```
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;

using System.Diagnostics;


namespace Project2

{

    public partial class Form1 : Form

    {

        Database myDB = new Database();


        // Inventory Amounts

        decimal strawberries = 0;

        decimal strawberrieslin = 0;

        decimal bananas = 0;

        decimal bananaslin = 0;

        decimal honey = 0;

        decimal honeylin = 0;

        decimal milk = 0;

        decimal milklin = 0;

        int largecups = 0;

        int largecupslin = 0;

        int smallcups = 0;
int smallcupslin = 0;
// Finances
decimal expenses = 0;
decimal sales = 0;
decimal saleslin = 0;
const decimal STRAWBERRY_SIZE_SMALL = 4;
const decimal STRAWBERRY_SIZE_LARGE = 6;
const decimal STRAWBERRY_PURCHASE_PRICE = 8;
const decimal STRAWBERRY_PURCHASE_QTY = 32;
const decimal BANANA_SIZE_SMALL = 4;
const decimal BANANA_SIZE_LARGE = 6;
const decimal BANANA_PURCHASE_PRICE = 0.7m;
const decimal BANANA_PURCHASE_QTY = 16;
const decimal HONEY_SIZE_SMALL = 1;
const decimal HONEY_SIZE_LARGE = 2;
const decimal HONEY_PURCHASE_PRICE = 9;
const decimal HONEY_PURCHASE_QTY = 16;
const decimal MILK_SIZE_SMALL = 6;
const decimal MILK_SIZE_LARGE = 12;
const decimal MILK_PURCHASE_PRICE = 3;
const decimal MILK_PURCHASE_QTY = 128;
const decimal SMALL_CUP_PURCHASE_PRICE = 12;
const int SMALL_CUP_PURCHASE_QTY = 250;
const decimal LARGE_CUP_PURCHASE_PRICE = 15;
const int LARGE_CUP_PURCHASE_QTY = 250;
const int SMALLSALE = 4;
const int LARGESALE = 5;

public Form1()
```

```
{
    InitializeComponent();
    chkStrawberries.Checked = true;
    rdoSmallCup.Checked = true;
    btnPlaceOrder.Enabled = false;
    strawberries = myDB.StrawUp();
    bananas = myDB.BanUp();
    honey = myDB.honeyUp();
    milk = myDB.milkUp();
    smallcups = myDB.smallcupsUp();
    largecups = myDB.largecupsUp();
    sales = myDB.salesUp();
    expenses = myDB.expensesUp();
    lblStrawberries.Text = strawberries + " oz";
    lblBananas.Text = bananas + " oz";
    lblHoney.Text = honey + " oz";
    lblMilk.Text = milk + " oz";
    lblSmallCups.Text = smallcups + " oz";
    lblLargeCups.Text = largecups + " oz";
    lblSales.Text = sales.ToString("c");
    lblExpenses.Text = expenses.ToString("c");
    lblProfit.Text = (sales - expenses).ToString("c");
    myDB.DisplayTransaction(OldTran);
}

private void btnStrawberries_Click(object sender, EventArgs e)
{
    strawberries += STRAWBERRY_PURCHASE_QTY;
    expenses += STRAWBERRY_PURCHASE_PRICE;
    lblStrawberries.Text = strawberries + " oz";
    lblExpenses.Text = expenses.ToString("c");
    lblProfit.Text = (sales - expenses).ToString("c");
    myDB.UpdateInv(strawberries, bananas, honey, milk, smallcups, largecups);
    myDB.Updateprofit(sales, expenses);
    myDB.DisplayTransaction(OldTran);
}

private void btnBananas_Click(object sender, EventArgs e)
{
    bananas += BANANA_PURCHASE_QTY;
    expenses += BANANA_PURCHASE_PRICE;
    lblBananas.Text =  bananas + " oz";
    lblExpenses.Text = expenses.ToString("c");
    lblProfit.Text = (sales- - expenses).ToString("c");
    myDB.UpdateInv(strawberries, bananas, honey, milk, smallcups, largecups);
    myDB.Updateprofit(sales, expenses);
}

private void btnHoney_Click(object sender, EventArgs e)
{
    honey += HONEY_PURCHASE_QTY;
    expenses += HONEY_PURCHASE_PRICE;
    lblHoney.Text =  honey + " oz";
    lblExpenses.Text = expenses.ToString("c");
    lblProfit.Text = (sales - expenses).ToString("c");
    myDB.UpdateInv(strawberries, bananas, honey, milk, smallcups, largecups);
    myDB.Updateprofit(sales, expenses);
}
```

```
private void btnMilk_Click(object sender, EventArgs e)
{
    milk += MILK_PURCHASE_QTY;
    expenses += MILK_PURCHASE_PRICE;
    lblMilk.Text =  milk + " oz";
    lblExpenses.Text = expenses.ToString("c");
    lblProfit.Text = (sales - expenses).ToString("c");
    myDB.UpdateInv(strawberries, bananas, honey, milk, smallcups, largecups);
    myDB.Updateprofit(sales, expenses);
}

private void btnSmallCups_Click(object sender, EventArgs e)
{
    smallcups += SMALL_CUP_PURCHASE_QTY;
    expenses += SMALL_CUP_PURCHASE_PRICE;
    lblSmallCups.Text =   smallcups.ToString();
    lblExpenses.Text = expenses.ToString("c");
    lblProfit.Text = (sales - expenses).ToString("c");
    myDB.UpdateInv(strawberries, bananas, honey, milk, smallcups, largecups);
    myDB.Updateprofit(sales, expenses);
}

private void btnLargeCups_Click(object sender, EventArgs e)
{
    largecups += LARGE_CUP_PURCHASE_QTY;
    expenses += LARGE_CUP_PURCHASE_PRICE;
    lblLargeCups.Text =  largecups.ToString ();
    lblExpenses.Text = expenses.ToString("c");
```

```
            lblProfit.Text = (sales - expenses).ToString("c");
            myDB.UpdateInv(strawberries, bananas, honey, milk, smallcups, largecups);
            myDB.Updateprofit(sales, expenses);
        }

        private void chkStrawberries_CheckedChanged(object sender, EventArgs e)
        {
            Debug.WriteLine("\n****************************\nStart Inventory Check Debugging\n
    ****************************\n");
            // Determine if there is sufficent inventory.
            // Variable used for inventory checks.
            decimal fruitRequiredban = 0;
            decimal fruitRequiredstr = 0;
            decimal honeyRequired = 0;
            decimal milkRequired = 0;
            // Check size
            if (rdoSmallCup.Checked)
            {
                // Checking fruit
                if (chkStrawberries.Checked)
                {
                    fruitRequiredstr = STRAWBERRY_SIZE_SMALL * nudQuantity.Value;
                }
                if (chkBananas.Checked)
                {
                    fruitRequiredban = BANANA_SIZE_SMALL * nudQuantity.Value;
                }
                // Checking Honey
                if (chkHoney.Checked)
                {
                    honeyRequired = HONEY_SIZE_SMALL * (fruitRequiredstr / STRAWBERRY_SIZE_SMALL + fruitRequiredban / BANANA_SIZE_SMALL);
                }
                // Setting Milk
                milkRequired = MILK_SIZE_SMALL * (fruitRequiredstr / STRAWBERRY_SIZE_SMALL + fruitRequiredban / BANANA_SIZE_SMALL);
            }
            else
            {
                // Checking fruit
                if (chkStrawberries.Checked)
                {
                    fruitRequiredstr = STRAWBERRY_SIZE_LARGE * nudQuantity.Value;
                }
                if (chkBananas.Checked)
                {
                    fruitRequiredban = BANANA_SIZE_LARGE * nudQuantity.Value;
                }
                // Checking Honey
                if (chkHoney.Checked)
                {
                    honeyRequired = HONEY_SIZE_LARGE * (fruitRequiredstr / STRAWBERRY_SIZE_LARGE + fruitRequiredban / BANANA_SIZE_LARGE);
                }
                // Setting Milk
                milkRequired = MILK_SIZE_LARGE * (fruitRequiredstr / STRAWBERRY_SIZE_LARGE + fruitRequiredban / BANANA_SIZE_LARGE);
            }
```

```
    Debug.WriteLineIf(chkStrawberries.Checked, "strawberries inv = " + strawberries + " || strawberries need = " + fruitRequiredstr + "\n");
    Debug.WriteLineIf(chkBananas.Checked, "banana inv = " + bananas + " ||banana need = " + fruitRequiredban + "\n");
    Debug.WriteLine("honey inv = " + honey + " || honey need = " + honeyRequired + "\n");
    Debug.WriteLine("milk inv = " + milk + " || milk need = " + milkRequired + "\n");
    Debug.WriteLineIf(rdoSmallCup.Checked, "smallcups inv = " + smallcups + " || smallcups need = " + nudQuantity.Value + "\n");
    Debug.WriteLineIf(rdoLargeCup.Checked, "largecups inv = " + largecups + "|| largecups need = " + nudQuantity.Value + "\n");
    Debug.WriteLine("\n");
    // Enable Place Order
    if (nudQuantity.Value > 0)
    {
       btnPlaceOrder.Enabled = true;
    }
    // Checking for cups
    if (rdoSmallCup.Checked && smallcups < (fruitRequiredstr / STRAWBERRY_SIZE_SMALL + fruitRequiredban / BANANA_SIZE_SMALL))
    {
       btnPlaceOrder.Enabled = false;
       Debug.WriteLine("Insufficent small cups\n");
       return;
    }
    else if (rdoLargeCup.Checked && largecups < (fruitRequiredstr / STRAWBERRY_SIZE_LARGE + fruitRequiredban / BANANA_SIZE_LARGE))
    {
       btnPlaceOrder.Enabled = false;
       Debug.WriteLine("Insufficent large cups\n");
       return;
    }
```

```
// Checking fruit
if (chkStrawberries.Checked && strawberries < fruitRequiredstr)
{
    btnPlaceOrder.Enabled = false;
    Debug.WriteLine("Insufficent strawberries\n");
    return;
}
if (chkBananas.Checked && bananas < fruitRequiredban)
{
    btnPlaceOrder.Enabled = false;
    Debug.WriteLine("Insufficent bananas\n");
    return;
}
// Checking honey
if (chkHoney.Checked && honey < honeyRequired)
{
    btnPlaceOrder.Enabled = false;
    Debug.WriteLine("Insufficent honey\n");
    return;
}
// Checking milk
if (milk < milkRequired)
{
    btnPlaceOrder.Enabled = false;
    Debug.WriteLine("Insufficent milk\n");
    return;
}
Debug.WriteLine("\n****************************\nEnd Inventory Check Debugging\n****************************\n");
```

```
}

private void btnPlaceOrder_Click(object sender, EventArgs e)
{
    decimal sumnub = 0;
    string Go = Convert.ToString(nudQuantity.Value) + " * ";
    Detail.Items.Clear();
    // Check size
    if (rdoSmallCup.Checked)
    {

        // Checking fruit
        Go += "Small ";
        if (chkStrawberries.Checked)
        {
            sumnub += nudQuantity.Value;
            strawberries -= STRAWBERRY_SIZE_SMALL * nudQuantity.Value;
            strawberrieslin += STRAWBERRY_SIZE_SMALL * nudQuantity.Value;
            Go += "Strawberry ";
        }
        if (chkBananas.Checked)
        {
            sumnub += nudQuantity.Value;
            bananas -= (BANANA_SIZE_SMALL * nudQuantity.Value);
            bananaslin += (BANANA_SIZE_SMALL * nudQuantity.Value);
            Go += "Banana ";
        }
        // Checking Honey
        if (chkHoney.Checked)
    {
        honey -= (HONEY_SIZE_SMALL * sumnub);
        honeylin += (HONEY_SIZE_SMALL * sumnub);
        Go += "with Honey";
    }
    // Setting Milk
    milk -= (MILK_SIZE_SMALL * sumnub);
    milklin += (MILK_SIZE_SMALL * sumnub);
    // Cups
    smallcups -= Convert.ToInt32(sumnub);
    smallcupslin += Convert.ToInt32(sumnub);
    // Sales
    sales += (SMALLSALE * sumnub);
    saleslin += (SMALLSALE * sumnub);
}
else
{
    Go += "Large ";
    // Checking fruit
    if (chkStrawberries.Checked)
    {
        sumnub += nudQuantity.Value;
        strawberries -= STRAWBERRY_SIZE_LARGE * nudQuantity.Value;
        Go += "Strawberry ";
        strawberrieslin += STRAWBERRY_SIZE_LARGE * nudQuantity.Value;
    }
    if (chkBananas.Checked)
    {
        sumnub += nudQuantity.Value;
        bananas -= BANANA_SIZE_LARGE * nudQuantity.Value;
        Go += "Banana ";
        bananaslin += BANANA_SIZE_LARGE * nudQuantity.Value;
    }
    // Checking Honey
    if (chkHoney.Checked)
    {
        honey -= HONEY_SIZE_LARGE * sumnub;
        Go += "with Honey";
        honeylin += HONEY_SIZE_LARGE * sumnub;
    }
    // Setting Milk
    milk -= MILK_SIZE_LARGE * sumnub;
    milklin += MILK_SIZE_LARGE * sumnub;
    // Cups
    largecups -= Convert.ToInt32(sumnub);
    largecupslin += Convert.ToInt32(sumnub);
    // Sales
    sales += LARGESALE * sumnub;
    saleslin += LARGESALE * sumnub;
}
listBox.Items.Add(Go + "\n");
// Updating output for the inventories
lblStrawberries.Text = strawberries + " oz";
lblBananas.Text =   bananas + " oz";
lblHoney.Text =  honey + " oz";
lblMilk.Text =  milk + " oz";
lblSmallCups.Text =  smallcups + " oz";
lblLargeCups.Text =  largecups + " oz";
```

```
    // Updating sales and outputs
    //sales += 0;
    // cost;
    lblSales.Text = sales.ToString("c");
    lblProfit.Text = (sales - expenses).ToString("c");

    // Resetting form
    nudQuantity.Value = 0;
    chkStrawberries.Checked = true;
    chkBananas.Checked = false;
    chkHoney.Checked = false;
    rdoSmallCup.Checked = true;
}

private void submit_Click(object sender, EventArgs e)
{
    int id;
    strawberrieslin = 0;
    bananaslin = 0;
    honeylin = 0;
    smallcupslin = 0;
    largecupslin = 0;
    milklin = 0;
    saleslin = 0;
    myDB.UpdateInv(strawberries, bananas, honey, milk, smallcups, largecups);
    myDB.Updateprofit(sales, expenses);
    id=myDB.Addtransaction();
    myDB.DisplayTransaction(OldTran);
    myDB.Addtoitems(listBox,id);
    for (int i = listBox.Items.Count - 1; i >= 0; i--)
    {
        listBox.Items.RemoveAt(i);
    }
    // Resetting form
    nudQuantity.Value = 0;
    chkStrawberries.Checked = true;
    chkBananas.Checked = false;
    chkHoney.Checked = false;
    rdoSmallCup.Checked = true;
}

private void btnRemove_Click(object sender, EventArgs e)
{
    strawberries += strawberrieslin;
    bananas += bananaslin;
    honey += honeylin;
    milk += milklin;
    smallcups += smallcupslin;
    largecups += largecupslin;
    sales -= saleslin;
    lblStrawberries.Text =  strawberries + " oz";
    lblBananas.Text =  bananas + " oz";
    lblHoney.Text =  honey + " oz";
    lblMilk.Text =  milk + " oz";
    lblSmallCups.Text =  smallcups + " oz";
    lblLargeCups.Text =  largecups + " oz";
    lblSales.Text = sales.ToString("c");
```

```
            lblProfit.Text = (sales - expenses).ToString("c");
            // remove the item of listbox
            for (int i = listBox.Items.Count - 1; i >= 0; i--)
            {
                listBox.Items.RemoveAt(i);
            }
            myDB.UpdateInv(strawberries, bananas, honey, milk, smallcups, largecups);
            myDB.Updateprofit(sales, expenses);
            // Resetting form
            nudQuantity.Value = 0;
            chkStrawberries.Checked = true;
            chkBananas.Checked = false;
            chkHoney.Checked = false;
            rdoSmallCup.Checked = true;
            strawberrieslin = 0;
            bananaslin = 0;
            honeylin = 0;
            smallcupslin = 0;
            largecupslin = 0;
            milklin = 0;
            saleslin = 0;
        }
        private void OldTran_SelectedIndexChanged(object sender, EventArgs e)
        {
            myDB.ShowDetail(OldTran.SelectedIndex, Detail);
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.Odbc;
using System.Diagnostics;

namespace Project2
{
    class Database
    {
        OdbcConnection dbConn;
        OdbcCommand dbCmd;
        OdbcDataReader dbReader;

        List<int> ids;

        public Database()
        {
            dbConn = new OdbcConnection("Driver={SQL Server};"
                + "Server=scb-sv-mgis-4.main.ad.rit.edu\\mgismssql;"
                + "DataBase=MGIS350_2181_Group5;"
                + "Uid=Group5;"
                + "Pwd=qqelnrkz&%6082FPEH");

            ids = new List<int>();
        }

        private  void ExecuteQuery(string Query)
        {
            try
            {
                Debug.WriteLine("");
                Debug.WriteLine("SQL Query: " + Query);
                Debug.WriteLine("");

                // Building command to database and executing the query string
                dbCmd = new OdbcCommand(Query, dbConn);
                // Opening the connection
                dbConn.Open();
                // Executing the query and closing the connection
                dbReader = dbCmd.ExecuteReader();
            }
            catch (Exception ex)
            {
                // Determine if database is open.  If so close it.
                if (dbConn.State.ToString() == "Open")
                {
                    this.CloseDatabase();
                }
                // Display Error
                System.Windows.Forms.MessageBox.Show("Error:\n\n" + ex.ToString() +
"\n");
            }
        }

        private void CloseDatabase()
        {
            // Determine if database is open.  If so close it.
            if (dbConn.State.ToString() == "Open")
            {
                // Closing connection
                dbConn.Close();
            }
        }

        public void DisplayTransaction( System.Windows.Forms.ListBox output)
        {
            //clear the current output
            output.Items.Clear();
            ids.Clear();

            // build our query to display the selected transactions

            string query = "SELECT * FROM p3Invoices;";
            ExecuteQuery(query);

            //LOOp through the query

            int i = 1;
            while (dbReader.Read())
            {
                if (dbReader.IsDBNull(0) == false)
                {
                    output.Items.Add("Invoice "+i+ " ordered on " + dbReader["orderdt"]);
                    ids.Add(Convert.ToInt32(dbReader["id"]));
                    i++;
                }
            }
            CloseDatabase();
        }

        public void ShowDetail(int input, System .Windows.Forms .ListBox output)
        {
            output.Items.Clear();

            string Query = "select lineItem from dbo.p3LineItems where invoiceId ="
+ids[input] + ";";
            ExecuteQuery(Query);
            // show the detail to listbox
            while (dbReader.Read())
            {
                if (dbReader.IsDBNull(0) == false)
                {
                    output.Items.Add(dbReader["lineItem"]);
                }
            }
            CloseDatabase();
        }

        public int Addtransaction()
        {
            string Query = "insert into dbo.p3Invoices(orderdt) output inserted.id
values(Getdate());";
            ExecuteQuery(Query);
            int id=1;
            while (dbReader.Read())
            {
                if (dbReader.IsDBNull(0) == false)
                {
                    id = Convert.ToInt32(dbReader["id"]);
                }
            }
            CloseDatabase();
            return id;
        }

        public void Addtoitems(System.Windows.Forms.ListBox output,int id)  //is the
basis transaction's box
        {
            for (int i = 0; i < output .Items .Count ; i++)
            {
                string Query = "insert into dbo.p3LineItems(invoiceId,lineItem)
values("+id+",'"+output.Items[i]+"');";
                ExecuteQuery(Query);
                CloseDatabase();
            }

        }

        public void UpdateInv(decimal S,decimal B,decimal H,decimal M,int Small, int Lar)
        {
            string Query = "update p3Inventory set strawberries=" + S + ",bananas=" + B +
",honey=" + H + ",milk=" + M + ",smallcups=" + Small + ",largecups=" + Lar + " where
id=1;";
            ExecuteQuery(Query);
            CloseDatabase();
        }

        public void Updateprofit(decimal sale, decimal expense)
        {
            string Query = "update p3Profits set sales=" + sale + ",expenses=" + expense
+ ";";
            ExecuteQuery(Query);
            CloseDatabase();
        }
        public decimal StrawUp()
        {

            string Query = "select strawberries from p3Inventory; ";
            ExecuteQuery(Query);
            decimal straw=0;
            while (dbReader.Read())
            {
                if (dbReader.IsDBNull(0) == false)
                {
                     straw= Convert .ToDecimal( dbReader["strawberries"]);
                }
            }
            CloseDatabase();
            return straw;
        }
        public decimal BanUp()
```

```
{
    string Query = "select bananas from p3Inventory; ";
    ExecuteQuery(Query);
    decimal bana = 0;
    while (dbReader.Read())
    {
        if (dbReader.IsDBNull(0) == false)
        {
            bana = Convert.ToDecimal(dbReader["bananas"]);
        }
    }
    CloseDatabase();
    return  bana;
}
public decimal honeyUp()
{
    string Query = "select honey from p3Inventory; ";
    ExecuteQuery(Query);
    decimal honey = 0;
    while (dbReader.Read())
    {
        if (dbReader.IsDBNull(0) == false)
        {
           honey= Convert.ToDecimal(dbReader["honey"]);
        }
    }
    CloseDatabase();
    return honey;
}
public decimal milkUp()
{
    string Query = "select milk from p3Inventory; ";
    ExecuteQuery(Query);
    decimal milk = 0;
    while (dbReader.Read())
    {
        if (dbReader.IsDBNull(0) == false)
        {
            milk= Convert.ToDecimal(dbReader["milk"]);
        }
    }
    CloseDatabase();
    return milk;
}
public int smallcupsUp()
{
    string Query = "select smallcups from p3Inventory; ";
    ExecuteQuery(Query);
    int smallcups = 0;
    while (dbReader.Read())
    {
        if (dbReader.IsDBNull(0) == false)
        {
            smallcups = Convert.ToInt32 (dbReader["smallcups"]);
        }
    }
    CloseDatabase();
    return smallcups;
```

```
        }
        public int largecupsUp()
        {
            string Query = "select largecups from p3Inventory; ";
            ExecuteQuery(Query);
            int largecups = 0;
            while (dbReader.Read())
            {
                if (dbReader.IsDBNull(0) == false)
                {
                    largecups = Convert.ToInt32(dbReader["largecups"]);
                }
            }
            CloseDatabase();
            return largecups;
        }
        public decimal salesUp()
        {
            string Query = "select sales from p3Profits;";
            ExecuteQuery(Query);
            decimal sales=0;
            while (dbReader.Read())
            {
                if (dbReader.IsDBNull(0) == false)
                {
                    sales= Convert.ToDecimal (dbReader["sales"]);
                }
            }
            CloseDatabase();
            return sales;
        }
        public decimal expensesUp()
        {
            string Query = "select expenses from p3Profits; ";
            ExecuteQuery(Query);
            decimal expenses=0;
            while (dbReader.Read())
            {
                if (dbReader.IsDBNull(0) == false)
                {
                    expenses = Convert.ToDecimal(dbReader["expenses"]);
                }
            }
            CloseDatabase();
            return expenses;
        }

    }
}
```