Project Reports

.NET Programming

Chhay Pheaktra (No. 04)

Class E1 Year3

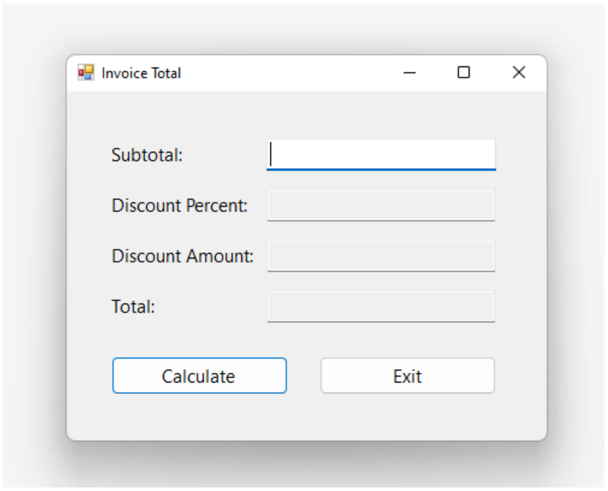
Contents

1. Chapter 02: How to design a Windows Form application

__Design only

1.1. Project 01: Invoice Total

❖ Preview



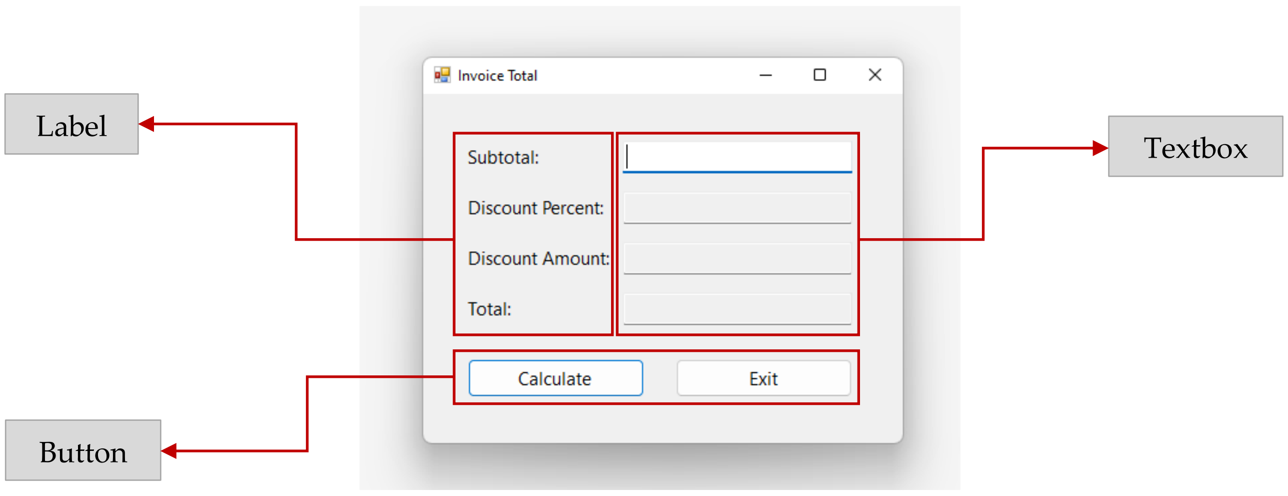
❖ Objectives

- ✓ How to design a windows form for calculating Discount Percent, Discount Amount and Total by providing only one value (Subtotal)
- ✓ How to use controls (Toolbox) to design Invoice Total form
- ✓ How to set property to each control for this windows form

❖ Applications

- ✓ Controls

This form is designed by 3 controls such us: **Label**, **Textbox** and **Button**. This form uses 4 Labels to specified 4 Textboxes and 2 Buttons, 1 for Calculate and 1 for Exit the form by you can see in the form below.



✓ Properties

After finished designing the form I have set the property to each control below:

➢ **Form**

Default name	Property	Value
Form1	Text	Invoice Total
	ActionButton	btnCalculate
	CancelButton	btnExit
	StartPosition	CenterScreen

➢ **Labels**

Default name	Property	Value
label1	Text	Subtotal:
	TextAlign	MiddleLeft
	TabIndex	0
label2	Text	Discount Percent:
	TextAlign	MiddleLeft
label3	Text	Discount Amount:

	TextAlign	MiddleLeft
label4	Text	Total:
	TextAlign	MiddleLeft

➢ Textboxes

Default name	Property	Value
textBox1	Name	txtSubTotal
	TabIndex	1
textBox2	Name	txtDiscountPercent
	ReadOnly	True
	TabStop	False
textBox3	Name	txtDiscountAmount
	ReadOnly	True
	TabStop	False
textBox4	Name	txtTotal
	ReadOnly	True
	TabStop	False

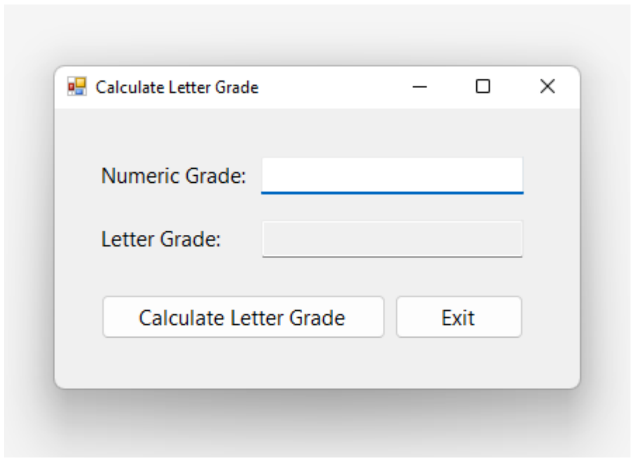
➢ Buttons

Default name	Property	Value
button1	Name	btnCalculate
	Text	&Calculate
	TabIndex	2
button2	Name	btnExit
	Text	E&xit
	TabIndex	3

Note: *This form cannot calculate yet because it is not imported code yet.*

1.2. Project 02: Calculate Letter Grade

❖ Preview



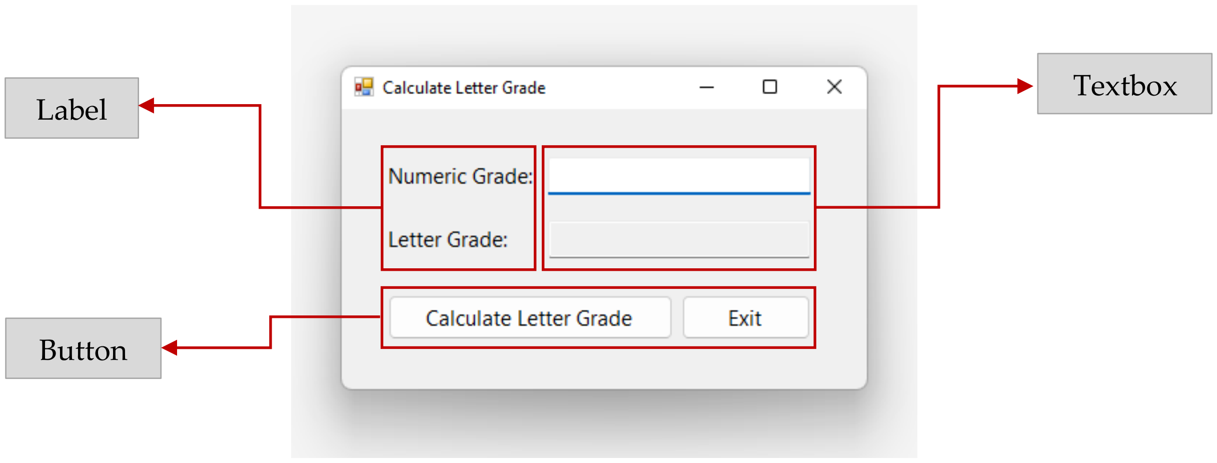
❖ Objectives

- ✓ How to design a windows form for calculating a grade of number that input in the Number Grade field then output the result in the Letter Grade field as a letter from A-F
- ✓ How to use controls (Toolbox) to design Calculate Letter Grade form
- ✓ How to set property to each control for this windows form

❖ Applications

- ✓ **Controls**

This form is designed by 3 controls such us: **Label**, **Textbox** and **Button**. This form uses 2 Labels to specified 2 Textboxes and 2 Buttons, 1 for Calculate and 1 for Exit the form by you can see in the form below.



✓ Properties

After finished designing the form I have set the property to each control below:

➤ **Form**

Default name	Property	Value
Form1	Text	Calculate Letter Grade
	ActionButton	btnCalculate
	CancelButton	btnExit
	StartPosition	CenterScreen

➤ **Labels**

Default name	Property	Value
label1	Text	Numeric Grade:
	TextAlign	MiddleLeft
	TabIndex	0
label2	Text	Letter Grade:
	TextAlign	MiddleLeft

➢ Textboxes

Default name	Property	Value
textBox1	Name	txtNumericGrade
	TabIndex	1
textBox2	Name	txtLetterGrade
	ReadOnly	True
	TabStop	False

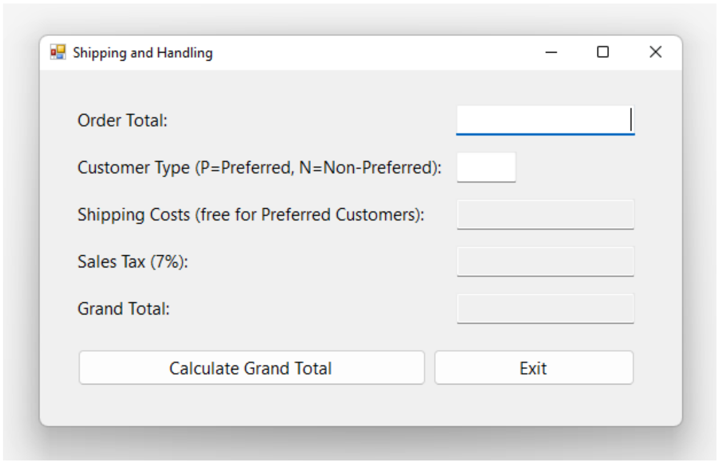
➢ Buttons

Default name	Property	Value
button1	Name	btnCalculate
	Text	&Calculate Letter Grade
	TabIndex	2
button2	Name	btnExit
	Text	E&xit
	TabIndex	3

Note: *This form cannot calculate yet because it is not imported code yet.*

1.3. Project 03: Shipping and Handling

❖ Preview


Shipping and Handling
Order Total:
Customer Type (P=Preferred, N=Non-Preferred):
Shipping Costs (free for Preferred Customers):
Sales Tax (7%):
Grand Total:
Calculate Grand Total
Exit

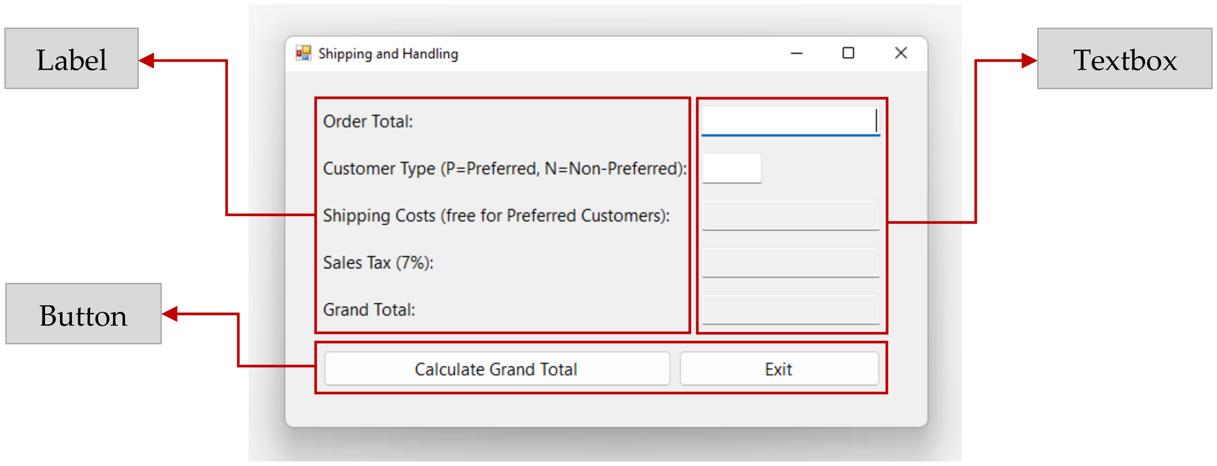
❖ Objectives

- ✓ How to design a windows form for calculating Shipping Costs, Sales Tax and Grand Total that need 2 values are Order Total and Customer Type
- ✓ How to use controls (Toolbox) to design Shipping and Handling form
- ✓ How to set property to each control for this windows form

❖ Applications

- ✓ **Controls**

This form is designed by 3 controls such us: **Label**, **Textbox** and **Button**. This form uses 5 Labels to specified 5 Textboxes and 2 Buttons, 1 for Calculate and 1 for Exit the form by you can see in the form below.



✓ Properties

After finished designing the form I have set the property to each control below:

➢ **Form**

Default name	Property	Value
Form1	Text	Shipping and Handling
	ActionButton	btnCalculate
	CancelButton	btnExit
	StartPosition	CenterScreen

➢ **Labels**

Default name	Property	Value
label1	Text	Order Total:
	TextAlign	MiddleLeft
	TabIndex	0
label2	Text	Customer Type (P=Preferred, N=Non-Prefrred):
	TextAlign	MiddleLeft
label3	Text	Shipping Costs (free for Preferred Customer):
	TextAlign	MiddleLeft
label4	Text	Sales Tax (7%)

	TextAlign	MiddleLeft
label5	Text	Grand Total:
	TextAlign	MiddleLeft

➢ Textbox

Default name	Property	Value
textBox1	Name	txtOrderTotal
	TextAlign	MiddleRight
	TabIndex	1
textBox2	Name	txtCustomerType
	TextAlign	MiddleRight
	TabIndex	2
textBox3	Name	txtShippingCosts
	TextAlign	MiddleRight
	ReadOnly	True
	TabStop	False
textBox4	Name	txtSalesTax
	TextAlign	MiddleRight
	ReadOnly	True
	TabStop	False
textBox5	Name	txtGrandTotal
	TextAlign	MiddleRight
	ReadOnly	True
	TabStop	False

➢ Button

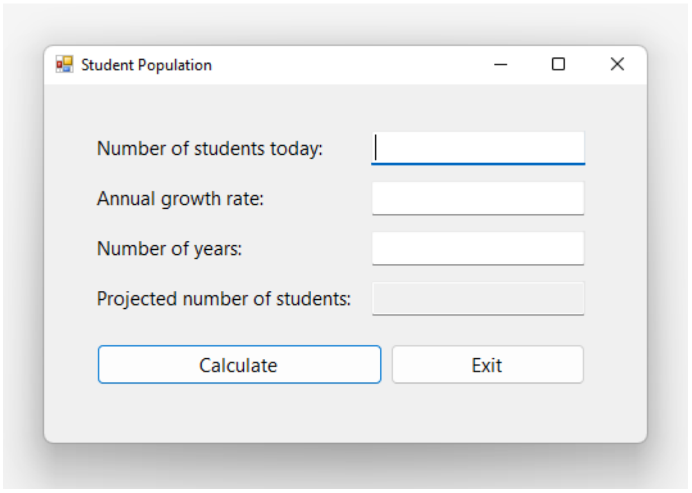
Default name	Property	Value
button1	Name	btnCalculate
	Text	&Calculate Letter Grade

	TabIndex	2
button2	Name	btnExit
	Text	E&xit
	TabIndex	3

Note: *This form cannot calculate yet because it is not imported code yet.*

1.4. Project 04: Student Population

❖ Preview



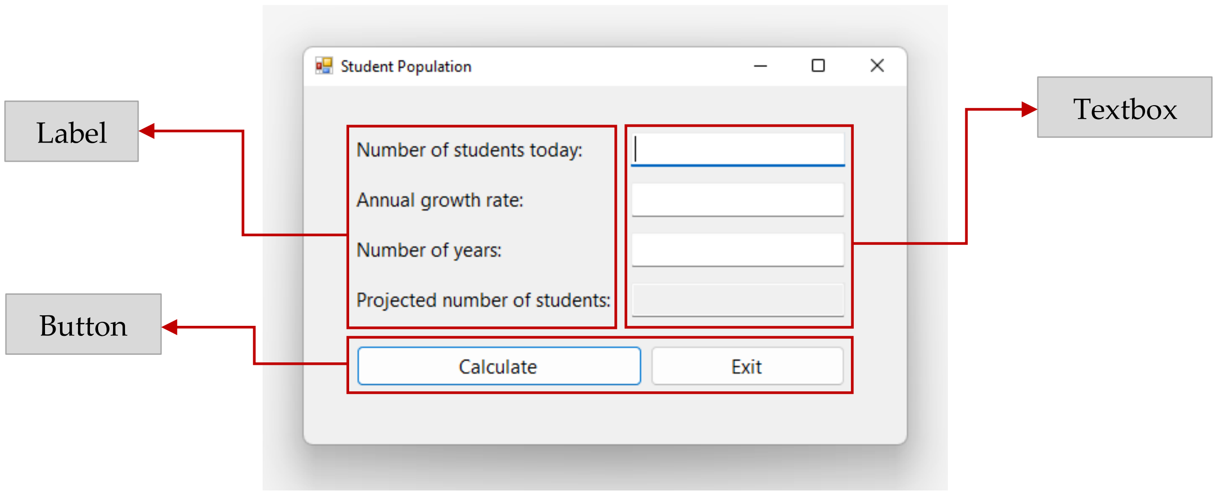
❖ Objectives

- ✓ How to design a windows form for calculating Projected number of student that needs 3 values are Number of students today, Annual growth rate and Number of years
- ✓ How to use controls (Toolbox) to design Student Population form
- ✓ How to set property to each control for this windows form

❖ Applications

✓ Controls

This form is designed by 3 controls such us: **Label**, **Textbox** and **Button**. This form uses 4 Labels to specified 4 Textboxes and 2 Buttons, 1 for Calculate and 1 for Exit the form by you can see in the form below.



- ✓ Properties

After finished designing the form I have set the property to each control below:

- ➢ **Form**

Default name	Property	Value
Form1	Text	Student Population
	ActionButton	btnCalculate
	CancelButton	btnExit
	StartPosition	CenterScreen

- ➢ **Labels**

Default name	Property	Value
label1	Text	Number of students today:
	TextAlign	MiddleLeft
	TabIndex	0
label2	Text	Annual growth rate:
	TextAlign	MiddleLeft
label3	Text	Number of years:
	TextAlign	MiddleLeft
label4	Text	Projected number of students:

	TextAlign	MiddleLeft

➢ Textbox

Default name	Property	Value
textBox1	Name	txtStuNumNow
	TabIndex	1
textBox2	Name	txtAGR
	TabIndex	2
textBox3	Name	txtNumYear
	TabIndex	3
textBox4	Name	txtStuNumProjected
	ReadOnly	True
	TabStop	False

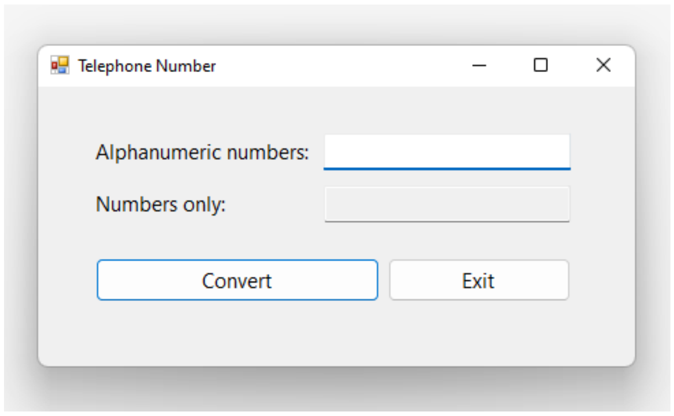
➢ Button

Default name	Property	Value
button1	Name	btnCalculate
	Text	&Calculate
	TabIndex	2
button2	Name	btnExit
	Text	E&xit
	TabIndex	3

Note: *This form cannot calculate yet because it is not imported code yet.*

1.5. Project 05: Telephone Number

❖ Preview



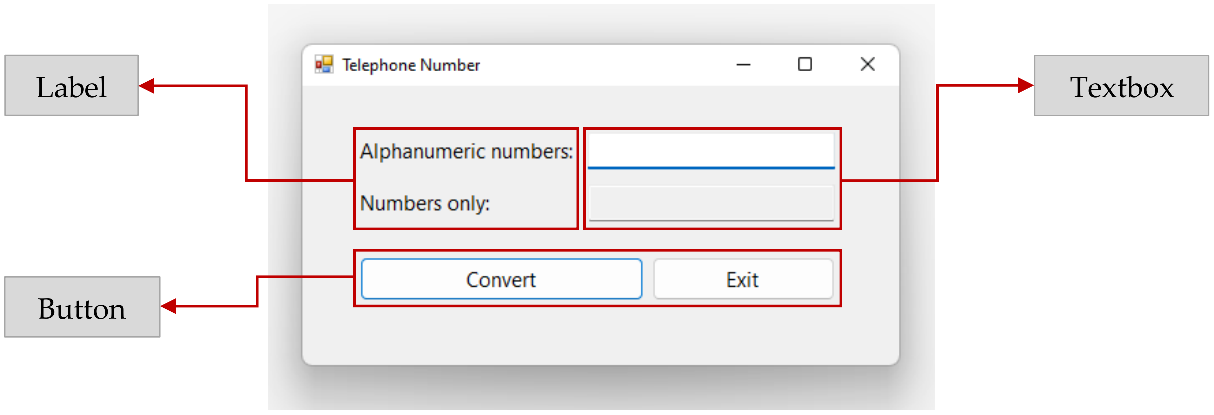
❖ Objectives

- ✓ How to design a windows form for converting Alphanumeric numbers to Number only
- ✓ How to use controls (Toolbox) to design Calculate Letter Grade form
- ✓ How to set property to each control for this windows form

❖ Applications

- ✓ **Controls**

This form is designed by 3 controls such us: **Label**, **Textbox** and **Button**. This form uses 2 Labels to specified 2 Textboxes and 2 Buttons, 1 for Calculate and 1 for Exit the form by you can see in the form below.



- ✓ Properties

After finished designing the form I have set the property to each control below:

- ➢ **Form**

Default name	Property	Value
Form1	Text	Telephone Number
	ActionButton	btnConvert
	CancelButton	btnExit
	StartPosition	CenterScreen

- ➢ **Labels**

Default name	Property	Value
label1	Text	Alphanumeric numbers:
	TextAlign	MiddleLeft
	TabIndex	0
label2	Text	Numbers only:
	TextAlign	MiddleLeft

➢ Textboxes

Default name	Property	Value
textBox1	Name	txtAlphaNum
	TabIndex	1
textBox2	Name	txtNum
	ReadOnly	True
	TabStop	False

➢ Buttons

Default name	Property	Value
button1	Name	btnConvert
	Text	&Convert
	TabIndex	2
button2	Name	btnExit
	Text	E&xit
	TabIndex	3

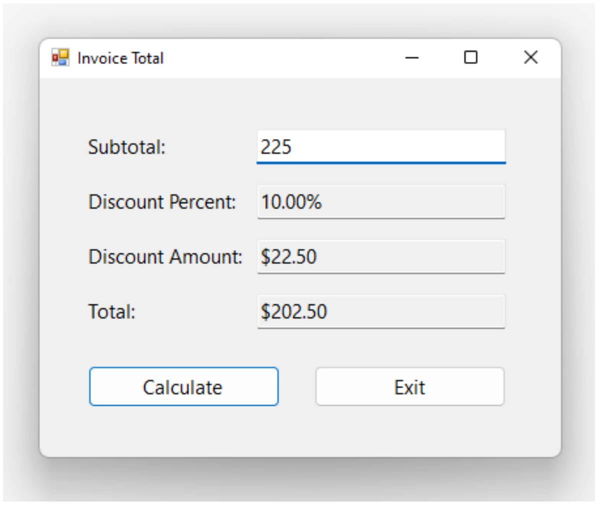
Note: *This form cannot calculate yet because it is not imported code yet.*

2. Chapter 03: How to code and test a Windows Form application

__Applying code

2.1. Project 01: Invoice Total

❖ Preview



Subtotal: 225
Discount Percent: 10.00%
Discount Amount: $22.50
Total: $202.50

❖ Objectives

This form is designed for calculating Discount Percent, Discount Amount and Total by providing only one value (Subtotal). By the way, when we enter a value to Subtotal field and click the Calculate button then other fields will automatically fill by itself. They can do that because of code that we apply to this form by following business rule below. And if you want to exit from the form you can click on Exit button instead of close button on top right.

❖ Applications

- ✓ Business rules
  - If the Subtotal is least than $100 then discount 0%
  - If the Subtotal between $100 and $250 then discount 10%
  - If the Subtotal between $250 and $500 then discount 15%
  - If the Subtotal is more than $500 then discount 20%
- ✓ Codes

In this form use 2 functions for Calculate and Exit.

For function Calculate have 4 data members

- subtotal (decimal): get value from txtSubtotal and convert to decimal
- discPercent (decimal): default value is 0
- discAmount (decimal): find discount amount by subtotal * discPercent
- invoiceTotal (decimal): find total cost by subtotal – discAmount

function Exit

Event Click

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}
```

Close method uses to close application

function Calculate

Data members

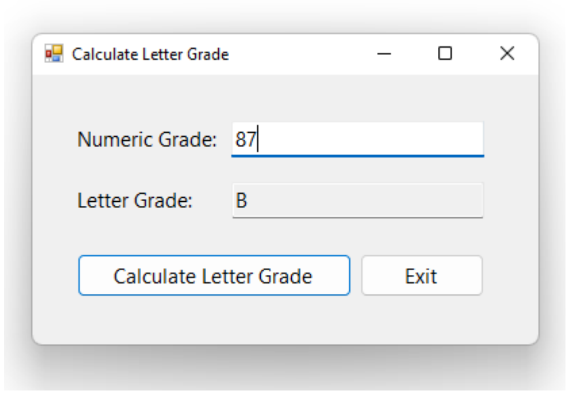
By following the business rules we get these conditions;

Output result to textbox

```
private void btnCalculate_Click(object sender, EventArgs e)
{
        decimal subtotal = Convert.ToDecimal(txtSubtotal.Text);
        decimal discPercent = 0m;

        if (subtotal > 0 && subtotal < 100) discPercent = 0m;
        else if (subtotal >= 100 && subtotal < 250) discPercent = 0.1m;
        else if (subtotal >= 250 && subtotal < 500) discPercent = 0.15m;
        else if (subtotal > 500) discPercent = 0.2m;

        decimal discAmount = subtotal * discPercent;
        decimal invoiceTotal = subtotal - discAmount;

        txtDiscountPecent.Text = discPercent.ToString("P");
        txtDiscountAmount.Text = discAmount.ToString("C");
        txtTotal.Text = invoiceTotal.ToString("C");

        txtSubtotal.Focus();
}
```

2.2. Project 02: Calculate Letter Grade

❖ Preview



Numeric Grade: 87

Letter Grade: B

❖ Objectives

This form is designed for calculating a grade of number that input in the Number Grade field then output the result in the Letter Grade field as a letter from A-F.

❖ Applications

✓ Business rules

- If the Numeric Grade between 90 and 100 then Letter Grade is A
- If the Numeric Grade between 80 and 90 then Letter Grade is B
- If the Numeric Grade between 70 and 80 then Letter Grade is C
- If the Numeric Grade between 60 and 70 then Letter Grade is D
- If the Numeric Grade between 0 and 60 then Letter Grade is F

✓ Codes

In this form use 2 functions for Calculate and Exit.

For function Calculate have 2 data members:

- alphanum (decimal): get value from txtNumGrade and convert to decimal
- letterGrade (char): declare for getting value and output

function Exit

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}
```

Event Click

Close method uses to close application

```
private void btnCalculate_Click(object sender, EventArgs e)
{
```

function Calculate

By following the business rules we get these conditions.

Data members

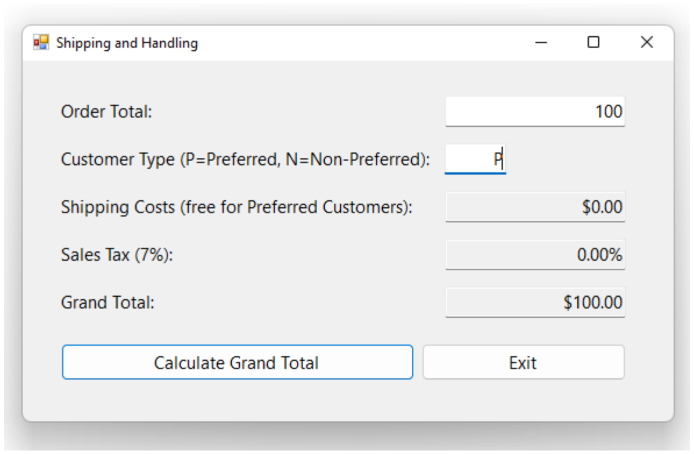
```
        decimal alphanum = Convert.ToDecimal(txtNumGrade.Text);
        char letterGrade;

        if (alphanum >= 0 && alphanum < 60) letterGrade = 'F';
        else if (alphanum >= 60 && alphanum < 70) letterGrade = 'D';
        else if (alphanum >= 70 && alphanum < 80) letterGrade = 'C';
        else if (alphanum >= 80 && alphanum < 90) letterGrade = 'B';
        else if (alphanum >= 90 && alphanum <=100) letterGrade = 'A';
        else
        {
            MessageBox.Show("Please fill a number from 0 to 100");
            letterGrade = ' ';
        }

        txtLetterGrade.Text = letterGrade.ToString();
        txtNumGrade.Focus();
}
```

Output result to textbox.

We don't allow to enter a number that least than 0 or bigger than 100, so we add this condition to check the value come.

2.3. Project 03: Shipping and Handling

❖ Preview


Shipping and Handling

Order Total:	100
Customer Type (P=Preferred, N=Non-Preferred):	P
Shipping Costs (free for Preferred Customers):	$0.00
Sales Tax (7%):	0.00%
Grand Total:	$100.00

❖ Objectives

In this project we will apply code to the form Shipping and Handling that we have designed in chapter 2 for calculating the Shipping Costs and the Sales Tax that depend on Customer Type and calculating the Grand Total.

❖ Applications

✓ Business rules

- If the Customer Type is a Preferred Customer, then Sales Tax is 0%
- If the Customer Type is a Non-Preferred Customer, then Sales Tax is 7%

- ✓ Codes

  In this form use 3 functions for Calculate and Exit and clear.

  For function Calculate have 5 data members:

  - orderTotal (decimal): get value from txtOrderTotal and convert to decimal
  - cusType (string): get value from txtCustomerType
  - salesTax (decimal): default value is 0
  - shippingCosts (decimal): orderTotal * salesTax
  - grandTotal (decimal): orderTotal + shippingCosts

function Exit

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}
```

Event Click

Close method uses to close application

```
private void btnCalculate_Click(object sender, EventArgs e)
{
```

function Calculate

By following the business rules we get these conditions.

```
        decimal orderTotal = Convert.ToDecimal(txtOrderTotal.Text);
        string cusType = txtCustomerType.Text;
        decimal salesTax = 0m;

        if (cusType == "P" || cusType == "p") salesTax = 0m;
        else if (cusType == "N" || cusType == "n") salesTax = 0.07m;
        else
        {
                clear();
                orderTotal = 0m;
                MessageBox.Show("Customer Type should be P or N");
        }
```

Data members

```
        decimal shippingCost = orderTotal * salesTax;
        decimal grandTotal = orderTotal + shippingCost;

        txtShippingCosts.Text = shippingCost.ToString("C");
        txtSalesTax.Text = salesTax.ToString("P");
        txtGrandTotal.Text = grandTotal.ToString("C");

        txtCustomerType.Focus();
}
```

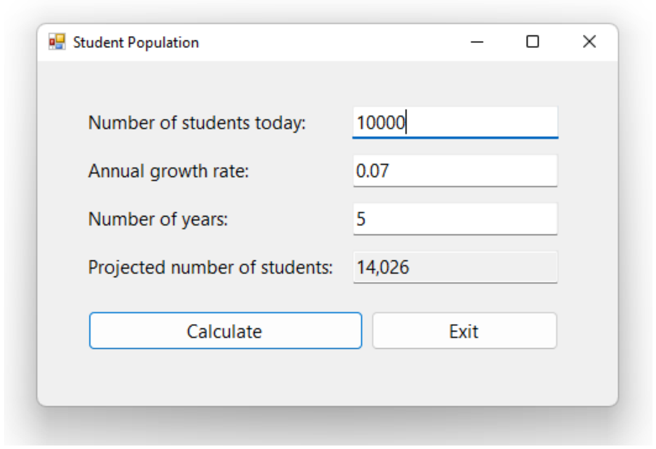
We allow you to enter only “P” or “No” to the Customer Type field. So, we call the function clear() then assign 0 to orderTotal and alert a message.

```
void clear()
{
        txtOrderTotal.Text = "";
        txtCustomerType.Text = "";
        txtShippingCosts.Text = "";
        txtSalesTax.Text = "";
        txtGrandTotal.Text = "";
}
```

function clear use for clearing the textboxes by assigning an empty value to each textbox.

2.4. Project 04: Student Population

❖ Preview


Student Population

Number of students today: 10000

Annual growth rate: 0.07

Number of years: 5

Projected number of students: 14,026

❖ Objectives

In this project we will apply code to the form Student Population that we have designed in chapter 2 for calculating the number of students in the future.

❖ Applications

✓ Business rules

To find Projected number of students we will follow the mathemathics rules:

$$V_{final} = V_{begin}(1 + agr)^t$$

- V_{final} = Projected number of students
- V_{begin} = Number of students today
- agr = Annual growth rate
- t = Number of years

✓ Codes

In this form use 2 functions for Calculate and Exit.

For function Calculate have 4 data members:

- stuNow (double): get value from txtStuNumNow and convert to double
- agr (double): get value from txtAGR and convert to double
- numYear (int): get value from txtNumYear and convert to int32
- stuNumProjected (double): default value is 0

function Exit

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}
```

Event Click

Close method uses to close application

```
private void btnCalculate_Click(object sender, EventArgs e)
{
```

function Calculate

Data members

```
        double stuNow = Convert.ToDouble(txtStuNumNow.Text);
        double agr = Convert.ToDouble(txtAGR.Text);
        int numYear = Convert.ToInt32(txtNumYear.Text);
        double stuNumProjected = 0;
```

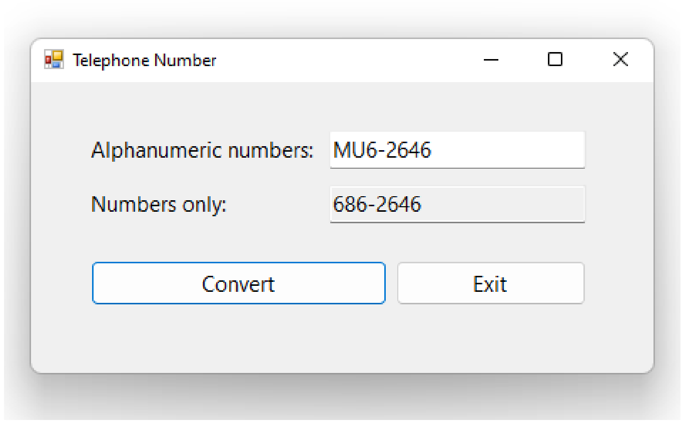
By following the math rule we get the code like this.

```
        stuNumProjected = stuNow * Math.Pow((1 + agr), numYear);

        txtStuNumProjected.Text = stuNumProjected.ToString("N0");

        txtStuNumNow.Focus();
}
```

Output result to textbox

2.5. Project 05: Telephone Number

❖ Preview



Alphanumeric numbers: MU6-2646

Numbers only: 686-2646

❖ Objectives

In this project we will apply code to the form Telephone Number that we have designed in chapter 2 for converting the alphanumeric numbers to numbers only.

❖ Applications

✓ Business rules

Numbers only	*Alphanumeric numbers*
2	A, B, C
3	D, E, F
4	G, H, I
5	J, K, L
6	M, N, O
7	P, Q, R, S
8	T, U, V
9	W, X, Y, Z

✓ Codes

In this form use 2 functions for Calculate and Exit.

For function Calculate have 4 data members:

- alpha (string): get value from txtAlphaNum and change to lowercase
- alphaArray (array): separate string to char array
- num (char): declare for getting char number
- res (string): result

function Exit

Event Click

Close method uses to close application

function Convert

Data members

We use foreach loop check condition and create a string by characters.

By following the business rules we get the codes like this.

Create a string by characters.

Output the string to textbox.

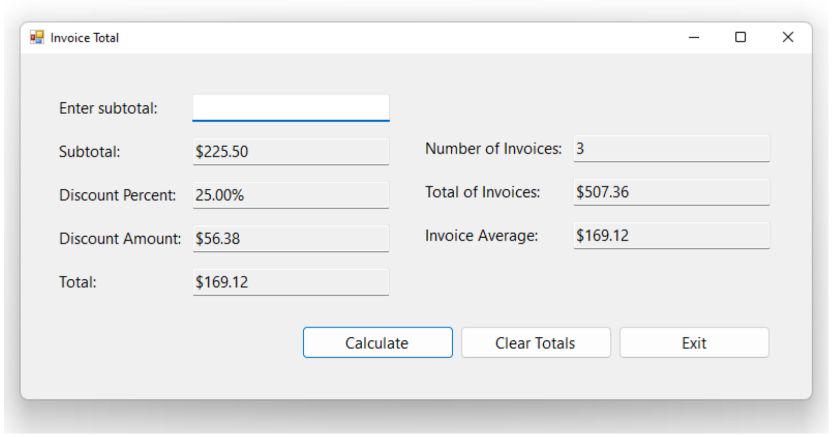
```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}

private void btnConvert_Click(object sender, EventArgs e)
{
        string alpha = txtAlphaNum.Text.ToLower();
        char[] alphaArray = alpha.ToCharArray();
        char num;
        string res = "";

        foreach (char c in alphaArray)
        {
                if (c == 'a' || c == 'b' || c == 'b') num = '2';
                else if (c == 'd' || c == 'e' || c == 'f') num = '3';
                else if (c == 'g' || c == 'h' || c == 'i') num = '4';
                else if (c == 'j' || c == 'k' || c == 'l') num = '5';
                else if (c == 'm' || c == 'n' || c == 'o') num = '6';
                else if (c == 'p' || c == 'q' || c == 'r' || c == 's') num = '7';
                else if (c == 't' || c == 'u' || c == 'v') num = '8';
                else if (c == 'w' || c == 'x' || c == 'y') num = '9';
                else num = c;

                res = res + num;
        }

        txtNum.Text = res;
}
```

3. Chapter 04: How to work with numeric and string data

__Update form Invoice Total by adding new field and update some codes

3.1. Project 01: Enhance the Invoice Total form

❖ Preview



Enter subtotal:			
Subtotal:	$225.50	Number of Invoices:	3
Discount Percent:	25.00%	Total of Invoices:	$507.36
Discount Amount:	$56.38	Invoice Average:	$169.12
Total:	$169.12		

❖ Objectives

We have designed the Invoice Total form in chapter 02 and apply codes in chapter 03. So, in this chapter we will add some new fields to the form and extra some codes to it. Also, in this form is for finding number of invoices, total of costs from all invoices and finding average of the invoice.

❖ Design

➢ Form

Default name	Property	Value
Form1	Text	Invoice Total
	ActionButton	btnCalculate
	CancelButton	btnExit
	StartPosition	CenterScreen

➢ Labels

Default name	Property	Value
label1	Text	Enter Subtotal:
	TextAlign	MiddleLeft
	TabIndex	0
label2	Text	Subtotal:
	TextAlign	MiddleLeft
label3	Text	Discount Percent:
	TextAlign	MiddleLeft
label4	Text	Discount Amount:
	TextAlign	MiddleLeft
label5	Text	Total:
	TextAlign	MiddleLeft
label6	Text	Number of Invoices:
	TextAlign	MiddleLeft
label7	Text	Total of Invoices:
	TextAlign	MiddleLeft
label8	Text	Invoice Average:
	TextAlign	MiddleLeft

➢ Textboxes

Default name	Property	Value
textBox1	Name	txtSubTotal
	TabIndex	1
textBox2	Name	txtViewSubTotal
	ReadOnly	True
	TabStop	False
textBox3	Name	txtDiscountPercent
	ReadOnly	True

	TabStop	False
textBox4	Name	txtDiscountAmount
	ReadOnly	True
	TabStop	False
textBox5	Name	txtTotal
	ReadOnly	True
	TabStop	False
textBox6	Name	txtInvoiceNum
	ReadOnly	True
	TabStop	False
textBox7	Name	txtInvoiceTotal
	ReadOnly	True
	TabStop	False
textBox8	Name	txtInvoiceAverage
	ReadOnly	True
	TabStop	False

➢ **Buttons**

Default name	**Property**	**Value**
button1	Name	btnCalculate
	Text	&Calculate
	TabIndex	2
button2	Name	btnClear
	Text	Clear &Totals
	TabIndex	3
button3	Name	btnExit
	Text	E&xit
	TabIndex	4

❖ Codes

```
private void btnExit_Click(object sender, EventArgs e)
{
            this.Close();
}
```

Global variables

```
int numOfInvoice = 0;
decimal totalOfInvoice = 0m;
decimal invoiceAverage = 0;
```

Data members

```
private void btnCalculate_Click(object sender, EventArgs e)
{
            decimal subtotal = Convert.ToDecimal(txtSubtotal.Text);
            decimal discPercent = 0.25m;
            decimal discAmount = Math.Round(subtotal * discPercent, 2);
            decimal invoiceTotal = subtotal - discAmount;

            txtViewSubtotal.Text = subtotal.ToString("C");
            txtDiscountPecent.Text = discPercent.ToString("P");
            txtDiscountAmount.Text = discAmount.ToString("C");
            txtTotal.Text = invoiceTotal.ToString("C");

            numOfInvoice++;
            totalOfInvoice += invoiceTotal;
            invoiceAverage = totalOfInvoice / numOfInvoice;

            txtInvoiceNum.Text = numOfInvoice.ToString();
            txtInvoiceTotal.Text = totalOfInvoice.ToString("C");
            txtInvoiceAverage.Text = invoiceAverage.ToString("C");

            txtSubtotal.Text = "";

            txtSubtotal.Focus();
}
```

Output result to textbox

Find number of invoices

Find total of totals

Find average of totals

Give empty value to Enter Subtotal field after calculated

Function Clear and add Event Click

```
private void btnClear_Click(object sender, EventArgs e)
{
        numOfInvoice = 0;
        totalOfInvoice = 0m;
        invoiceAverage = 0m;

        txtInvoiceNum.Text = "";
        txtInvoiceTotal.Text = "";
        txtInvoiceAverage.Text = "";
        txtSubtotal.Text = "";
}
```

Assign value 0 to global variable

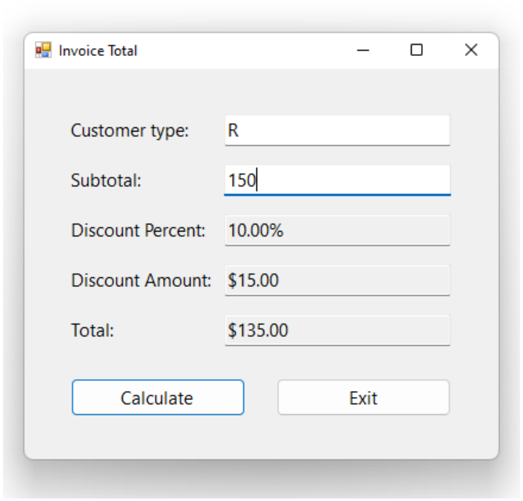
Assign and output empty value

4. Chapter 05: How to code control structures

__Update some form from last chapter and add new form with its codes

4.1. Project 01: Invoice Total

❖ Preview



Customer type: R

Subtotal: 150

Discount Percent: 10.00%

Discount Amount: $15.00

Total: $135.00

❖ Objectives

This project is purposed to add a new field (Customer Type) to Invoice Total form that is designed on chapter 02 and change some condition by following the customer style.

❖ Applications

✓ Business rules (base on customer type)

Customer Type	Subtotal ($)	Discount (%)
R	0 – 100	0
	100 – 250	10
	250 – 500	15
	500 – Unlimited	20
C	0 – 250	20
	250 – Unlimited	30
other	Not Set	40

✓ Codes

In this form use 2 functions: Calculate and Exit.

For function Calculate uses 5 data members:

- cusType (string): get value from TextBox txtCusType
- subtotal (decimal): get value from txtSubtotal and convert to decimal
- discPercent (decimal): default value is 0
- discAmount (decimal): find discount amount by subtotal * discPercent
- invoiceTotal (decimal): find total cost by subtotal – discAmount

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}

private void btnCalculate_Click(object sender, EventArgs e)
{
        string cusType = txtCusType.Text;
        decimal subtotal = Convert.ToDecimal(txtSubtotal.Text);
        decimal discPercent = 0m;

        if (cusType == "R" || cusType == "r")
        {
                if (subtotal > 0 && subtotal < 100) discPercent = 0m;
                else if (subtotal >= 100 && subtotal < 250) discPercent = 0.1m;
                else if (subtotal >= 250 && subtotal < 500) discPercent = 0.15m;
                else if (subtotal > 500) discPercent = 0.2m;
        }
```

If customer type is "R" we will follow this condition

```
        else if (cusType == "C" || cusType == "c")
        {
                if (subtotal > 0 && subtotal < 250) discPercent = 0.2m;
                else discPercent = 0.3m;
        }
```

If customer type is "C" we will follow this condition

```
        else
        {
                discPercent = 0.4m;
        }
```

If customer type is not "R" and "C" we will follow this condition

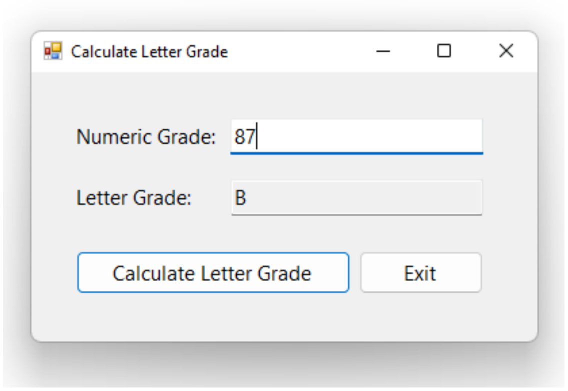
```
        decimal discAmount = subtotal * discPercent;
        decimal invoiceTotal = subtotal - discAmount;
```

Calculate and sign result to variable

```
        txtDiscountPecent.Text = discPercent.ToString("P");
        txtDiscountAmount.Text = discAmount.ToString("C");
        txtTotal.Text = invoiceTotal.ToString("C");

        txtSubtotal.Focus();
}
```

4.2. Project 02: Calculate Letter Grade

❖ Preview



Numeric Grade: 87

Letter Grade: B

❖ Objectives

This form is designed for calculating a grade of number that input in the Number Grade field then output the result in the Letter Grade field as a letter from A-F.

❖ Applications

✓ Business rules

Numeric Grade	**Letter Grade**
90 – 100	A
80 – 89	B
70 – 79	C
60 – 69	D
0 – 59	F

✓ Codes

In this form use 2 functions for Calculate and Exit.

For function Calculate have 2 data members:

- alphanum (decimal): get value from txtNumGrade and convert to decimal
- letterGrade (char): declare for getting value and output

function Exit

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}
```

Close method uses to close application

Event Click

```
private void btnCalculate_Click(object sender, EventArgs e)
{
```

function Calculate

Data members

```
        decimal alphanum = Convert.ToDecimal(txtNumGrade.Text);
        char letterGrade;
```

By following the business rules we get these conditions.

```
        if (alphanum >= 0 && alphanum < 60) letterGrade = 'F';
        else if (alphanum >= 60 && alphanum < 70) letterGrade = 'D';
        else if (alphanum >= 70 && alphanum < 80) letterGrade = 'C';
        else if (alphanum >= 80 && alphanum < 90) letterGrade = 'B';
        else if (alphanum >= 90 && alphanum <=100) letterGrade = 'A';
        else
        {
            MessageBox.Show("Please fill a number from 0 to 100");
            letterGrade = ' ';
        }
```

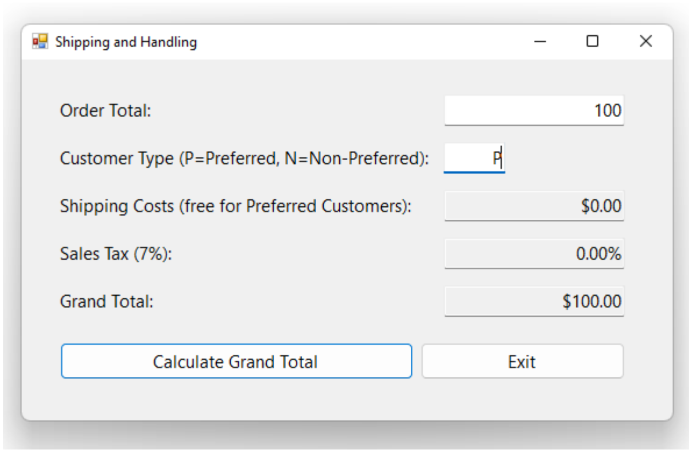
Output result to textbox.

```
        txtLetterGrade.Text = letterGrade.ToString();
        txtNumGrade.Focus();
}
```

We don't allow to enter a number that least than 0 or bigger than 100, so we add this condition to check the value come.

4.3. Project 03: Shipping and Handling

❖ Preview


Shipping and Handling

Order Total:	100
Customer Type (P=Preferred, N=Non-Preferred):	P
Shipping Costs (free for Preferred Customers):	$0.00
Sales Tax (7%):	0.00%
Grand Total:	$100.00

❖ Objectives

In this project we will apply code to the form Shipping and Handling that we have designed in chapter 2 for calculating the Shipping Costs and the Sales Tax that depend on Customer Type and calculating the Grand Total.

❖ Applications

- ✓ Business rules

Customer Type	Order Total ($)	Shipping Cost ($)
N	0 – 25	5
	25 – 500	8
	500 – 1,000	10
	1,000 – 5,000	20
	5,000 – Unlimited	20
P	Not Set	Free

- ✓ Codes

  In this form use 3 functions for Calculate and Exit and Clear.

  For function Calculate have 5 data members:

  - orderTotal (decimal): get value from txtOrderTotal and convert to decimal
  - cusType (string): get value from txtCustomerType
  - shippingCosts (decimal): default value is 0
  - salesTax (decimal): find tax of sales, tax is 7% = orderTotal * 0.07
  - grandTotal (decimal): orderTotal + shippingCosts + salesTax

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}
private void btnCalculate_Click(object sender, EventArgs e)
{
        decimal orderTotal = Convert.ToDecimal(txtOrderTotal.Text);
        decimal shippingCost = 0m;
        string cusType = txtCustomerType.Text;

        if (cusType == "P" || cusType == "p") shippingCost = 0m;
```

If customer type is "P" we will follow this condition

```
        else if (cusType == "N" || cusType == "n")
        {
                if (orderTotal > 5000) shippingCost = 20m;
                else if (orderTotal > 1000) shippingCost = 10m;
                else if (orderTotal > 500) shippingCost = 8m;
                else if (orderTotal > 25) shippingCost = 5m;
                else shippingCost = 0m;
        }
```

If customer type is "N" we will follow this condition

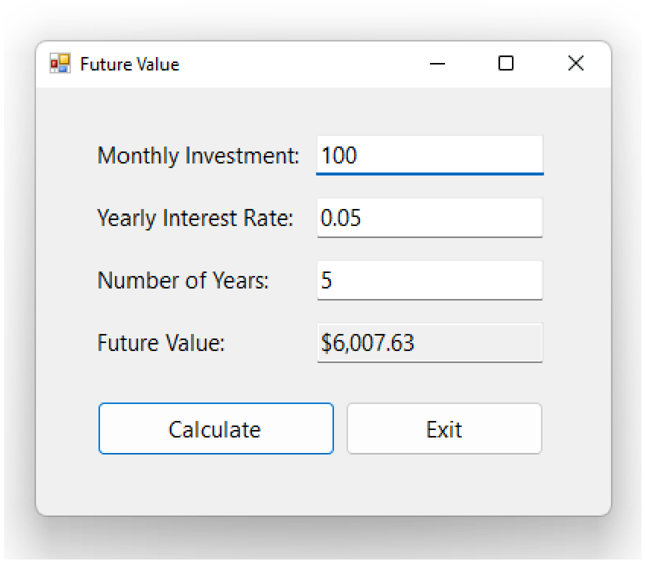
```
        else
        {
                clear();
                orderTotal = 0m;
                MessageBox.Show("Customer Type should be P or N");
        }
```

User must enter "P" or "N" on customer type

```
        decimal salesTax = orderTotal * 0.07m;
        decimal grandTotal = orderTotal + shippingCost + salesTax;

        txtShippingCosts.Text = shippingCost.ToString("C");
        txtSalesTax.Text = salesTax.ToString("C");
        txtGrandTotal.Text = grandTotal.ToString("C");

        txtCustomerType.Focus();
}

void clear()
{
        txtOrderTotal.Text = "";
        txtCustomerType.Text = "";
        txtShippingCosts.Text = "";
        txtSalesTax.Text = "";
        txtGrandTotal.Text = "";
}
```

4.4. Project 04: Future Value

❖ Preview



Monthly Investment: 100
Yearly Interest Rate: 0.05
Number of Years: 5
Future Value: $6,007.63

❖ Objectives

This form is purpose to find value that could have in the future. So, in this project we will design a new form and apply code to follow the purpose above.

❖ Design

➢ Form

Default name	Property	Value
Form1	Text	Future Value
	ActionButton	btnCalculate
	CancelButton	btnExit
	StartPosition	CenterScreen

➢ Labels

Default name	Property	Value
label1	Text	Monthly Investment:
	TextAlign	MiddleLeft
	TabIndex	0
label2	Text	Yearly Interest Rate:
	TextAlign	MiddleLeft
label3	Text	Number of Years:
	TextAlign	MiddleLeft
label4	Text	Future Value:
	TextAlign	MiddleLeft

➢ Textboxes

Default name	Property	Value
textBox1	Name	txtMonthlyInvestment
	TabIndex	1
textBox2	Name	txtYearlyInterestRate
	TabIndex	2
textBox3	Name	txtNumYears
	TabIndex	3

textBox4	Name	txtFutureValue
	ReadOnly	True
	TabStop	False

➢ Buttons

Default name	**Property**	**Value**
button1	Name	btnCalculate
	Text	&Calculate
	TabIndex	4
button2	Name	btnExit
	Text	E&xit
	TabIndex	5

❖ Codes

This is form uses 2 functions: Calculate and Exit.

In function Calculate have 6 data members:

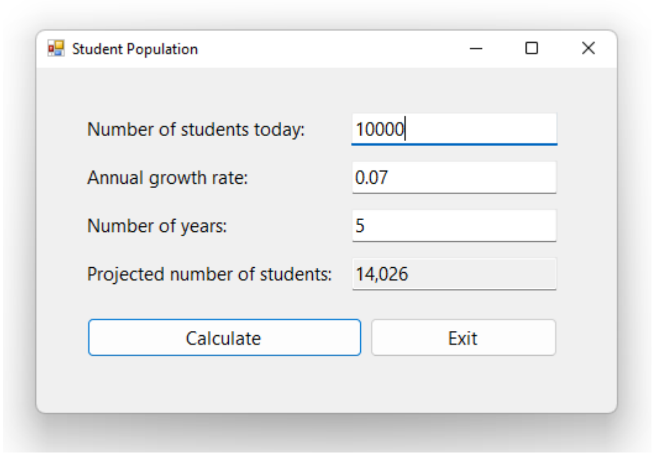
- monthlyInvestment (decimal): get value from txtMonthlyInvestment and convert to decimal
- yearlyInterestRate (decimal): get value from txtYearlyInterestRate and convert to decimal
- years (int): get value from txtNumYears and convert to int32
- months (int): years * 12
- monthlyInvestment (decimal): yearlyInterestRate / 12 /100
- futureValue (decimal): default value is 0

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}

private void btnCalculate_Click(object sender, EventArgs e)
{
        decimal monthlyInvesment = Convert.ToDecimal(txtMonthlyInvestment.Text);
        decimal yearlyInterestRate = Convert.ToDecimal(txtYearlyInterestRate.Text);
        int years = Convert.ToInt32(txtNumYears.Text);
        int months = years * 12;
        decimal monthlyInterestRate = yearlyInterestRate / 12 / 100;
        decimal futureValue = 0m;

        for (int i = 0; i < months; i++)
        {
                futureValue = (futureValue + monthlyInvesment) * (1 + monthlyInterestRate);
        }

        txtFutureValue.Text = futureValue.ToString("C");
        txtMonthlyInvestment.Focus();
}
```

Use for-loop to find future value month by month

4.5. Project 05: Student Population

❖ Preview



Number of students today: 10000
Annual growth rate: 0.07
Number of years: 5
Projected number of students: 14,026

❖ Objectives

In project we don't design the form again and we will take the form that designed on chapter 02 and codes that apply in chapter 03 then use for-loop for this project.

❖ Codes

In this form use 2 functions: Calculate and Exit.

In function Calculate have 6 data members:

- stuNow (double): get value from txtStuNumNow and convert to double
- agr (double): get value from txtAGR and convert to double
- numYear (int): get value form txtNumYear and convert to int32
- tmp (double): default value is 0
- a (double): 1 + agr (use in for-loop)
- stuNumProjected (double): stuNow * tmp

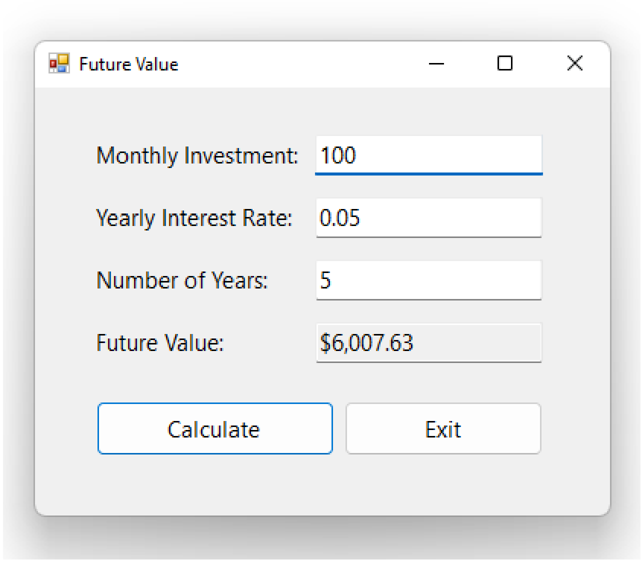
```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}

private void btnCalculate_Click(object sender, EventArgs e)
{
        double stuNow = Convert.ToDouble(txtStuNumNow.Text);
        double agr = Convert.ToDouble(txtAGR.Text);
        int numYear = Convert.ToInt32(txtNumYear.Text);

        double tmp = 1;

        for (int i=0; i<numYear; i++)
        {
                double a = 1 + agr;
                tmp = tmp * a;
        }

        double stuNumProjected = stuNow * tmp;

        txtStuNumProjected.Text = stuNumProjected.ToString("N0");
        txtStuNumNow.Focus();
}
```

5. Chapter 06: How to code methods and event handlers

__Update form Future Value by create new method and add event to textbox

5.1. Project 01: Future Value

❖ Preview



Monthly Investment: 100
Yearly Interest Rate: 0.05
Number of Years: 5
Future Value: $6,007.63

❖ Objectives

This form is purpose to find value that could have in the future. So, in this project we will create a new method to calculate and add event to textbox .

❖ Codes

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}

private void btnCalculate_Click(object sender, EventArgs e)
{
        decimal monthlyInvestment = Convert.ToDecimal(txtMonthlyInvestment.Text);
        decimal yearlyInterestRate = Convert.ToDecimal(txtYearlyInterestRate.Text);
        int years = Convert.ToInt32(txtNumYears.Text);
        int months = years * 12;
        decimal monthlyInterestRate = yearlyInterestRate / 12 / 100;

        decimal futureValue = this.CalculateFutureValue(monthlyInvestment, monthlyInterestRate, months);

        txtFutureValue.Text = futureValue.ToString("C");
        txtMonthlyInvestment.Focus();
}

private decimal CalculateFutureValue(decimal monthlyInvestment, decimal monthlyInterestRate, int months)
{
        decimal futureValue = 0m;
        for (int i = 0; i < months; i++)
        {
                futureValue = (futureValue + monthlyInvestment) * (1 +
                                monthlyInterestRate);
        }

        return futureValue;
}

private void ClearFutureValue(object sender, EventArgs e)
{
        txtFutureValue.Text = "";
}
```

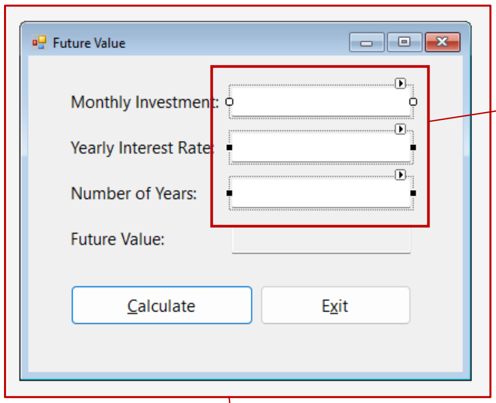
Call function Calculate

Create function Calculate

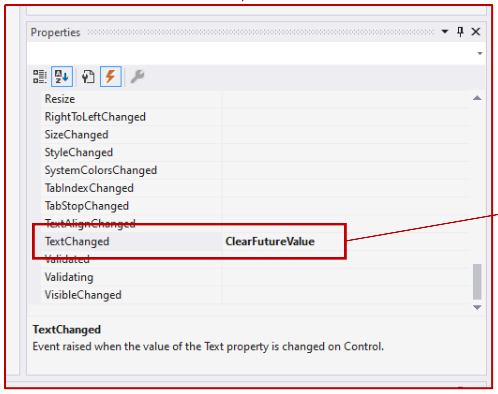
Create function Clear
this function will affect when user
clear some value on textbox.

❖ Note

To add event to textbox go to design then follow these several steps:



Use Ctrl key and select these 3 textboxes



Properties

Resize	
RightToLeftChanged	
SizeChanged	
StyleChanged	
SystemColorsChanged	
TabIndexChanged	
TabStopChanged	
TextAlignChanged	
TextChanged	ClearFutureValue
Validated	
Validating	
VisibleChanged	

TextChanged
Event raised when the value of the Text property is changed on Control.

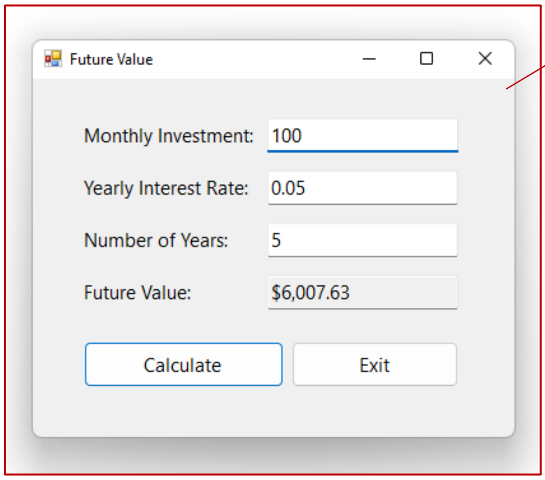
In the property find TextChanged event then select ClearFutureValue

6. Chapter 07: How to handle exceptions and valid data

__Update form Future Value by adding validation

6.1. Project 01: Future Value

❖ Preview



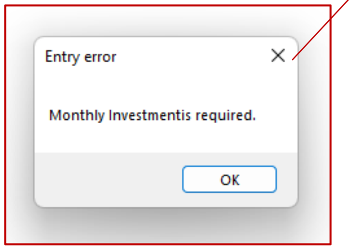
Monthly Investment: 100

Yearly Interest Rate: 0.05

Number of Years: 5

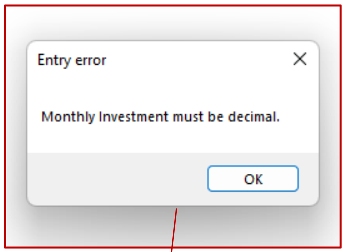
Future Value: $6,007.63

Is valid data

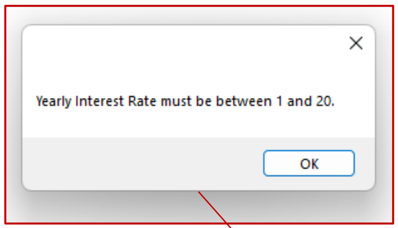

Entry error

Monthly Investmentis required.

It will require user to input when user calculate without input.

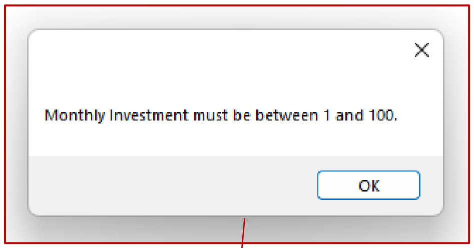

Entry error

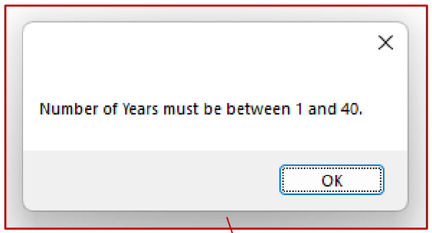
Monthly Investment must be decimal.


Yearly Interest Rate must be between 1 and 20.

If user input any text to each field it will alert message to user.

If user input Yearly Interest Rate least than 1 and greater than 20, it will alert a message to user


Monthly Investment must be between 1 and 100.


Number of Years must be between 1 and 40.

If user input Monthly Investment least than 1 and greater than 100, it will alert a message to user

If user input Number of Year least than 1 and greater than 20, it will alert a message to user

❖ Objectives

This project is purpose to add validation to the form Future Value.

❖ Codes

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}
```

Try block

```
private void btnCalculate_Click(object sender, EventArgs e)
{
        try
        {
                if (isValidData())
                {
                        decimal monthlyInvestment =
                                Convert.ToDecimal(txtMonthlyInvestment.Text);
                        decimal yearlyInterestRate =
                                Convert.ToDecimal(txtYearlyInterestRate.Text);
                        int years = Convert.ToInt32(txtNumYears.Text);
                        int months = years * 12;
                        decimal monthlyInterestRate = yearlyInterestRate / 12 / 100;

                        decimal futureValue = this.CalculateFutureValue(monthlyInvestment,
                                monthlyInterestRate, months);

                        txtFutureValue.Text = futureValue.ToString("C");
                        txtMonthlyInvestment.Focus();
                }
        }
        catch (Exception ex)
        {
                MessageBox.Show(ex.Message + "\n\n" + ex.GetType().ToString() + "\n" +
                        ex.StackTrace, "Exception");
        }
}
```

Catch block

```
private decimal CalculateFutureValue(decimal monthlyInvestment, decimal monthlyInterestRate, int months)
{
        decimal futureValue = 0m;
        for (int i = 0; i < months; i++)
        {
                futureValue = (futureValue + monthlyInvestment) * (1 +
                        monthlyInterestRate);
        }

        return futureValue;
}

private void ClearFutureValue(object sender, EventArgs e)
{
        txtFutureValue.Text = "";
}

public bool isValidData()
{
        return ( isPresent(txtMonthlyInvestment, "Monthly Investment") &&
                isDecimal(txtMonthlyInvestment, "Monthly Investment") &&
                isWithinRange(txtMonthlyInvestment, "Monthly Investment", 1, 100) &&
                isPresent(txtYearlyInterestRate, "Yearly Interest Rate") &&
                isDecimal(txtYearlyInterestRate, "Yearly Interest Rate") &&
                isWithinRange(txtYearlyInterestRate, "Yearly Interest Rate", 1, 20) &&
                isPresent(txtNumYears, "Number of Years") &&
                isDecimal(txtNumYears, "Number of Years") &&
                isWithinRange(txtNumYears, "Number of Years", 1, 40) );
}


public bool isPresent(TextBox textBox, string name)
{
        if (textBox.Text == "")
        {
                MessageBox.Show(name + "is required.", "Entry error");
                return false;
        }
        return true;
}
```

Function Calculate

Function Clear

Use to check the input from user is valid or not.

Call function that's used in valid function

Use to check if textbox is empty or not.

```
public bool isDecimal(TextBox textBox, string name)
```

Use to check the value in textbox is decimal or not.

```
{
        try
        {
                Convert.ToDecimal(textBox.Text);
                return true;
        }
        catch (FormatException)
        {
                MessageBox.Show(name + " must be decimal.", "Entry error");
                textBox.Focus();
                return false;
        }
}

public bool isInt32(TextBox textBox, string name)
```

Use to check the value in textbox is integer or not.

```
{
        try
        {
                Convert.ToInt32(textBox.Text);
                return true;
        }
        catch (FormatException)
        {
                MessageBox.Show(name + " must be integer.", "Entry error");
                textBox.Focus();
                return false;
        }
}
```

Use validate minimum and maximum.

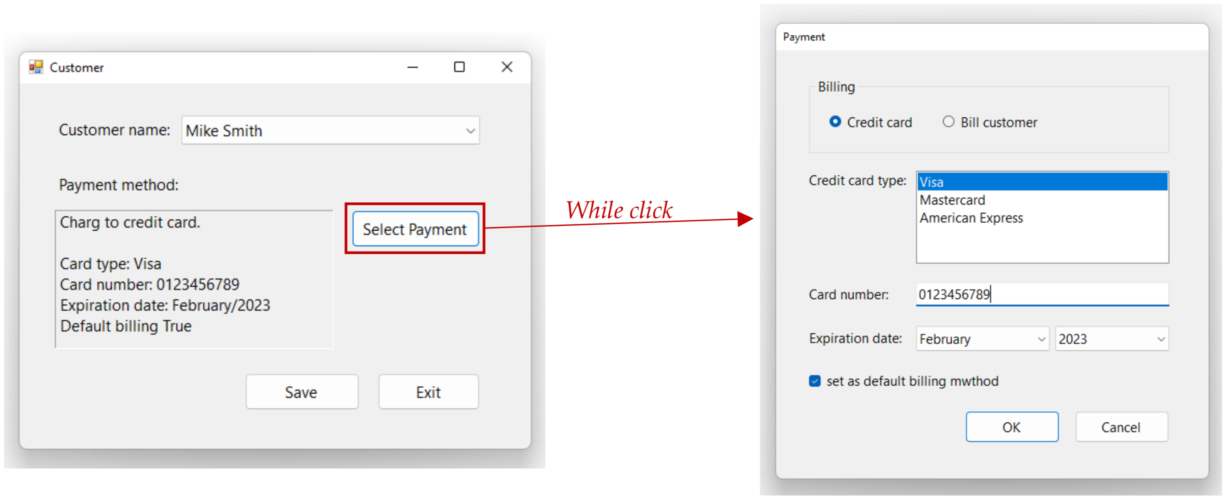
```
public bool isWithinRange(TextBox textBox, string name, decimal min, decimal max)
{
        decimal number = Convert.ToDecimal(textBox.Text);
        if (number < min || number > max)
        {
                MessageBox.Show(name + " must be between " + min + " and " + max + ".");
                return false;
        }
        return true;
}
```

7. Chapter 10: More skills for working Windows forms and controls

__Design a new form using some new controls and apply complex codes

7.1. Project 01: Customer Payment

❖ Preview



Customer name: Mike Smith

Payment method:

Charg to credit card.

Card type: Visa
Card number: 0123456789
Expiration date: February/2023
Default billing True

Billing

● Credit card ○ Bill customer

Credit card type: Visa
Mastercard
American Express

Card number: 0123456789

Expiration date: February 2023

☑ set as default billing mwthod

❖ Objectives

This project is purpose to design a new form using some controls such as ComboBox, GroupBox, RadioButton, ListBox, CheckBox. We will design 2 windows forms Customer and Payment (Customer is the main windows).

❖ Designs

➢ Form (Customer)

Default name	Property	Value
Form1	Text	Customer
	ActionButton	btnSave
	CancelButton	btnExit
	StartPosition	CenterScreen

➢ Labels

Default name	Property	Value
label1	Text	Customer name:
	TextAlign	MiddleLeft
	TabIndex	0
label2	Text	Payment method:
	TextAlign	MiddleLeft
label3	Name	lblPayment
	BorderStyle	Fixed3D
	AutoSize	False
	Text	""

➢ ComboBox

Default name	Property	Value
comboBox1	Name	cboNames:
	DropDownStyle	DropDownList
	TabIndex	1

➢ Buttons

Default name	Property	Value
button1	Name	btnSelectPayment
	Text	Select Payment
	TabIndex	2
button2	Name	btnSave
	Text	Save
	TabIndex	3
button3	Name	btnExit
	Text	Exit
	TabIndex	4

➢ **Form (Payment)**

Default name	Property	Value
Form2	Text	Payment
	ActionButton	btnOK
	CancelButton	btnCancel
	StartPosition	CenterScreen
	ControlBox	False
	MaximizeBox	False
	FormBorderStyle	FixedDialog

➢ **Group Box**

Default name	Property	Value
groupBox1	Text	Billing

➢ **Radio Button**

Default name	Property	Value
radioButton1	Name	rdoCreditCard
	Checked	True
	TabIndex	1
radioButton2	Name	rdoBillCustomer
	TabIndex	2

➢ **List Box**

Default name	Property	Value
listBox1	Name	lstCreditCardType
	TabIndex	3

➢ **Text Box**

Default name	Property	Value
textBox1	Name	txtCardNumber
	TabIndex	4

➢ Combo Box

Default name	Property	Value
comboBox1	Name	cboExpirationMonth
	DropDownStyle	DropDownList
	TabIndex	5
comboBox1	Name	cboExpirationYear
	DropDownStyle	DropDownList
	TabIndex	6

➢ Text Box

Default name	Property	Value
checkBox1	Name	chkDefault
	Checked	True
	TabIndex	7

➢ Buttons

Default name	Property	Value
button1	Name	btnOK
	Text	OK
	TabIndex	8
button2	Name	btnCancel
	Text	Cancel
	TabIndex	9

❖ Codes

✓ Customer form

```
private void btnExit_Click(object sender, EventArgs e)
{
        this.Close();
}

bool isDataSaved = true;

private void frmCustomer_Load(object sender, EventArgs e)
{
        cboNames.Items.Add("Mike Smith");
        cboNames.Items.Add("Nancy Jones");
}

private void DataChanged(object sender, EventArgs e)
{
        isDataSaved = false;
}

private void btnSelectPayment_Click(object sender, EventArgs e)
{
        Form paymentForm = new frmPayment();
        DialogResult selectedBtn = paymentForm.ShowDialog();
        if (selectedBtn == DialogResult.OK)
        {
                lblPayment.Text = (string)paymentForm.Tag;
        }
}

private void btnSave_Click(object sender, EventArgs e)
{
        if (IsValidData())
        {
                SaveData();
        }
}
```

```
private void SaveData()
{
        cboNames.SelectedIndex = -1;
        lblPayment.Text = "";
        isDataSaved = true;
        cboNames.Focus();
}

private bool IsValidData()
{
        if (cboNames.SelectedIndex == -1)
        {
                MessageBox.Show("You must select a customer.", "Entry error");
                cboNames.Focus();
                return false;
        }
        if (lblPayment.Text == "")
        {
                MessageBox.Show("You must enter a payment.", "Entry error");
                return false;
        }
        return true;
}

private void frmCustomer_FormClosing(object sender, FormClosingEventArgs e)
{
        if (isDataSaved == false)
        {
                string message = "This form contains unsaved data. \n\n Do you want to save
                                it?";
                DialogResult button = MessageBox.Show(message, "Customer",
                MessageBoxButtons.YesNoCancel,
                MessageBoxIcon.Warning);

                if (button == DialogResult.Yes)
                {
                        if (IsValidData())
                                this.SaveData();
                        else
                                e.Cancel = true;
                }
                if (button == DialogResult.Cancel)
                {
                        e.Cancel = true;
                }
        }
}
```

✓ Payment form

```
private void frmPayment_Load(object sender, EventArgs e)
{
        lstCreditCardType.Items.Add("Visa");
        lstCreditCardType.Items.Add("Mastercard");
        lstCreditCardType.Items.Add("American Express");
        lstCreditCardType.SelectedIndex = 0;

        string[] months = {
                "Select a month...","January","February","March","April","May","June",
                "July", "August","September","October","November","December"
        };
        foreach (string month in months)
        {
                cboExpirationMonth.Items.Add(month);
        }
        cboExpirationMonth.SelectedIndex = 0;

        int year = DateTime.Today.Year;
        int endYear = year + 8;

        cboExpirationYear.Items.Add("Select a year...");
        while (endYear > year)
        {
                cboExpirationYear.Items.Add(year);
                year++;
        }
        cboExpirationYear.SelectedIndex = 0;
}

private void btnOK_Click(object sender, EventArgs e)
{
        if (IsValidData())
        {
                this.SaveData();
        }
}
```

```
private bool IsValidData()
{
        if (rdoCreditCard.Checked)
        {
                if (lstCreditCardType.SelectedIndex == -1)
                {
                        MessageBox.Show("You must select a credit card type.", "Entry
                                error");
                        lstCreditCardType.Focus();
                        return false;
                }
                if (txtCardNumber.Text == "")
                {
                        MessageBox.Show("You must enter a card number.", "Entry error");
                        txtCardNumber.Focus();
                        return false;
                }
                if (cboExpirationMonth.SelectedIndex == 0)
                {
                        MessageBox.Show("You must select a month.", "Entry error");
                        cboExpirationMonth.Focus();
                        return false;
                }
                if (cboExpirationYear.SelectedIndex == 0)
                {
                        MessageBox.Show("You must select a year.", "Entry error");
                        cboExpirationYear.Focus();
                        return false;
                }
        }
        return true;
}


private void rdoCreditCard_CheckedChanged(object sender, EventArgs e)
{
        if (rdoCreditCard.Checked)
                EnableControls();
        else
                DisableControls();
}
```

```
private void SaveData()
{
        string msg = null;
        if (rdoCreditCard.Checked == true)
        {
                msg += "Charg to credit card. \n";
                msg += "\n";
                msg += "Card type: " + lstCreditCardType.Text + "\n";
                msg += "Card number: " + txtCardNumber.Text + "\n";
                msg += "Expiration date: " + cboExpirationMonth.Text + "/" +
                        cboExpirationYear.Text + "\n";
        }
        else
        {
                msg += "Send bill to customer. \n";
                msg += "\n";
        }

        bool isDefaultBilling = chkDefault.Checked;
        msg += "Default billing " + isDefaultBilling;

        this.Tag = msg;
        this.DialogResult = DialogResult.OK;
}


private void EnableControls()
{
        lstCreditCardType.Enabled = true;
        txtCardNumber.Enabled = true;
        cboExpirationMonth.Enabled = true;
        cboExpirationYear.Enabled = true;
}

private void DisableControls()
{
        lstCreditCardType.Enabled = false;
        txtCardNumber.Enabled = false;
        cboExpirationMonth.Enabled = false;
        cboExpirationYear.Enabled = false;
}
```

Thank You

The End

Lectured by

Kheam Hong