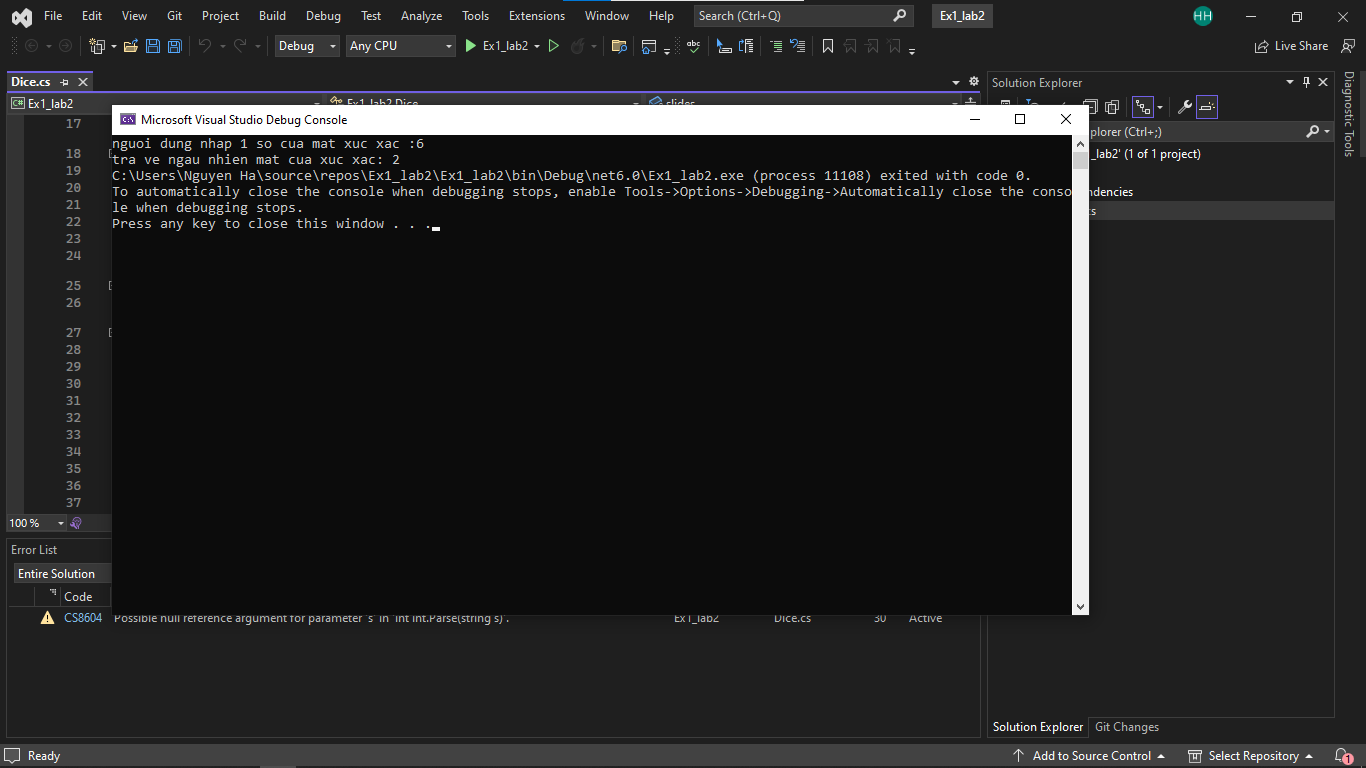
Họ và tên: Hồ Nguyên Hà . Mã SV: 20it043

Bài Tập Tuần 2

**Ex1:**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Ex1\_lab2

{

class Dice

{

private int slides;

public Dice(int slides) {

this.slides = slides;

}

public int roll()

{

Random random = new Random();

return random.Next(1, slides); ;

}

}

class Program

{

static void Main(string[] args)

{

Console.Write("nguoi dung nhap 1 so cua mat xuc xac :");

int a = int.Parse(Console.ReadLine());

Dice dice = new Dice(a);

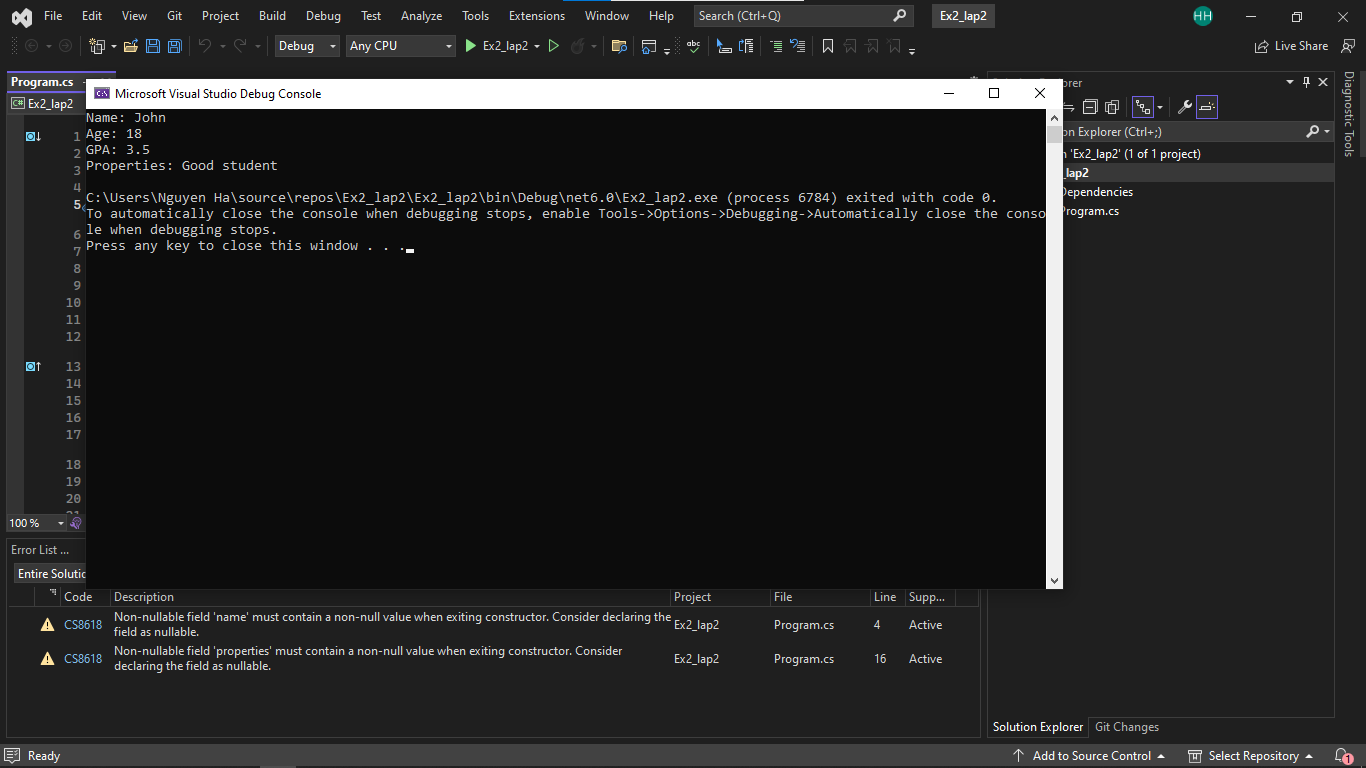
Console.Write("tra ve ngau nhien mat cua xuc xac: " + dice.roll());

}

}

}

**Ex2:**



public class Person

{

public int age;

public string name;

public void DisplayPerson()

{

Console.WriteLine("Name: {0}", name);

Console.WriteLine("Age: {0}", age);

}

}

public class Student : Person

{

public float gpa;

public string properties;

public void DisplayStudent()

{

Console.WriteLine("Name: {0}", name);

Console.WriteLine("Age: {0}", age);

Console.WriteLine("GPA: {0}", gpa);

Console.WriteLine("Properties: {0}", properties);

}

}

public class Program

{

static void Main(string[] args)

{

Student student = new Student();

student.name = "John";

student.age = 18;

student.gpa = 3.5f;

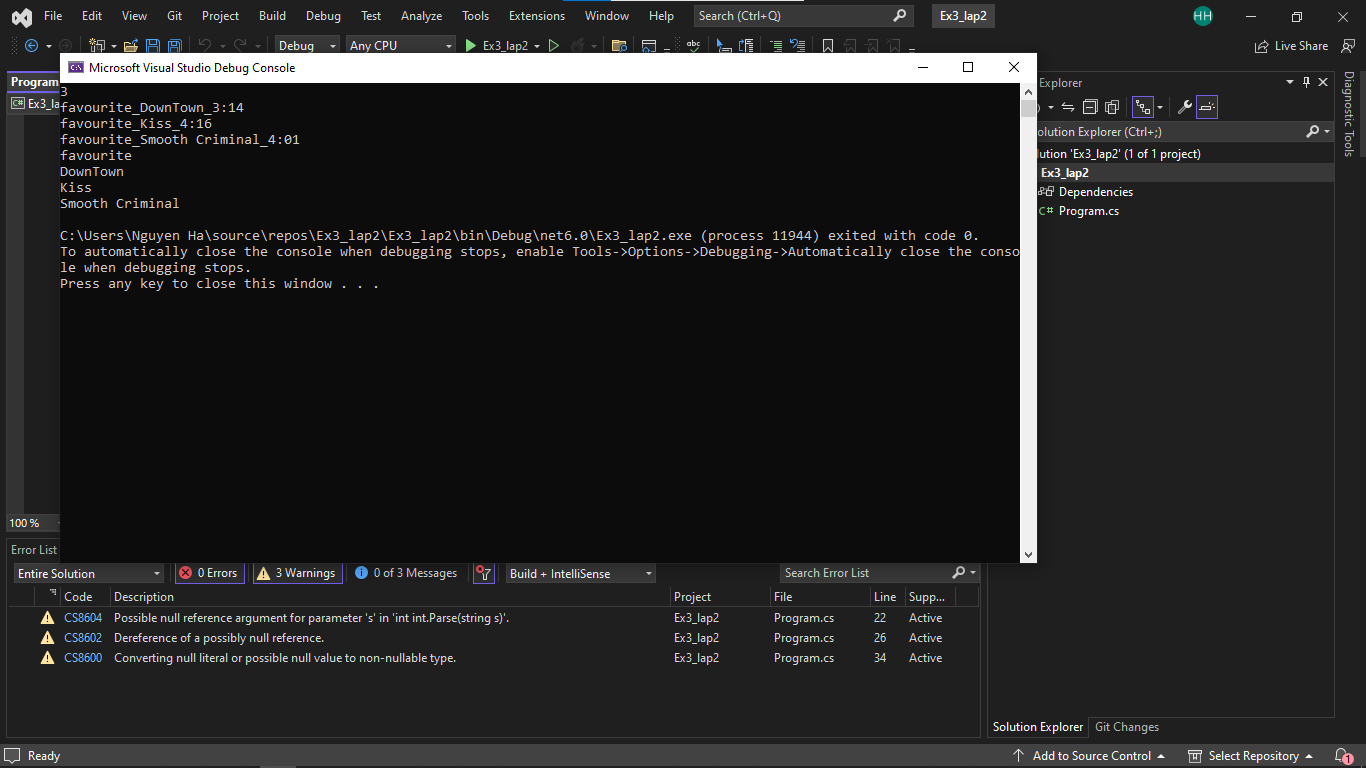
student.properties = "Good student";

student.DisplayStudent();

}

}

**Ex3:**



using System;

using System.Collections.Generic;

class Song

{

public string Typelist { get; set; }

public string Name { get; set; }

public string Time { get; set; }

public Song(string typelist, string name, string time)

{

Typelist = typelist;

Name = name;

Time = time;

}

}

class Program

{

static void Main(string[] args)

{

int n = int.Parse(Console.ReadLine());

List<Song> songs = new List<Song>();

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

{

string[] data = Console.ReadLine().Split('\_');

string typelist = data[0];

string name = data[1];

string time = data[2];

Song song = new Song(typelist, name, time);

songs.Add(song);

}

string typelistFilter = Console.ReadLine();

foreach (Song song in songs)

{

if (song.Typelist == typelistFilter || typelistFilter == "all")

{

Console.WriteLine(song.Name);

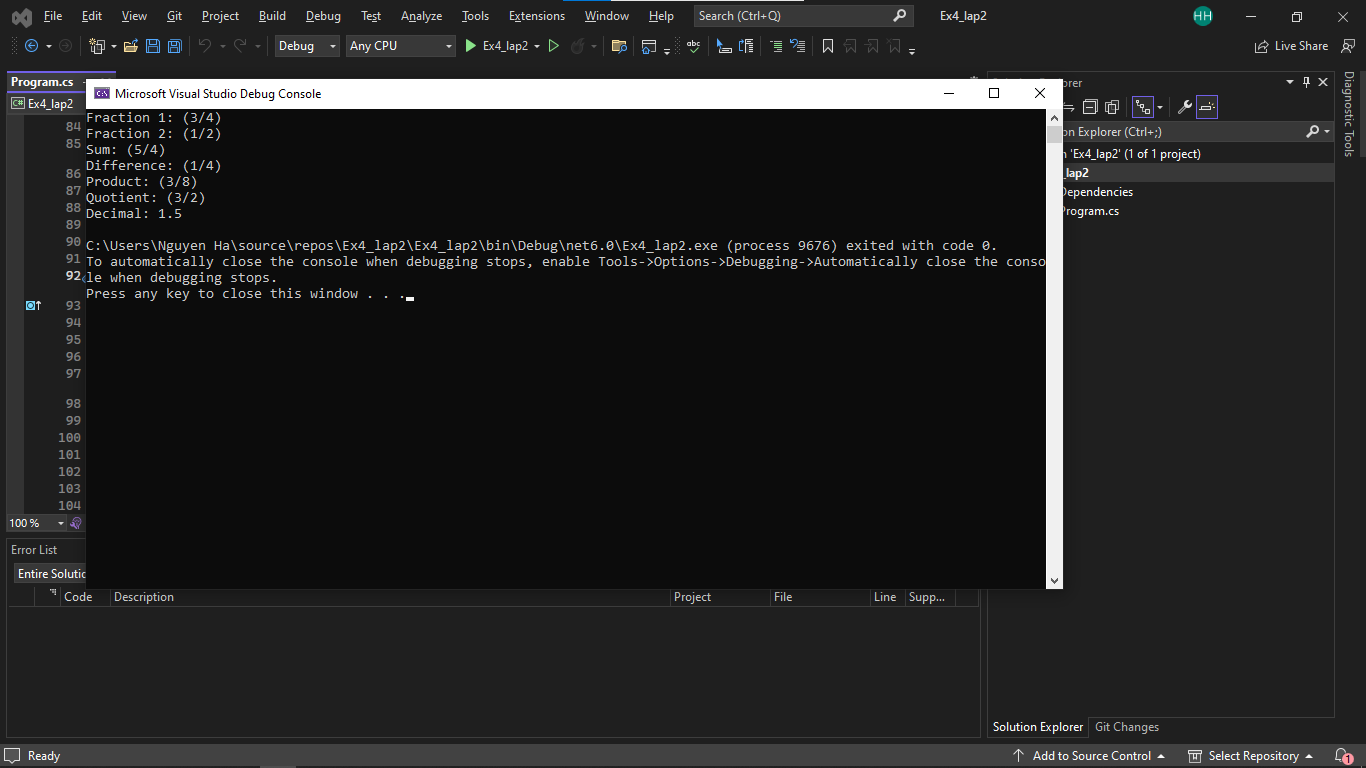
}

}

}

}

**Ex4:**



using System;

namespace Fraction

{

class Program

{

static void Main(string[] args)

{

// Create two fractions

Fraction fraction1 = new Fraction(3, 4);

Fraction fraction2 = new Fraction(1, 2);

// Print the fractions

Console.WriteLine($"Fraction 1: {fraction1}");

Console.WriteLine($"Fraction 2: {fraction2}");

// Add the fractions

Fraction sumFraction = fraction1 + fraction2;

Console.WriteLine($"Sum: {sumFraction}");

// Subtract the fractions

Fraction differenceFraction = fraction1 - fraction2;

Console.WriteLine($"Difference: {differenceFraction}");

// Multiply the fractions

Fraction productFraction = fraction1 \* fraction2;

Console.WriteLine($"Product: {productFraction}");

// Divide the fractions

Fraction quotientFraction = fraction1 / fraction2;

Console.WriteLine($"Quotient: {quotientFraction}");

// Convert the quotient to a decimal

double decimalFraction = quotientFraction.ToDecimal();

Console.WriteLine($"Decimal: {decimalFraction}");

}

}

class Fraction

{

private int numerator;

private int denominator;

public Fraction(int numerator, int denominator)

{

this.numerator = numerator;

this.denominator = denominator;

Normalize();

}

private void Normalize()

{

int commonDivisor = Gcd(numerator, denominator);

numerator /= commonDivisor;

denominator /= commonDivisor;

}

private int Gcd(int a, int b)

{

if (b == 0)

return a;

return Gcd(b, a % b);

}

public static Fraction operator +(Fraction f1, Fraction f2)

{

int numerator = f1.numerator \* f2.denominator + f2.numerator \* f1.denominator;

int denominator = f1.denominator \* f2.denominator;

return new Fraction(numerator, denominator);

}

public static Fraction operator -(Fraction f1, Fraction f2)

{

int numerator = f1.numerator \* f2.denominator - f2.numerator \* f1.denominator;

int denominator = f1.denominator \* f2.denominator;

return new Fraction(numerator, denominator);

}

public static Fraction operator \*(Fraction f1, Fraction f2)

{

int numerator = f1.numerator \* f2.numerator;

int denominator = f1.denominator \* f2.denominator;

return new Fraction(numerator, denominator);

}

public static Fraction operator /(Fraction f1, Fraction f2)

{

int numerator = f1.numerator \* f2.denominator;

int denominator = f1.denominator \* f2.numerator;

return new Fraction(numerator, denominator);

}

public override string ToString()

{

return $"({numerator}/{denominator})";

}

public double ToDecimal()

{

return (double)numerator / (double)denominator;

}

}

}