```
// Program 4
// CIS 199-02
// Due: 4/25/2017
// Grading ID: B3049

// The GroundPackage class contains properties that are used to hold information that can
be used in the form application.


using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Program4
{
    public class GroundPackage
    {
        public const int MINIMUM_ZIP = 00000; // Minimum zip code
        public const int MAXIMUM_ZIP = 99999; // Maximum zip code
        public const int MINIMUM_LENGTH = 0; // Minimum Length
        public const int MINIMUM_WIDTH = 0; // Minimum Width
        public const int MINIMUM_HEIGHT = 0; // Minimum Height
        public const int MINIMUM_WEIGHT = 0; // Minimum Weight

        private int originZip; // Origin Zip code
        private int destinationZip; // Destination Zip Code
        private double lengthInches; // Length of package (inches)
        private double widthInches; // Width of package (inches)
        private double heightInches; // Height of package (inches)
        private double weightPounds; // Weight of package (pounds)
        private double cost; // Cost of packaging (dollars)

        // Preconditions: None
        // Postconditions: GroundPackage has information stored
        public GroundPackage(int zipOrigin, int zipDestination, double lengthIn, double
widthIn, double heightIn, double weightLb)
        {
            OriginZip = zipOrigin;
            DestinationZip = zipDestination;
            Length = lengthIn;
            Width = widthIn;
            Height = heightIn;
            Weight = weightLb;
        }

        // Preconditions: Zip must be within the required zip range
        // Postconditions: origin zip is returned
        public int OriginZip
        {
            get
            {
                return originZip;
            }
            set
            {
                if ((value > MINIMUM_ZIP) && (value < MAXIMUM_ZIP))
                    originZip = value;
            }
        }
        // Preconditions: Zip must be within the required zip range
        // Postconditions: Destination zip is returned
        public int DestinationZip
        {
            get
            {
                return destinationZip;
            }
            set
            {
                if ((value > MINIMUM_ZIP) && (value < MAXIMUM_ZIP))
                    destinationZip = value;
            }
        }

        // Preconditions: Must be greater than minimum length
        // Postconditions: Length is returned

        public double Length
        {
            get
            {
                return lengthInches;
            }
            set
            {
                if (value > MINIMUM_LENGTH)
                    lengthInches = value;
            }
        }

        // Preconditions: Must be greater than minimum width
        // Postconditions: Width is returned

        public double Width
        {
            get
            {
                return widthInches;
            }
            set
            {
                if (value > MINIMUM_WIDTH)
                    widthInches = value;
            }
        }

        // Preconditions: Height must be greater than minimum height
        // Postconditions: Height is returned

        public double Height
        {
            get
            {
                return heightInches;
```

```
            }
            set
            {
                if (value > MINIMUM_HEIGHT)
                    heightInches = value;
            }
        }

        // Preconditions: Weight must be greater than minimum
        // Postconditions: Weight is returned
        public double Weight
        {
            get
            {
                return weightPounds;
            }
            set
            {
                if (value > MINIMUM_WEIGHT)
                    weightPounds = value;
            }
        }

        // Preconditions: None
        // Postconditions: Returns the calculation of zonedistance

        public int ZoneDistance
        {
            get
            {
                return (originZip / 10000) - (destinationZip / 10000);
            }
        }

        // Preconditions: Varaibles must be declared and return valid values
        // Postcondition: Cost is calculated and returned
        public double CalcCost()
        {
            return cost = .20 * (Length + Width + Height) + .5 * (ZoneDistance + 1) * 
(Weight);
        }

        // Preconditions: None
        // Postcondition: Returns a string of information of the desired package
        public override string ToString()
        {
            return "Origin zip = " + originZip.ToString() + System.Environment.NewLine +
                "Destination Zip = " + destinationZip.ToString() + 
System.Environment.NewLine +
                "Length = " + lengthInches.ToString() + " inches" + 
System.Environment.NewLine +
                "Width = " + widthInches.ToString() + " inches" + 
System.Environment.NewLine +
                "Height = " + heightInches.ToString() + " inches" + 
System.Environment.NewLine +
                "Weight = " + weightPounds.ToString() + " pounds" + 
System.Environment.NewLine +
                "Cost = " + cost.ToString("c");
        }
    }
}
```

```
// Program 4
// CIS 199-02
// Due: 4/25/2017
// Grading ID: B3049

// This program uses the information that is input into each text box and adds the
information into a list box
// which contains all of the inputs, as well as the packaging cost. The user is also able
to use the send to and from
// UofL buttons to change the origin and destination zip of the selected package order.




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 Program4
{
    public partial class Form1 : Form
    {
        List<GroundPackage> orderList = new List<GroundPackage>();

        const int uOfLZip = 40292; // UofL Zip Code

        public Form1()
        {
            InitializeComponent();
        }

        private void addListButton_Click(object sender, EventArgs e)
        {
            int originZip; //Origin zip input variable
            int destinationZip;// Destination zip input variable
            double length; // Length input variable
            double width; // Width input variable
            double height; // Height input variable
            double weight; // Weight input variable
            double cost; // Cost input variable

            // If all variables can be parsed, the cost of packaging is calculated and is
added to the information in the orderlistbox
            if (int.TryParse(originZipInput.Text, out originZip))
            {
                if (int.TryParse(destinationZipInput.Text, out destinationZip))
                {
                    if (double.TryParse(lengthInput.Text, out length))
                    {
                        if (double.TryParse(widthInput.Text, out width))
                        {
                            if (double.TryParse(heightInput.Text, out height))
                            {
                                if (double.TryParse(weightInput.Text, out weight))
                                {
                                    GroundPackage package = new GroundPackage(originZip,
destinationZip, length, width, height, weight);
                                    cost = package.CalcCost();
                                    orderList.Add(package);
                                    orderListBox.Items.Add(cost.ToString("c"));
                                }
                                else // If weight cannot be parsed, show error message
                                {
                                    MessageBox.Show("Enter Weight");
                                }
                            }
                            else // If height cannot be parsed, show error message
                            {
                                MessageBox.Show("Enter Height");
                            }
                        }
                        else // If width cannot be parsed, show error message
                        {
                            MessageBox.Show("Enter Width");
                        }
                    }
                    else // If length cannot be parsed, show error message
                    {
                        MessageBox.Show("Enter Length");
                    }
                }
                else // If destination zip cannot be parsed, show error message
                {
                    MessageBox.Show("Enter Destination Zip");
                }
            }
            else // If origin zip cannot be parsed, show error message
            {
                MessageBox.Show("Enter Origin Zip");
            }
        }

        // Precondition: Item must be selected in listbox
        // Postcondition: Details of the packaging are shown when the details button is
pressed
        private void detailsButton_Click(object sender, EventArgs e)
        {
            int index = orderListBox.SelectedIndex;

            if (index <= -1) // If item is not selected, therefore index = -1, show error
message
            {
                MessageBox.Show("Select a package order");
            }
            else  // If item selected, details shown in orderlist
            {
                MessageBox.Show(orderList[index].ToString());
            }
        }

        // Precondition: Item must be selected on the listbox
        // Postcondition: The destination zip is set to the University of Louisville's
zip code, which is 40292 and update cost
        private void sendToUofLButton_Click(object sender, EventArgs e)
        {
            int index = orderListBox.SelectedIndex;

            if (index <= -1) //If item is not selected, therefore index = -1, show error
message
            {
                MessageBox.Show("Select a package order");
            }
            else // If item is selected, change destinationZip to the uOfLZip and update
cost
            {
                orderList[index].DestinationZip = uOfLZip;
                orderListBox.Items[index] = orderList[index].CalcCost().ToString("c");
                MessageBox.Show("Package has been reset");

            }
        }

        // Precondition: Item must be selected on the listbox
        // Postcondition: The origin zip is set to the University of Louisvile's zip
code, which is 40292
        private void sendFromUofLButton_Click(object sender, EventArgs e)
        {
            int index = orderListBox.SelectedIndex;

            if (index <= -1)// If item is not selected, therefore index = -1, show error
message
            {
                MessageBox.Show("Select a package order");
            }
            else // If item is selected, update origin zip to uOfLZip and update cost
            {
                orderList[index].OriginZip = uOfLZip;
                orderListBox.Items[index] = orderList[index].CalcCost().ToString("c");
                MessageBox.Show("Package has been reset");
            }
        }

        // Preconditions: None
        // Postconditions: Clears listbox
        private void clearButton_Click(object sender, EventArgs e)
        {
            orderListBox.Items.Clear();
        }

        // Preconditions: None
        // Postconditions: Clears input boxes
        private void button1_Click(object sender, EventArgs e)
        {
            originZipInput.Clear();
            destinationZipInput.Clear();
            lengthInput.Clear();
            widthInput.Clear();
            heightInput.Clear();
            weightInput.Clear();
        }
    }
}
```