STATE-OF-THE-ART PROGRAMMING IN VISUAL BASIC AND C#: A COMPARATIVE STUDY

Muhammad Khurram Zahoor[1,*], Faisal Mehmood[1]

[1]Department of Petroleum and Gas Engineering, University of Engineering and Technology, Lahore, Pakistan.

*Corresponding author (mkzahoor@yahoo.com)- 33 -

ABSTRACT: *Software's has become the utmost part of routine technical and non-technical assignments. Moreover, they are still even further influencing engineering and industrial work. Due to this reason learning programming languages is becoming more important for better future prospects, particularly for the students. This paper gives a good insight of comparative programming in two different languages, i.e, Microsoft Visual Basic and C#. To accomplish this task, one of the most commonly calculated parameter in petroleum and process industry; gas gravity, has been coded in both languages, giving a comparison of coding in both languages while approaching towards the same target/ output.*

Keywords**:** Programming, Visual Basic, Software, C#

INTRODUCTION

Programming is becoming a kind of necessity in almost every sphere of life. One can learn it, to develop simple programs resulting into easing the daily work, requiring a lot of calculations. In most cases when the students enter in universities, they are not familiar with the programming and have the concepts like, what is it? How to do? is it difficult?. Do we really need to do and apply in our field of study?. The answer regarding the requirement to learn it is pretty obvious, i.e., "yes", because programming is becoming a kind of necessity in almost every sphere of life. Therefore, students of every branch of study should be compelled to become familiarize with coding and programming environments.

It is usually presumed that Visual Basic is comparatively easy to handle particularly for those having no programming experience, while C# (C Sharp) requires a bit more detailed information while coding. However, both are first-class programming languages, sharing the common language runtime in the .NET Framework [1-4]. Though some differences exist between the two, but they are equally powerful tools for the developers, providing multiple ways of coding while approaching towards the same endpoint (final result). To illustrate the programming technique in both languages, gas gravity calculation procedure for hydrocarbon gases has been chosen because of its widely practiced parameter, so explaining its calculation procedure in itself will not be a core issue.

GAS GRAVITY

Gas gravity can also be referred to as specific gravity and is the ratio of the apparent molecular weight of the gas to that of air [5]. Mathematically, it can be derived as follows [6]:

Gas gravity in terms of density can be given as:

$$\gamma_g = \frac{\rho_{gas}}{\rho_{air}} \qquad (1)$$

For ideal gases, we know:

$$PV = nRT \qquad (2)$$

Where,

$$n = \frac{m}{MW_{app.}} \qquad (3)$$

Substituting equation (3) in (2), we get,

$$PV = \frac{m}{MW_{app.}}RT$$

$$\Rightarrow P \times MW_{app.} = \rho_{gas}RT \quad (\text{as, } \rho_{gas} = m/V)$$

$$\text{or, } \rho_{gas} = \frac{P \times MW_{app.}}{RT} \qquad (4)$$

Similarly for air:

$$\rho_{air} = \frac{P \times MW_{air}}{RT} \qquad (5)$$

Substituting equations (4) and (5) in equation (1), we get:

$$\gamma_g = \frac{P \times MW_{app.}}{RT} \times \frac{RT}{P \times MW_{air}}$$

$$\Rightarrow \gamma_g = \frac{MW_{app.}}{MW_{air}}$$

Where apparent molecular weight of gas can be given as [6]:

$$MW_{app.} = \sum_{i=1} y_i \cdot MW_i \qquad (6)$$

The above equation (6), shows that apparent molecular weight of any gas mixture can be calculated by adding the product of mole fraction of each component "i", and its molecular weight [6]. Also, total mole fraction will always be equal to "1".

PROGRAMMING PROCESS IN VISUAL BASIC AND C#

To develop a program (software) for calculating gas gravity by using both the languages for comparison purposes, text boxes has been used to insert and import data and as well as to export the calculated results. Text boxes have been used

as being one of the easily understandable concept and also at the same time requiring less coding proficiency. While programming, it should also be remembered that Visual Basic is not case-sensitive while C# is a case-sensitive language [2, 3, 7].

The developed program can be used to calculate gas gravity for a gas mixture comprising of up to ten components. The coding has been accomplished by using Visual Studio. Following form/platform as shown in Figure 1 has been created. The figure also explains the layout and working of the program

Coding for Importing Values

Under button named "Import Values", the following code (in Visual Basic and in C#) has been written, as given in Table 1. So that if the textboxes contains the specific hydrocarbon component (C1, C2,……,C10), then their molecular weight will be automatically imported in the specified boxes with a single click, while running the program. While, in case of other components, the required component and correspondingly its molecular weight can be inserted manually.

Coding for Calculating Gas Gravity

To calculate gas gravity from the available data the coding have been done in the following manner (Table 2) under the "Calculate" button (as shown in figure 1), using Visual Basic and C# programming language. To prevent the execution error (in case of empty textboxes) while calculating gas gravity of gas mixtures having less than ten components, by using C# code as given in Table 2, the respective C# code is modified as mentioned in Appendix 1.

Coding for Exiting from the Program

To exit from the application the following code (Table 3) has been used under "Exit" button:

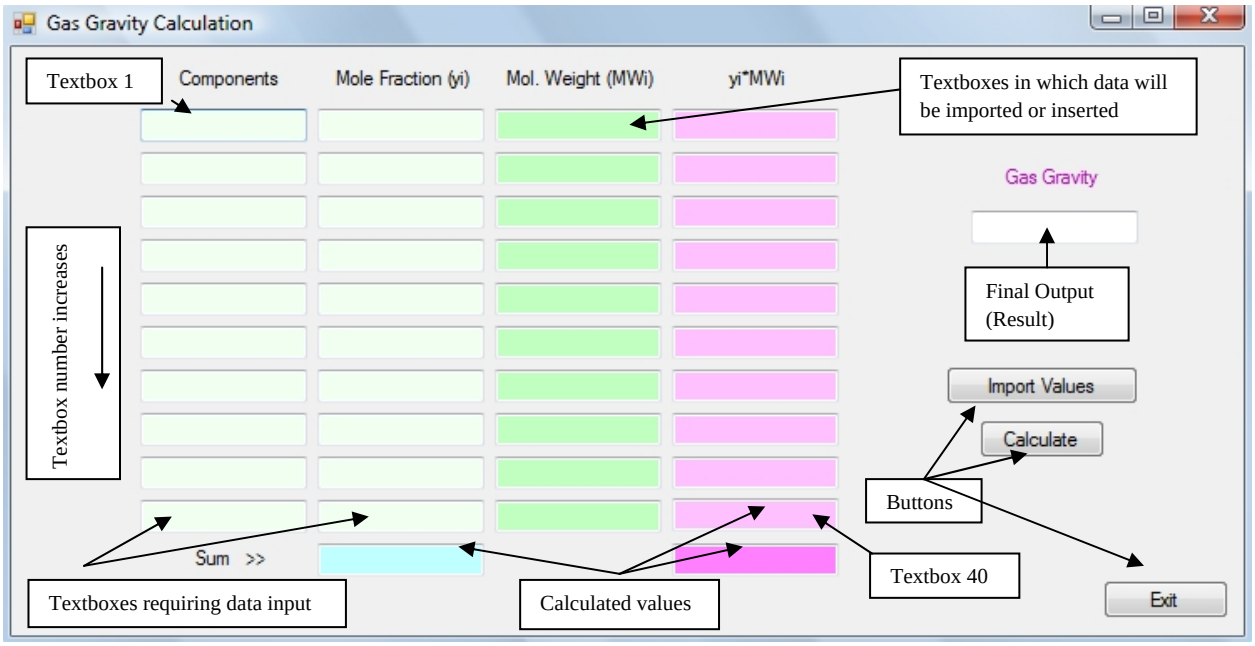
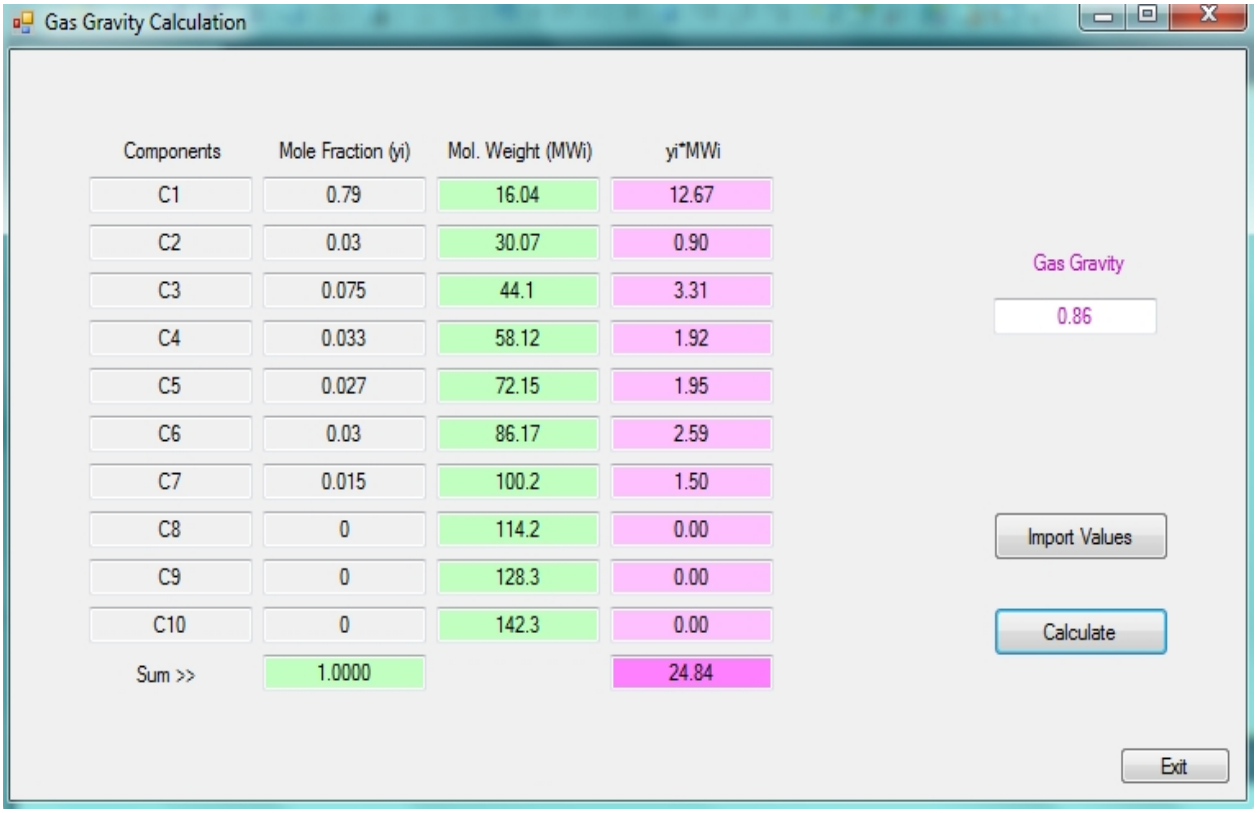


Figure 1: Layout of the platform to calculate gas gravity

RESULTS

The final results obtained in both cases are same. The difference is in the way in which data is entered leading to execution of the process (as mentioned in comments in Table 2, while coding for C#, that if simple code is used, then text boxes containing data required for calculation, must have some value). The following figure 2 shows, the results obtained by using the mentioned codes in Table 1 to 3, for a gas mixture having seven components. To avoid execution error in case of C# empty textboxes have been assigned a value "0" (zero) as shown in figure (2b), as zero will not affect the final output. While when the coding in C# (under "Calculate" button) as mentioned in Appendix 1 is used, the layout and the results becomes similar to as shown in Figure 2a. Figure (3) shows the output generated in case of gas mixture comprising of ten components. In this case, layout as well as the generated results are the same, as all the textboxes will be filled or have values.

(a)Based on coding in Microsoft Visual Basic



(b) Based on coding in Microsoft C#

Figure 2: Gas gravity calculations for gas mixture having less than ten components

Figure 3: Gas gravity calculation for gas mixture having ten components

CONCLUSIONS

Programming can be used to reduce the work load and simple coding can be done without knowing the programming languages at the fullest. The results show that how programming can be accomplished in different ways using Visual Basic and C#, while achieving the same final result(s). In addition, it gives an idea that how both of these languages, differs from each other in structure, while coding. The coding lines mentioned to calculate gas gravity can be decreased in these programming languages, for example, by using "datagridview" option instead of textboxes, but it requires, some proficiency in any of these languages.

NOMENCLATURE

m – Mass
n – Number of moles
P – Pressure
R – Gas constant
T – Temperature
V – Volume
$MW_{app.}$ – Apparent molecular weight of the gas mixture
MW_{air} – Molecular weight of air
MW_i – Molecular weight of each component in the gas mixture
y_i – Mole fraction of each component in the gas mixture
ρ – Density
γ_g – Gas gravity

REFERENCES

[1] C. Utley, A Programmer's Introduction to Visual Basic(R).Net, Sams Publishing, USA (2002).
[2] D. Zak, Programming with Microsoft - Visual Basic 2008. 4th ed., Course Technology, Cengage Learning, Boston (2010).
[3] J. Farrell, Microsoft Visual C# 2008: An Introduction to Object-Oriented Programming, 3rd ed., Course Technology, Cengage Learning, Boston (2008).
[4] H. Schildt, The Complete Reference: C# 3.0., McGraw-Hill, USA (2009).
[5] B. Guo, W.C. Lyons, and A. Ghalambor, Petroleum Production Engineering: A Computer - Assisted Approach, Elsevier Inc., Oxford (2007).
[6] T. Ahmed, Hydrocarbon Phase Behavior, Gulf Pub. Co., Houston (1989).
[7] P. J. Deitel, and H.M. Deitel, Visual Basic 2008 - How to Program, Pearson Education, Inc., New Jersey (2009).

Table 1: Coding for importing molecular weight of gas mixture components

In Visual Basic	In C#

If TextBox1.Text = "C1" Then
TextBox21.Text = 16.04
End If
If TextBox2.Text = "C2" Then
TextBox22.Text = 30.07
End If
If TextBox3.Text = "C3" Then
TextBox23.Text = 44.1
End If
	{
	
	if (textBox1.Text.Contains("C1"))
	textBox21.Text = 16.04.ToString();
	if (textBox2.Text.Contains("C2"))
	textBox22.Text = 30.07.ToString();
	if (textBox3.Text.Contains("C3"))
	textBox23.Text = 44.1.ToString();

```
If TextBox4.Text = "C4" Then
  TextBox24.Text = 58.12
End If
If TextBox5.Text = "C5" Then
  TextBox25.Text = 72.15
End If
If TextBox6.Text = "C6" Then
  TextBox26.Text = 86.17
End If
If TextBox7.Text = "C7" Then
  TextBox27.Text = 100.2
End If
If TextBox8.Text = "C8" Then
  TextBox28.Text = 114.2
End If
If TextBox9.Text = "C9" Then
  TextBox29.Text = 128.3
End If
If TextBox10.Text = "C10" Then
  TextBox30.Text = 142.3
End If
```

```
if (textBox4.Text.Contains("C4"))
  textBox24.Text = 58.12.ToString();
if (textBox5.Text.Contains("C5"))
  textBox25.Text = 72.15.ToString();
if (textBox6.Text.Contains("C6"))
  textBox26.Text = 86.17.ToString();
if (textBox7.Text.Contains("C7"))
  textBox27.Text = 100.2.ToString();
if (textBox8.Text.Contains("C8"))
  textBox28.Text = 114.2.ToString();
if (textBox9.Text.Contains("C9"))
  textBox29.Text = 128.3.ToString();
if (textBox10.Text.Contains("C10"))
  textBox30.Text = 142.3.ToString();

}
```

Table 2: Coding for calculating gas gravity

In Visual Basic

```
'To calculate gas gravity
'M represents M.W
Dim y1M1, y2M2, y3M3, y4M4, y5M5, y6M6,
y7M7, y8M8, y9M9, y10M10, summolefraction,
sumyiMi, gasgravity
y1M1 = Val(TextBox11.Text) * Val(TextBox21.Text)
TextBox31.Text = Format(y1m1, "0.00")
y2M2 = Val(TextBox12.Text) * Val(TextBox22.Text)
TextBox32.Text = Format(y2m2, "0.00")
y3M3 = Val(TextBox13.Text) * Val(TextBox23.Text)
TextBox33.Text = Format(y3m3, "0.00")
y4M4 = Val(TextBox14.Text) * Val(TextBox24.Text)
TextBox34.Text = Format(y4m4, "0.00")
y5M5 = Val(TextBox15.Text) * Val(TextBox25.Text)
TextBox35.Text = Format(y5m5, "0.00")
y6M6 = Val(TextBox16.Text) * Val(TextBox26.Text)
TextBox36.Text = Format(y6m6, "0.00")
y7M7 = Val(TextBox17.Text) * Val(TextBox27.Text)
TextBox37.Text = Format(y7m7, "0.00")
y8M8 = Val(TextBox18.Text) * Val(TextBox28.Text)
TextBox38.Text = Format(y8m8, "0.00")
y9M9 = Val(TextBox19.Text) * Val(TextBox29.Text)
 TextBox39.Text = Format(y9m9, "0.00")
y10M10 = Val(TextBox20.Text) *
Val(TextBox30.Text)
TextBox40.Text = Format(y10m10, "0.00")
summolefraction = Val(TextBox11.Text) +
Val(TextBox12.Text) + Val(TextBox13.Text) +
Val(TextBox14.Text) + Val(TextBox15.Text) +
Val(TextBox16.Text) + Val(TextBox17.Text) +
Val(TextBox18.Text) + Val(TextBox19.Text) +
Val(TextBox20.Text)
TextBox41.Text = Format(summolefraction,
"0.0000")
sumyiMi = y1M1 + y2M2 + y3M3 + y4M4 + y5M5 +
y6M6 + y7M7 + y8M8 + y9M9 + y10M10
```

In C#

```
// To calculate gas gravity
//Simple code but no textbox should be empty
otherwise, will give error while execution
//To avoid the error every text box apart from the ones
meant for components, should have some value
//M represents M.W

{
 double y1M1, y2M2, y3M3, y4M4, y5M5, y6M6,
y7M7, y8M8, y9M9, y10M10,
summolefraction,sumyiMi,mwair,gasgravity;
y1M1 = double.Parse(textBox11.Text) *
double.Parse(textBox21.Text);
 textBox31.Text = y1M1.ToString("0.00");
y2M2 = double.Parse(textBox12.Text) *
double.Parse(textBox22.Text);
 textBox32.Text = y2M2.ToString("0.00");
y3M3 = double.Parse(textBox13.Text) *
double.Parse(textBox23.Text);
 textBox33.Text = y3M3.ToString("0.00");
y4M4 = double.Parse(textBox14.Text) *
double.Parse(textBox24.Text);
 textBox34.Text = y4M4.ToString("0.00");
y5M5 = double.Parse(textBox15.Text) *
double.Parse(textBox25.Text);
 textBox35.Text = y5M5.ToString("0.00");
y6M6 = double.Parse(textBox16.Text) *
double.Parse(textBox26.Text);
 textBox36.Text = y6M6.ToString("0.00");
y7M7 = double.Parse(textBox17.Text) *
double.Parse(textBox27.Text);
 textBox37.Text = y7M7.ToString("0.00");
y8M8 = double.Parse(textBox18.Text) *
double.Parse(textBox28.Text);
 textBox38.Text = y8M8.ToString("0.00");
y9M9 = double.Parse(textBox19.Text) *
```

TextBox42.Text = Format(sumyiMi, "0.00")
gasgravity = Val(TextBox42.Text) / 28.97
TextBox43.Text = Format(gasgravity, "0.00")
	double.Parse(textBox29.Text);
	textBox39.Text = y9M9.ToString("0.00");
	y10M10 = double.Parse(textBox20.Text) *
	double.Parse(textBox30.Text);
	textBox40.Text = y10M10.ToString("0.00");
	summolefraction = double.Parse(textBox11.Text) +
	double.Parse(textBox12.Text) +
	double.Parse(textBox13.Text) +
	double.Parse(textBox14.Text) +
	double.Parse(textBox15.Text) +
	double.Parse(textBox16.Text) +
	double.Parse(textBox17.Text) +
	double.Parse(textBox18.Text) +
	double.Parse(textBox19.Text) +
	double.Parse(textBox20.Text);
	textBox41.Text = summolefraction.ToString("0.0000");
	sumyiMi = y1M1 + y2M2 + y3M3 + y4M4 + y5M5 +
	y6M6 + y7M7 + y8M8 + y9M9 + y10M10;
	textBox42.Text = sumyiMi.ToString("0.00");
	gasgravity = double.Parse(textBox42.Text) / (mwair =
	28.97);
	textBox43.Text = gasgravity.ToString("0.00");
	}

Table 3: Code to exit from the application

In Visual Basic	In C#
'Code to exit the application under exit button End	//Code to exit the application under exit button { Application.Exit(); }

APPENDIX 1

Modified C# Code to Calculate Gas Gravity

```
// In this code empty textboxes has been assigned a “0” (zero) value{
 double y1M1, y2M2, y3M3, y4M4, y5M5, y6M6, y7M7, y8M8, y9M9, y10M10, summolefraction, sumyiMi, mwair, gasgravity;
y1M1 = Convert.ToDouble((textBox11.Text != string.Empty ? textBox11.Text : 0.ToString())) * Convert.ToDouble((textBox21.Text != string.Empty ? textBox21.Text : 0.ToString()));
textBox31.Text = y1M1.ToString("0.00");
y2M2 = Convert.ToDouble((textBox12.Text != string.Empty ? textBox12.Text : 0.ToString())) * Convert.ToDouble((textBox22.Text != string.Empty ? textBox22.Text : 0.ToString()));
textBox32.Text = y2M2.ToString("0.00");
y3M3 = Convert.ToDouble((textBox13.Text != string.Empty ? textBox13.Text : 0.ToString())) * Convert.ToDouble((textBox23.Text != string.Empty ? textBox23.Text : 0.ToString()));
textBox33.Text = y3M3.ToString("0.00");
y4M4 = Convert.ToDouble((textBox14.Text != string.Empty ? textBox14.Text : 0.ToString())) * Convert.ToDouble((textBox24.Text != string.Empty ? textBox24.Text : 0.ToString()));
textBox34.Text = y4M4.ToString("0.00");
y5M5 = Convert.ToDouble((textBox15.Text != string.Empty ? textBox15.Text : 0.ToString())) * Convert.ToDouble((textBox25.Text != string.Empty ? textBox25.Text : 0.ToString()));
textBox35.Text = y5M5.ToString("0.00");
y6M6 = Convert.ToDouble((textBox16.Text != string.Empty ? textBox16.Text : 0.ToString())) * Convert.ToDouble((textBox26.Text != string.Empty ? textBox26.Text : 0.ToString()));
textBox36.Text = y6M6.ToString("0.00");
y7M7 = Convert.ToDouble((textBox17.Text != string.Empty ? textBox17.Text : 0.ToString())) * Convert.ToDouble((textBox27.Text != string.Empty ? textBox27.Text : 0.ToString()));
textBox37.Text = y7M7.ToString("0.00");
y8M8 = Convert.ToDouble((textBox18.Text != string.Empty ? textBox18.Text : 0.ToString())) * Convert.ToDouble((textBox28.Text != string.Empty ? textBox28.Text : 0.ToString()));
textBox38.Text = y8M8.ToString("0.00");
y9M9 = Convert.ToDouble((textBox19.Text != string.Empty ? textBox19.Text : 0.ToString())) * Convert.ToDouble((textBox29.Text != string.Empty ? textBox29.Text : 0.ToString()));
textBox39.Text = y9M9.ToString("0.00");
y10M10 = Convert.ToDouble((textBox20.Text != string.Empty ? textBox20.Text : 0.ToString())) * Convert.ToDouble((textBox30.Text != string.Empty ? textBox30.Text : 0.ToString()));
textBox40.Text = y10M10.ToString("0.00");
```

```
summolefraction = Convert.ToDouble((textBox11.Text != string.Empty ? textBox11.Text : 0.ToString())) +
Convert.ToDouble((textBox12.Text != string.Empty ? textBox12.Text : 0.ToString())) +
Convert.ToDouble((textBox13.Text != string.Empty ? textBox13.Text : 0.ToString())) +
Convert.ToDouble((textBox14.Text != string.Empty ? textBox14.Text : 0.ToString())) +
Convert.ToDouble((textBox15.Text != string.Empty ? textBox15.Text : 0.ToString())) +
Convert.ToDouble((textBox16.Text != string.Empty ? textBox16.Text : 0.ToString())) +
Convert.ToDouble((textBox17.Text != string.Empty ? textBox17.Text : 0.ToString())) +
Convert.ToDouble((textBox18.Text != string.Empty ? textBox18.Text : 0.ToString())) +
Convert.ToDouble((textBox19.Text != string.Empty ? textBox19.Text : 0.ToString())) +
Convert.ToDouble((textBox20.Text != string.Empty ? textBox20.Text : 0.ToString()));
textBox41.Text = summolefraction.ToString("0.0000");
sumyiMi = y1M1 + y2M2 + y3M3 + y4M4 + y5M5 + y6M6 + y7M7 + y8M8 + y9M9 + y10M10;
textBox42.Text = sumyiMi.ToString("0.00");
gasgravity = double.Parse(textBox42.Text) / (mwair = 28.97);
textBox43.Text = gasgravity.ToString("0.00");
}
```