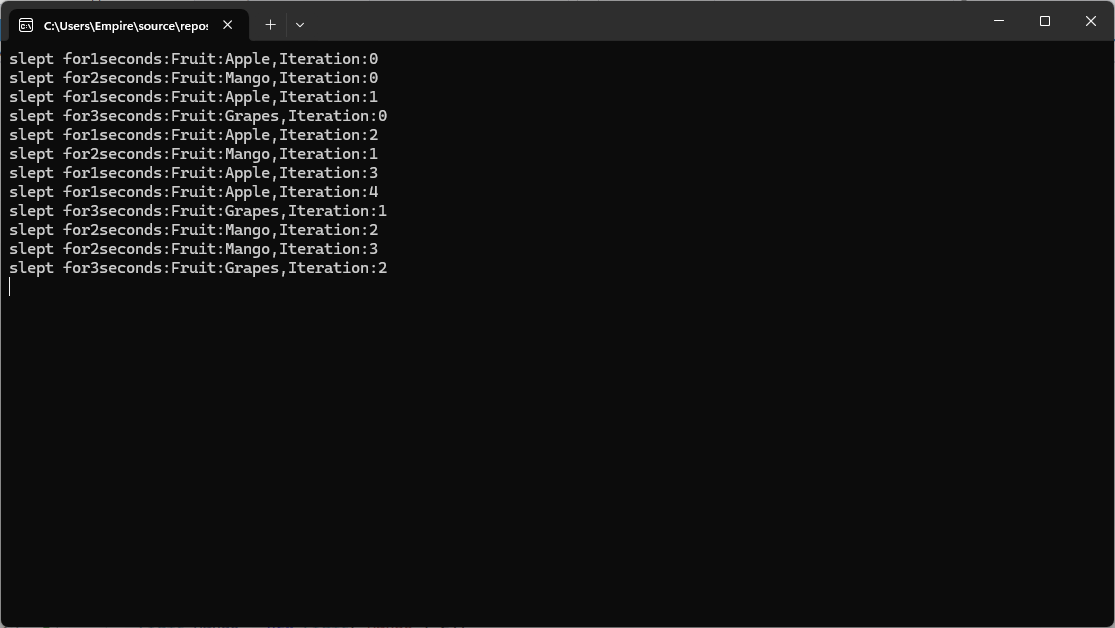
t2.Join ();

t3.Join ();

}

}





**Task# 02**

Write C# program that implements a multithread application that has three threads.

* First thread generates random integer for every second and
* If the value is even, second thread computes and prints square of number.
* If the value is odd, the third thread will print the value of cube of number.

using System;

Using System.Security.Cryptography;

using System.Threading;

class Program

{

Static into shared Number = 0;

static AutoResetEvent numberGeneratedEvent = newAutoResetEvent(false);

static AutoResetEvent processedEvent = new AutoResetEvent(true);

static bool stop = false;

static void Main()

{

Thread generatorThread = new Thread(GenerateRandomNumbers);

Thread evenThread = new Thread(ProcessEvenNumbers);

Thread oddThread = new Thread(ProcessOddNumbers);

generatorThread.Start();

evenThread.Start();

oddThread.Start();

Thread.Sleep(30000);

stop = true;

numberGeneratedEvent.Set();

processedEvent.Set();

generatorThread.Join();

evenThread.Join();

oddThread.Join();

Console.WriteLine(&quot; Program finished.&quot;);

}

static void GenerateRandomNumbers()

{

Random rand = new Random();

int i = 1;

while (i & lt;= 10)

{

processedEvent.WaitOne();

sharedNumber = rand.Next(1, 101);

Console.WriteLine($&quot; Generated number: {sharedNumber}

&quot;);

numberGeneratedEvent.Set();

Thread.Sleep(1000);

i++;

}

}

static void ProcessEvenNumbers()

{

while (!stop)

{

numberGeneratedEvent.WaitOne();

if (stop) break;

if (sharedNumber % 2 == 0)

{

int square = sharedNumber \* sharedNumber;

Console.WriteLine($&quot; Even Thread: Number = { sharedNumber}, Square = { square}

&quot;);

processedEvent.Set();

}

else

{

numberGeneratedEvent.Set();

}

}

}

static void ProcessOddNumbers()

{

while (!stop)

{

numberGeneratedEvent.WaitOne();

if (stop) break;

if (sharedNumber % 2 != 0)

{

int cube = sharedNumber \* sharedNumber \* sharedNumber;

Console.WriteLine($&quot; Odd Thread: Number = {sharedNumber}, Cube = { cube}

&quot;);

processedEvent.Set();

}

else

{

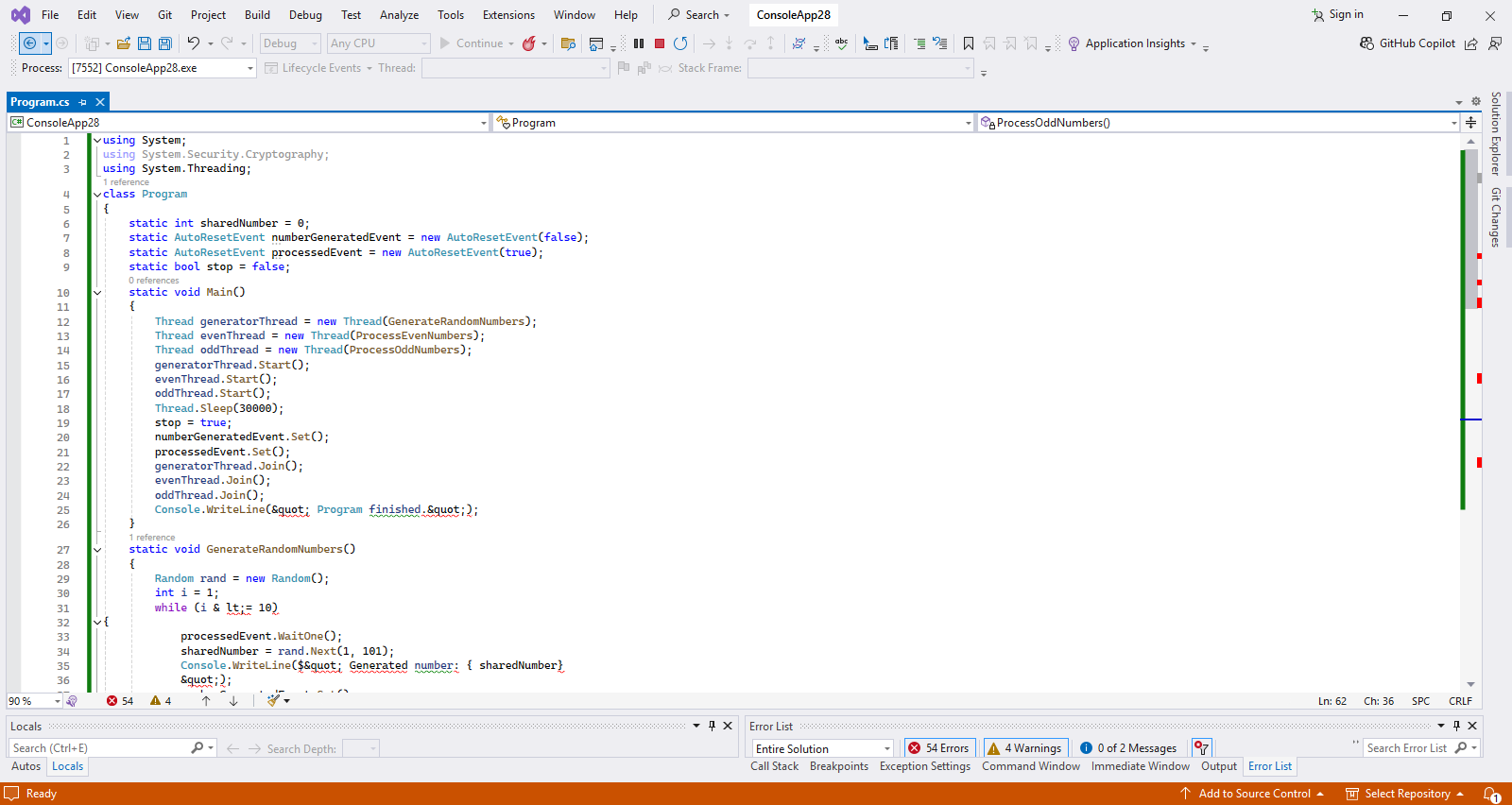
numberGeneratedEvent.Set();

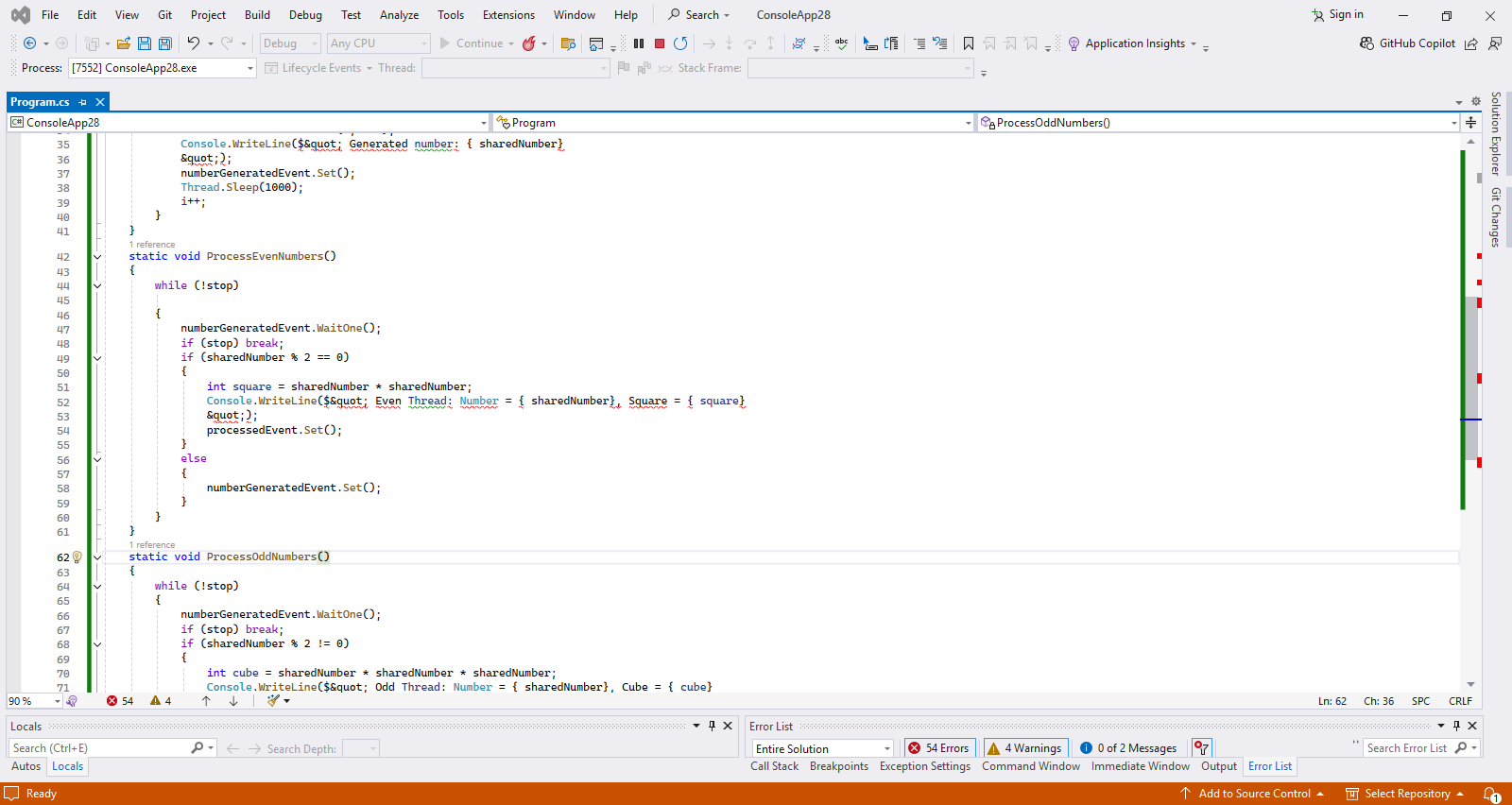
}

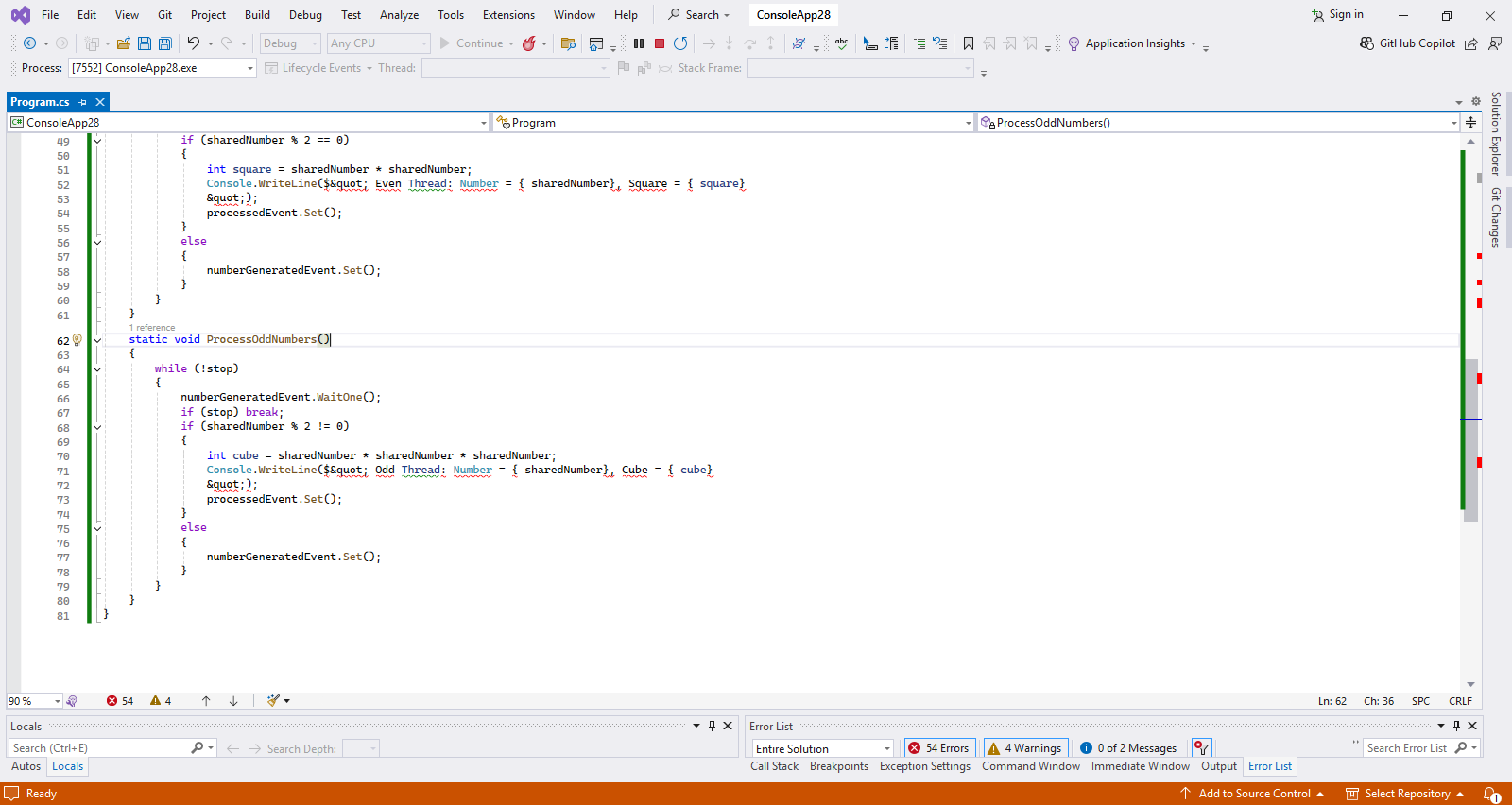
}

}

}







Output:

s