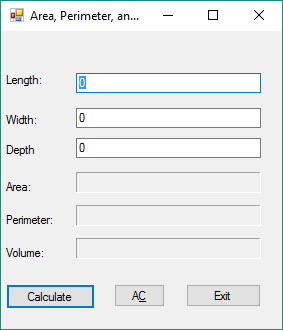
Project 4 documentation:

This version of Project 4 calculates Area, Perimeter, and Volume.

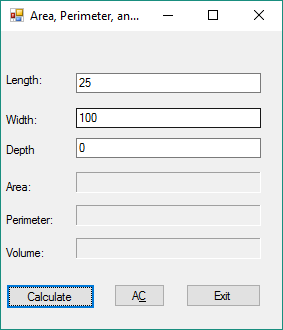
When you first start running the program:

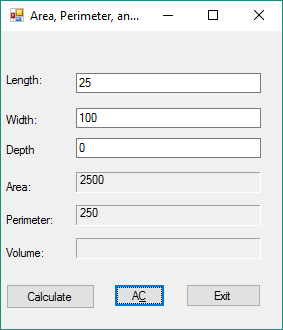
Focus starts in Length.

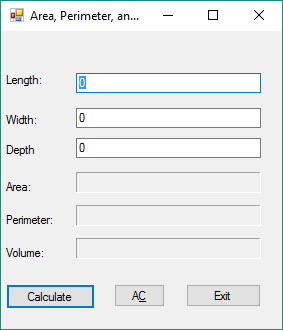
Tab order: Length, Width, Depth, Calculate, AC (all Clear), Exit. Enter defaults to Calculate.

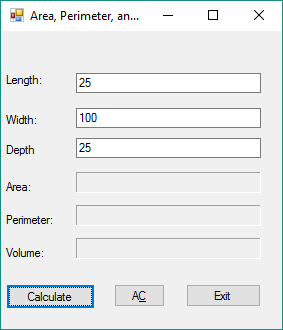
AC can also be activated by typing C (thought it was supposed to be Alt+C but C works and Alt+C doesn’t. Why?

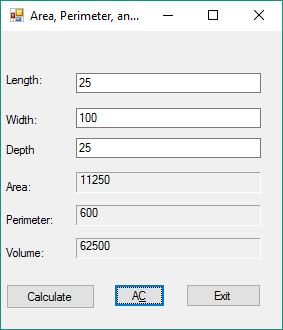
ESC activates Exit.

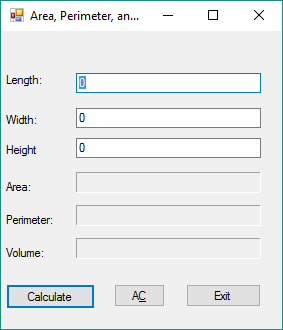
before calculating first value

After calculating Area and Perimeter for first value set

After clicking AC

Before calculating Area, Perimeter, and Volume

After calculating Area, Perimeter, and Volume of 3d object. I was trying to think of why you’d need the surface area of a 3d object, but it’s hard to think of where this would normally come up. The base, calculated using L\*W would normally be more useful. However, this would necessitate changing “Depth” to “Height” to make it clear which measurements were which. Probably a good idea anyway (program has been updated to include this change).

 Also changed internal variable names to match.

Plans for future improvements:

* Add data validation
  + Is it a number?
  + Is it a positive number?
* Add exception handling
  + What do you do if input fails data validation?
  + Other types of exceptions that should be looked for?

Code for Form1.cs:

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 WindowsFormsApplication1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void btnCalculate\_Click(object sender, EventArgs e)

{

calAnswer(); //calculate results

}

private void Form1\_Load(object sender, EventArgs e)

{

populate();

}

/\*the populate function sets text boxes to default setting 0.

\* Could have used txtLength.Clear() instead\*/

private void populate()

{

txtLength.Text = "0";

txtWidth.Text = "0";

txtDepth.Text = "0";

txtArea.Text = "";

txtPerimeter.Text = "";

txtVolume.Text = "";

txtLength.Focus();

}

/\* calAnswer is the function that processes input.

\* This function is where I would do data validation and exception handling

\*/

private void calAnswer()

{

decimal length = Convert.ToDecimal(txtLength.Text); //change Length to decimal

decimal width = Convert.ToDecimal(txtWidth.Text); //change width to decimal

decimal depth = Convert.ToDecimal(txtDepth.Text); //change depth to decimal

decimal area = length \* width; //calculate area

decimal perimeter = 2 \* (length + width); //calculate perimeter

if (depth !=0){ //check if a value for depth has been entered (ie, object is 3d)

decimal volume = area \* depth; //calculate volume

txtVolume.Text = Convert.ToString(volume); //display volume

perimeter = (4 \* length) + (4 \* width) + (4 \* depth); //calculate perimeter of 3d object

area = 2 \* ((length \* width) + (length \* depth) + (width \* depth));// calculate area of 3d object

}

txtArea.Text = Convert.ToString(area); //display area

txtPerimeter.Text = Convert.ToString(perimeter); //display perimeter

btnAC.Focus();

} private void btnExit\_Click(object sender, EventArgs e) //Exit program (also works using ESC)

{

this.Close();

}

private void btnAC\_Click(object sender, EventArgs e) //AC=All Clear. Clears all fields for next set of values

{

populate();

}

private void label3\_Click(object sender, EventArgs e)

{

}

private void label6\_Click(object sender, EventArgs e)

{

}

private void label4\_Click(object sender, EventArgs e)

{

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

}

}