1. Practice exercises for Mid Term Test 2024

Income Tax Calculation

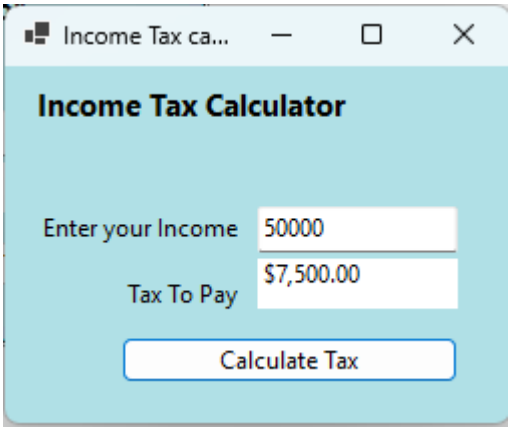
Income tax is payable based on the tax rate applied to a specific range of taxable income, which is called a tax bracket. Following is a fictitious tax rate schedule:

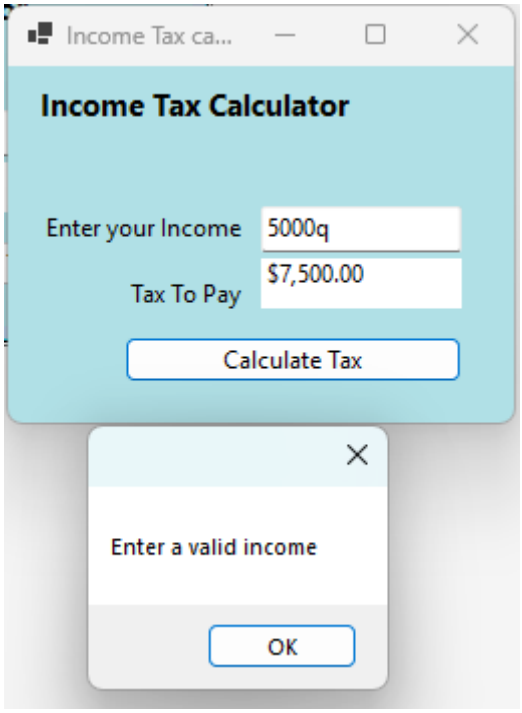
Tax Rate	Tax Bracket
10%	$0 – $10,000
15%	$10001 – $50,000
25%	$50,001 – $100,000
30%	over $100,000

Create an application that lets the user enter his or her taxable income.

The program should then check the following

- Which tax bracket he or she is in,
- Calculate and display the amount of the tax to be paid,
- What is his/her net income after paying tax.
- Also validate that the inputted income is correct format and exists. Create a MessageBox to display errors.





Distance Calculator

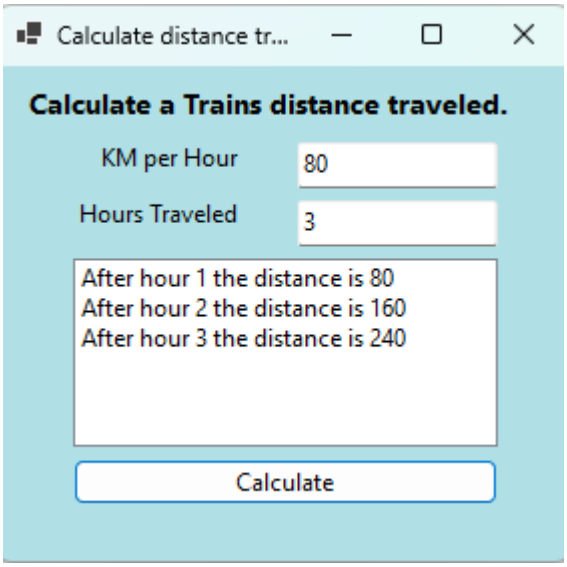
If you know a vehicle's speed and the amount of time it has travelled, you can calculate the distance it has travelled as follows:

$$Distance = Speed \times Time$$

For example, if a train travels 80 Kms per hour for 3 hours, the distance travelled is 240kms.

Create an application where the user enters a vehicle's speed and the number of hours travelled into text boxes.

When the user clicks the *Calculate* button, the application should use a loop to display in a listbox the distance the vehicle has travelled for each hour of that time period.



Calculate interest on deposit

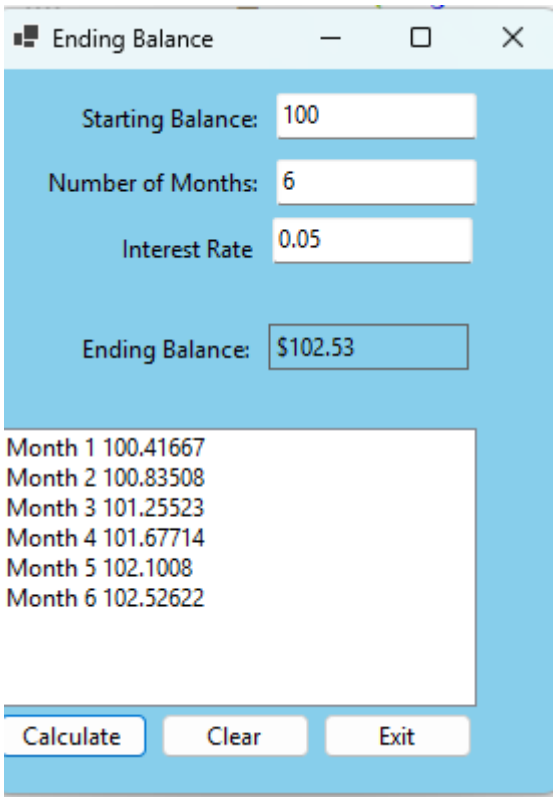
Create a calculator that takes in the following items and returns the amount of interest and principal.

The project includes:

- A Starting Balance of money invested
- The number of months the money is invested
- The interest rate per year (5% = 0.05)
- Interest is added to the Principal each month, so it's a monthly compound.

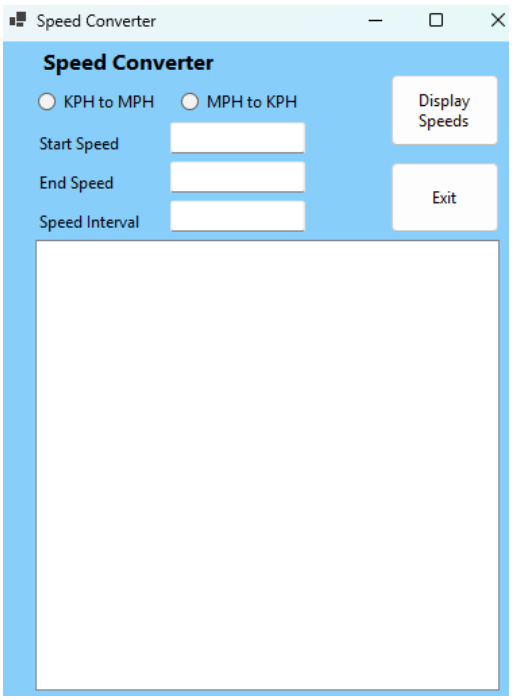
Things you must do:

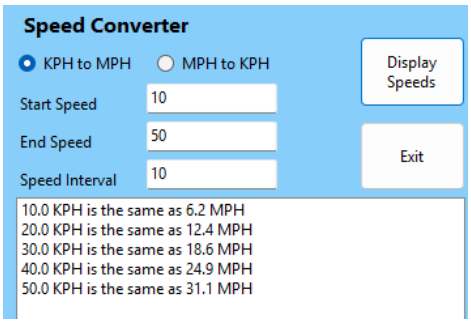
- Create validation for the three input textboxes
- Use a loop to loop around each month until you reach the amount of months
- The Annual interest rate of 5% is divided by 12 to calculate the interest per month. Use that monthly interest to calculate the balance for each month.
- Each month send the data to the ListBox
- Display the Final Balance return in a label.

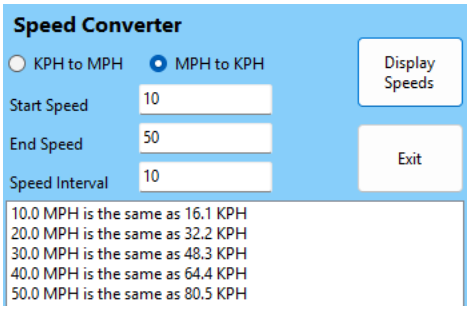


Speed Converter

Create this project from the video.







Code that you will need

```
// Constants
const double KPH_TO_MPH = 0.6214;
const double MPH_TO_KPH = 1.60934;

// Variables
double kph;     // Kilometers per hour
double mph;     // Miles per hour


double startSpeed = 0, endSpeed = 0;
int speedInterval = 0;
```

Income Tax Calculation Answer

```
private void btnCalculateTax_Click(object sender, EventArgs e)
{
    Single income = 0, tax = 0;
    if (Single.TryParse(txtIncome.Text, out income))
    {
        Single.TryParse(txtIncome.Text, out income);
    }
    else
    {
        MessageBox.Show("Enter a valid income ");
    }

    if (income <= 10000)
    {
        tax = income * 0.10f;
    }
    else if (income <= 50000)
    {
        tax = income * 0.15f;
    }
    else if (income <= 100000)
    {
        tax = income * 0.25f;
    }
    else
    {
        tax = income * 0.30f;
    }
    lblOutput.Text = tax.ToString("C");
}
```

Distance Calculator Answer

```
private void btnCalculate_Click(object sender, EventArgs e)
{
    //distance = speed X Time

    Single speed = 0, hoursTotal = 0, hourCount = 1;

    //parse the input and return the data (no validation necessary)
    Single.TryParse(txtKmHour.Text, out speed);
    Single.TryParse(txtHours.Text, out hoursTotal);

    //run the while loop
    while (hourCount <= hoursTotal)
    {
        lbxOutput.Items.Add("After hour " + hourCount + " the distance is "
          + speed * hourCount);
        hourCount++;

    }
}
```

Calculate interest on deposit Answer

```
private void calculateButton_Click(object sender, EventArgs e)
{
    // Local variables
    Single balance;           // The account balance
    Single interestRate;      // the interest rate
    int totalMonths;                // The number of totalMonths
    int count = 1;             // Loop counter, initialized with 1

    // Get the starting balance, all have to be true with &&.
    if (Single.TryParse(txtStartingBal.Text, out balance)
        && int.TryParse(txtMonths.Text, out totalMonths)
        && Single.TryParse(txtinterestRate.Text, out interestRate))
    {
        // Get the number of totalMonths.

        //get the monthy interest rate
        Single monthlyRate = interestRate / 12;

        // The following loop calculates the ending balance.
        while (count <= totalMonths)
        {
            // Add this month's interest to the balance.
            balance *= (1 + monthlyRate);

            //show in the listbox
            lbxOutput.Items.Add("Month " + count + " " + balance);

            // Add one to the loop counter.
            count++;
        }
        // Display the ending balance.
        LblEndingBalance.Text = balance.ToString("c");
    }
    else
    {
        // Invalid starting balance was entered.
        MessageBox.Show("Invalid value for TextBox.");
    }
}
```

Speed Converter Answer

```
private void displayButton_Click(object sender, EventArgs e)
   {
       // Constants
       const double KPH_TO_MPH = 0.6214;
       const double MPH_TO_KPH = 1.60934;

       // Variables
       double kph;     // Kilometers per hour
       double mph;     // Miles per hour
        double startSpeed = 0, endSpeed = 0;
       int speedInterval = 0;

       //input validation
       if ((double.TryParse(txtStartSpeed.Text, out startSpeed)))
       {
           double.TryParse(txtStartSpeed.Text, out startSpeed);
       }
       else
       {
           MessageBox.Show("Invalid data in Start Speed");
       }

       if ((double.TryParse(txtEndSpeed.Text, out endSpeed)))
       {
           double.TryParse(txtEndSpeed.Text, out endSpeed);
       }
       else
       {
           MessageBox.Show("Invalid data in End Speed");
       }
       if ((int.TryParse(txtSpeedInterval.Text, out speedInterval)))
       {
           int.TryParse(txtSpeedInterval.Text, out speedInterval);
       }
       else
       {
           MessageBox.Show("Invalid data in Speed Interval");
       }

       if (rbKphToMph.Checked)
       {
           // Display the table of speeds.
           for (int i = (int)startSpeed; i <= endSpeed; i+= speedInterval)
           {
               // Calculate miles per hour.
               mph = i * KPH_TO_MPH;

               // Display the conversion.
               outputListBox.Items.Add(i.ToString("n1") +
                   " KPH is the same as " + mph.ToString("n1") + " MPH");
           }
        }

       if (rbMphToKph.Checked)
       {
           // Display the speed table.
```

```
        for (int i = (int)startSpeed; i <= endSpeed; i+= speedInterval)
       {
            // Calculate miles per hour.
            kph = i * MPH_TO_KPH;

            // Display the conversion.
            outputListBox.Items.Add(i.ToString("n1") +
                " MPH is the same as " + kph.ToString("n1") + " KPH");
        }
     }
}

private void exitButton_Click(object sender, EventArgs e)
{
    // Close the form.
    this.Close();
}

private void outputListBox_SelectedIndexChanged(object sender, EventArgs e)
{
    outputListBox.Items.Clear();
}
```