|  |
| --- |
| A screenshot of a cell phone  Description automatically generated |
| using System;  using System.Collections.Generic;  using System.ComponentModel;  using System.Data;  using System.Drawing;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows.Forms;  namespace MidtermAssignment  {  public partial class Form1 : Form  {  public Form1()  {  InitializeComponent();  }  private void Form1\_Load(object sender, EventArgs e)  {  }  private void label2\_Click(object sender, EventArgs e)  {  }  private void btnCalculate(object sender, EventArgs e)  {  double years = 0.0;  double growthRate = 0.0;  double population = 0.00; // theres no such thing as a fraction of a person so i keep pop a double at first then type cast it over to int for the display.  double populationChange = 0.0;  int populationOutput = 0;  int populationChangeOutput = 0;  string yearDisplayLabel = "";  string populationDisplayLabel = "";  string populationChangeDisplayLabel = "";  ///////////////////begin data validation for Numbers of Year //////////////////////////  if (txtNumYears.Text.Equals("")) //if you left the years empty  {  DialogResult blankErrorNumYear = MessageBox.Show("Number Of Years has been left blank. " +  "Number of Years is HAS BEEN set to 1.", "Missing Number of Years ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);  if (blankErrorNumYear == DialogResult.OK)  {  years = 1.0;  }  }  else if (txtNumYears.Text.StartsWith("-")) // negative year check  {  DialogResult negativeErrorNumYear = MessageBox.Show("Number Of Years has been given a NEGATIVE NUMBER. " +  "Number of Years is HAS BEEN set to 1.", "Number of Years NEGATIVE NUMBER ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);  if (negativeErrorNumYear == DialogResult.OK)  {  years = 1.0;  }  }  else  {  bool isDouble = Double.TryParse(txtNumYears.Text, out years); // non number input.  if (isDouble)  {  years = Convert.ToDouble(txtNumYears.Text); // years shouldnt be negative.  }  else  {  DialogResult incorrectInputErrorNumYear = MessageBox.Show("Number Of Years has been given an incorrect Input. " +  "You entered " + txtNumYears.Text + " Number of Years is HAS BEEN set to 1.", "Number of Years NEGATIVE NUMBER ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);  if (incorrectInputErrorNumYear == DialogResult.OK)  {  years = 1.0;  }  }  }  //////////////// //begin data validation for Annual Growth Rate ////////////////////////  if (txtAnnualGrowthRate.Text.Equals("")) // empty field  {  DialogResult blankErrorGrowth = MessageBox.Show("Annual Growth Rate has been left blank. " +  "Annual Growth Rate is HAS BEEN set to 0.", "Missing Annual Growth Rate ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);  if (blankErrorGrowth == DialogResult.OK)  {  growthRate = 0.0;  }  }  else if (txtAnnualGrowthRate.Text.StartsWith("-"))  {  DialogResult negativeErrorGrowth = MessageBox.Show("Annual Growth Rate has been given a NEGATIVE NUMBER. " +  "Annual Growth Rate is HAS BEEN set to 0.", "Annual Growth Rate NEGATIVE NUMBER ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);  if (negativeErrorGrowth == DialogResult.OK)  {  growthRate = 0.0;  }  }  else  {  bool isDouble = Double.TryParse(txtAnnualGrowthRate.Text, out growthRate);  if (isDouble)  {  growthRate = Convert.ToDouble(txtAnnualGrowthRate.Text); // growth rate shouldnt be negative otherwise it wouldnt be a growth but rather a shrinkage.  }  else  {  DialogResult incorrectInputErrorGrowth = MessageBox.Show("Annual Growth Rate has been given an incorrect Input." +  " You entered " + txtNumYears.Text + " Number of Years is HAS BEEN set to 0.", "Number of Years NEGATIVE NUMBER ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);  if (incorrectInputErrorGrowth == DialogResult.OK)  {  growthRate = 0.0;  }  }  }  ///////////////// //begin data validation for Population //////////////////  if (txtPopulation.Text.Equals(""))  {  DialogResult blankErrorPopulation = MessageBox.Show("Population has been left blank. Population is HAS BEEN set to 10000.", "Missing Population ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);  if (blankErrorPopulation == DialogResult.OK)  {  population = 10000;  }  }  else if (txtPopulation.Text.StartsWith("-"))  {  DialogResult negativeErrorPopulation = MessageBox.Show("Population has been given a NEGATIVE NUMBER. Population is HAS BEEN set to 10000.", "Population NEGATIVE NUMBER ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);  if (negativeErrorPopulation == DialogResult.OK)  {  growthRate = 0.0;  }  }  else  {  bool isInt = Double.TryParse(txtPopulation.Text, out population);  if (isInt)  {  population = Convert.ToInt32(txtPopulation.Text); // population shouldnt be negative.  }  else  {  DialogResult incorrectInputErrorPopulation = MessageBox.Show("Population has been given an incorrect Input. You entered " + txtNumYears.Text + " Population HAS BEEN set to 10000.", "Population NEGATIVE NUMBER ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);  if (incorrectInputErrorPopulation == DialogResult.OK)  {  population = 10000;  }  }  }  for (int i = 0; i < years + 1 ; i++)  {  if (i == 0)  {  populationOutput = (int)population;  populationChangeOutput = (int)populationChange;    yearDisplayLabel += i.ToString() + "\n";  populationChangeDisplayLabel += populationChangeOutput.ToString() + "\n";  populationDisplayLabel += populationOutput.ToString() + "\n";  }  else  {  populationChange = population \* growthRate;  population += populationChange;  populationOutput = (int)population;  populationChangeOutput = (int)populationChange;    yearDisplayLabel += i.ToString() + "\n";  populationChangeDisplayLabel += populationChangeOutput.ToString() + "\n";  populationDisplayLabel += populationOutput.ToString() + "\n";  }  }    lblYearOutput.Text = yearDisplayLabel;  lblYearOutput.Visible = true;  lblPopulationOutput.Text = populationDisplayLabel;  lblPopulationOutput.Visible = true;  lblPopulationChangeOutput.Text = populationChangeDisplayLabel;  lblPopulationChangeOutput.Visible = true;  }  private void btnClear\_Click\_1(object sender, EventArgs e)  {  lblYearOutput.Visible = false;  lblPopulationOutput.Visible = false;  lblPopulationChangeOutput.Visible = false;  txtAnnualGrowthRate.Clear();  txtNumYears.Clear();  txtPopulation.Clear();  }  private void btnExit\_Click\_1(object sender, EventArgs e)  {  this.Close();  }  private void btnClearResults\_Click(object sender, EventArgs e)  {  lblYearOutput.Visible = false;  lblPopulationOutput.Visible = false;  lblPopulationChangeOutput.Visible = false;  }  }  } |