C# Group projEct

Too Many Daves

Mckinley Magale & Jonathan Roddy

K00220137 & K0021218

**Brief:**

You are required to write a C# desktop application to help Mrs. McCave to manage her children. The desktop application should have a GUI front-end and help Mrs. McCave with the following family organizational matters.

Note: All Mrs. McCaves family details can be found in the mccaveFamily.txt file. The file contains the child’s nicknames, date of birth and some comments about each child.

1. Upload data – upload data from file.

2. Birthdays – This option will display the children’s names whom have a birthday falling in the next seven days - it should display what age they will be on their next birthday and child’s comments.

3. List children by age - This should print a list of the children's nicknames ordered by age - oldest first.

4. List children in alphabetical order - This should print a list of the children's nicknames in alphabetical order.

5. List any/all multiple births: nicknames of children who are part of a multiple birth. The program should state if they are twins, triplets, quads etc.

6. Add new child - nickname, date of birth and comment should be saved for the new child. The new details should be save in the file and program data refreshed.

7. Name the next baby - An exciting algorithm that you come up with for Mrs. McCave’s next baby name.

8. Calculate Mrs. McCave’s children’s allowance for the current month. As per the rules set down by the Irish state see [http://www.citizensinformation.ie/en/social\_welfare/social\_welfare\_payments/social\_welfare\_payments\_to\_families\_and\_children/child\_ benefit.html](http://www.citizensinformation.ie/en/social_welfare/social_welfare_payments/social_welfare_payments_to_families_and_children/child_%20benefit.html)

9. Calculate Mrs. McCave’s children’s allowance for the year.

10. Planning school times: This will enable Mrs. McCave to do some essential financial plaining. Mrs. McCave should be able to enter a year and the program should produce the projected major educational mile stones for her children.

These include Preschool, Primary school, Secondary School, Third Level, Finished. Note: Assume that all kids go to primary school if they are 5 in that current September, any kids under 5 in that current year are deemed to be pre-schoolers. Assume all children spend 8 years in primary school and 6 years in secondary school and enter third level directly after finishing secondary school and complete a four year in third level.

11. Produce a dynamic info graphic of the McCave family. The info graphic should convey as must information as possible about the McCave family.

**Code:**

**Daves.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace TermProject

{

public class Daves

{

public string name { get; set; }

public string DOB { get; set; }

public string details { get; set; }

public int birth { get; set; }

public int age { get; set; }

public int twins { get; set; }

}

}

**frmMain.cs:**

using System;

using System.IO;

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 TermProject

{

public partial class frmMain : Form

{

public frmMain()

{

InitializeComponent();

}

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

{

Form loadForm = new frmLoad();

loadForm.ShowDialog();

}

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

{

Form birthdayForm = new frmBirthday();

birthdayForm.ShowDialog();

}

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

{

Form letterForm = new frmLetter();

letterForm.ShowDialog();

}

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

{

Form multipleForm = new frmMultiple();

multipleForm.ShowDialog();

}

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

{

Form nameGenerator\_Form = new frmNameGen();

nameGenerator\_Form.ShowDialog();

}

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

{

Form monthlyForm = new frmMonthly();

monthlyForm.ShowDialog();

}

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

{

Form futureForm = new frmFuture\_Plans();

futureForm.ShowDialog();

}

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

{

Form ListByAge = new frmListAge();

ListByAge.ShowDialog();

}

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

{

Form NewChild = new frmNewChild();

NewChild.ShowDialog();

}

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

{

Form Yearly = new frmYearlyAllowance();

Yearly.ShowDialog();

}

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

{

// Form DynamicInfographic = new frmDynamicInfographic();

//DynamicInfographic.ShowDialog();

}

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

{

}

}

}

**frmLoad.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 TermProject

{

public partial class frmLoad : Form

{

public frmLoad()

{

InitializeComponent();

}

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

{

try

{

OpenFileDialog op = new OpenFileDialog();

if(op.ShowDialog()== DialogResult.OK)

{

txtPath.Text = op.FileName;

}

}

catch

{

}

}

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

{

try

{

try

{

string[] lines = System.IO.File.ReadAllLines(txtPath.Text.Trim());

foreach (string line in lines)

{

listTextFile.Items.Add(line);

}

}

catch { }

}

catch { }

}

}

}

**frmBirthday.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;

using System.IO;

namespace TermProject

{

public partial class frmBirthday : Form

{

List<Daves> McCaves = new List<Daves>();

public frmBirthday()

{

InitializeComponent();

listBirthday.View = View.Details;

listBirthday.FullRowSelect = true;

listBirthday.Columns.Add("Name", 200);

listBirthday.Columns.Add("Age", 100);

listBirthday.Columns.Add("Traits", 200);

string[] date = new string[3];

int[] birth = new int[3];

string[] row = new string[3];

string[] column = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\list.txt");

foreach (string line in column)

{

row = line.Split(',');

McCaves.Add(new Daves() { name = row[0], DOB = row[1], details = row[2] });

}

for (int i = 0; i <= McCaves.Count; i++)

{

int years = 0;

int day=0, month=0, year=0, days=0;

date = McCaves[i].DOB.Split();

day = Int32.Parse(date[0]);

month = Int32.Parse(date[1]);

year = Int32.Parse(date[2]);

DateTime age = new DateTime(DateTime.Now.Year, DateTime.Now.Month, DateTime.Now.Day);

DateTime dob2 = new DateTime(year, month, day);

years = DateTime.Now.Subtract(age).Days;

days = DateTime.Now.Subtract(dob2).Days;

years = (years / 365);

days = (days \* -1);

if (days > 0)

{

years--;

}

if ((days <= 7) & (days >= 0))

{

add(McCaves[i].name, " " + years + " ", McCaves[i].details);

}

};

}

private void add(string name, string dob, string traits)

{

string[] row = { name, dob, traits };

ListViewItem item = new ListViewItem(row);

listBirthday.Items.Add(item);

}

}

}

**frmListAge.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;

using System.IO;

namespace TermProject

{

public partial class frmListAge : Form

{

public static int Age(string DOB)

{

string[] DMY = new string[3];

int years = 0;

DMY = DOB.Split('/');

int day = Int32.Parse(DMY[0]);

int month = Int32.Parse(DMY[1]);

int year = Int32.Parse(DMY[2]);

DateTime age = new DateTime(year, DateTime.Now.Month, DateTime.Now.Day);

DateTime dob2 = new DateTime(2018, month, day);

years = DateTime.Now.Subtract(age).Days;

int days = DateTime.Now.Subtract(dob2).Days;

days += 1;

years = (years / 365);

if (days < 0)

{

years--;

}

return years;

}

public static int dob(string dob)

{

string[] date = new string[3];

date = dob.Split('/');

int day = Int32.Parse(date[0]);

int month = Int32.Parse(date[1]);

int year = Int32.Parse(date[2]);

DateTime newDob = new DateTime(year, month, day);

int dates = DateTime.Now.Subtract(newDob).Days;

return dates;

}

public frmListAge()

{

InitializeComponent();

string[] DMY = new string[3];

int[] DOB = new int[3];

string[] row = new string[3];

string[] lines = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\list.txt");

List<Daves> McCaves = new List<Daves>();

foreach (string line in lines)

{

row = line.Split('/');

McCaves.Add(new Daves() { name = row[0], DOB = row[1], details = row[2], birth = 0 });

}

lvListAge.View = View.Details;

lvListAge.FullRowSelect = true;

lvListAge.Columns.Add("Name", 130);

lvListAge.Columns.Add("Age", 130);

List<Daves> temp = new List<Daves>();

temp = McCaves;

for (int y = 0; y <= McCaves.Count; y++)

{

McCaves[y].birth = dob(McCaves[y].DOB);

}

McCaves = McCaves.OrderBy(x => x.birth).ToList();

for (int j = 0; j < McCaves.Count; j++)

{

McCaves[j].age = Age(McCaves[j].DOB);

add(McCaves[j].name, "" + McCaves[j].age);

}

}

private void add(string name, string dob)

{

string[] row = { name, dob };

ListViewItem item = new ListViewItem(row);

lvListAge.Items.Add(item);

}

}

}

**frmLetter:**

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 TermProject

{

public partial class frmLetter : Form

{

public frmLetter()

{

InitializeComponent();

List<Daves> McCaves = new List<Daves>();

string[] date = new string[3];

int[] birth = new int[3];

string[] row = new string[3];

string[] column = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\list.txt");

foreach (string line in column)

{

row = line.Split(',');

McCaves.Add(new Daves() { name = row[0], DOB = row[1], details = row[2] });

}

listLetter.View = View.Details;

listLetter.FullRowSelect = true;

listLetter.Columns.Add("Name", 200);

List<Daves> sort = new List<Daves>();

sort = McCaves.OrderBy(Daves => Daves.name).ToList();

for(int i=0;i<sort.Count;i++)

{

listLetter.Items.Add(sort[i].name);

}

}

}

}

**frmMultiple.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 TermProject

{

public partial class frmMultiple : Form

{

List<Daves> McCaves = new List<Daves>();

public frmMultiple()

{

InitializeComponent();

string[] date = new string[3];

int[] birth = new int[3];

string[] row = new string[3];

string[] column = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\list.txt");

foreach (string line in column)

{

row = line.Split(',');

McCaves.Add(new Daves() { name = row[0], DOB = row[1], details = row[2] });

}

//Single Borns Only

listSingle.View = View.Details;

listSingle.FullRowSelect = true;

listSingle.Columns.Add("~~~~~~~~~", 100);

//Twins Only

listTwins.View = View.Details;

listTwins.FullRowSelect = true;

listTwins.Columns.Add("~~~~~~~~~", 100);

//Triplets Only

listTriplets.View = View.Details;

listTriplets.FullRowSelect = true;

listTriplets.Columns.Add("~~~~~~~~~", 100);

for (int i = 0; i < McCaves.Count; i++)

{

int dates;

dates = DOB(McCaves[i].DOB);

for (int count = 0; count < McCaves.Count; count++)

{

int day = DOB(McCaves[count].DOB);

if (count != i)

{

if (dates == day)

{

McCaves[i].twins++;

}

}

}

}

for (int j = 0; j < McCaves.Count; j++)

switch (McCaves[j].twins)

{

case 0:

listSingle.Items.Add(McCaves[j].name);

break;

case 1:

listTwins.Items.Add(McCaves[j].name);//twin box

break;

case 2:

listTriplets.Items.Add(McCaves[j].name);//triplet box

break;

default:

Console.WriteLine("Error, Data not Found!");

break;

}

}

public static int DOB(string dob)

{

string[] date = new string[3];

date = dob.Split('/');

int day = Int32.Parse(date[0]);

int month = Int32.Parse(date[1]);

int year = Int32.Parse(date[2]);

DateTime newDob = new DateTime(year, month, day);

int dates = DateTime.Now.Subtract(newDob).Days;

return dates;

}

}

}

**frmNewChild.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;

using System.IO;

namespace TermProject

{

public partial class frmNewChild : Form

{

List<Daves> McCaves = new List<Daves>();

public frmNewChild()

{

InitializeComponent();

}

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

{

string[] row = new string[3];

string[] lines = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\list.txt");

List<Daves> McCaves = new List<Daves>();

foreach (string line in lines)

{

row = line.Split(',');

McCaves.Add(new Daves() { name = row[0], DOB = row[1], details = row[2], birth = 0 });

}

}

private void add(string name, string dob, string comments)

{

string[] row = { name, dob, comments };

ListViewItem item = new ListViewItem(row);

}

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

{

string NewName = txtName.Text;

string NewDob = dtpDOB.Text;

string NewComment = txtComment.Text;

McCaves.Add(new Daves() { name = NewName, DOB = NewDob, details = NewComment });

TextWriter tw = new StreamWriter(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\list.txt", true);

//Write to file

for (int i = 0; i < McCaves.Count; i++)

{

tw.WriteLine(McCaves[i].name + "," + McCaves[i].DOB + "," + McCaves[i].details);

}

MessageBox.Show("Congratulations on your new child " + txtName.Text + "!");

//Close File

tw.Close();

txtName.Text = "";

dtpDOB.Text = "";

txtComment.Text = "";

for (int i = 0; i < McCaves.Count; i++)

{

add(McCaves[i].name, McCaves[i].DOB, McCaves[i].details);

}

txtName.Text = "";

txtComment.Text = "";

}

private void dtpDOB\_ValueChanged(object sender, EventArgs e)

{

}

}

}

**frmNameGen.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 TermProject

{

public partial class frmNameGen : Form

{

string[] firstNames = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\nameList.txt");

string[] middleNames = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\middleName\_List.txt");

Random rand = new Random();

public frmNameGen()

{

InitializeComponent();

}

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

{

int iFirst = rand.Next(firstNames.Length);

int iMiddle = rand.Next(middleNames.Length);

this.txtBox\_First.Text = firstNames[iFirst];

this.txtBox\_Middle.Text = middleNames[iMiddle];

}

}

}

**frmMonthly.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;

using System.IO;

namespace TermProject

{

public partial class frmMonthly : Form

{

List<Daves> McCaves = new List<Daves>();

public frmMonthly()

{

InitializeComponent();

listMonthly.View = View.Details;

listMonthly.FullRowSelect = true;

listMonthly.Columns.Add("Name", 200);

listMonthly.Columns.Add("Total", 200);

string[] date = new string[3];

int[] birth = new int[3];

string[] row = new string[3];

string[] column = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\list.txt");

foreach (string line in column)

{

row = line.Split(',');

McCaves.Add(new Daves() { name = row[0], DOB = row[1], details = row[2] });

}

int presentMonth = DateTime.Now.Month;

int presentDay = DateTime.Now.Day;

int presentYear = DateTime.Now.Year;

presentYear -= 18;

DateTime peak = new DateTime(presentYear,presentMonth,presentDay);

int adult = DateTime.Now.Subtract(peak).Days;

int gained = 0;

for (int i = 0; i < McCaves.Count; i++)

{

int days;

days = DOB(McCaves[i].DOB);

for (int j = 0; j < McCaves.Count; j++)

{

int day = DOB(McCaves[j].DOB);

if (j != i)

{

if (days == day)

{

McCaves[i].twins++;

}

}

}

Console.WriteLine(McCaves[i].name + " " + days + " " + McCaves[i].twins);

Console.WriteLine();

if (adult > days)

{

switch (McCaves[i].twins)

{

case 0:

gained += 140;

add(McCaves[i].name, "140");

break;

case 1:

gained += 210;

add(McCaves[i].name, "210");

break;

case 2:

gained += 280;

add(McCaves[i].name, "280");

break;

default:

Console.WriteLine("Error,Error");

break;

}

}

txtDisplay.Text = ("€" + gained);

}

}

private void add(string name, string money)

{

string[] row = { name, money };

ListViewItem info = new ListViewItem(row);

listMonthly.Items.Add(info);

}

public static int DOB(string dob)

{

string[] date = new string[3];

date = dob.Split('/');

int day = Int32.Parse(date[0]);

int month = Int32.Parse(date[1]);

int year = Int32.Parse(date[2]);

DateTime newDob = new DateTime(year, month, day);

int dates = DateTime.Now.Subtract(newDob).Days;

return dates;

}

}

}

**frmYearlyAllowance.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 TermProject

{

public partial class frmYearlyAllowance : Form

{

public frmYearlyAllowance()

{

InitializeComponent();

}

public static int dob(string dob)

{

string[] date = new string[3];

date = dob.Split('/');

int day = Int32.Parse(date[0]);

int month = Int32.Parse(date[1]);

int year = Int32.Parse(date[2]);

DateTime newDob = new DateTime(year, month, day);

int dates = DateTime.Now.Subtract(newDob).Days;

return dates;

}

List<Daves> McCaves = new List<Daves>();

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

{

string[] DMY = new string[3];

int[] DOB = new int[3];

string[] row = new string[3];

string[] lines = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\list.txt");

foreach (string line in lines)

{

row = line.Split(',');

McCaves.Add(new Daves() { name = row[0], DOB = row[1], details = row[2], birth = 0 });

}

lvMonthlyEarnings.View = View.Details;

lvMonthlyEarnings.FullRowSelect = true;

lvMonthlyEarnings.Columns.Add("Month", 100);

lvMonthlyEarnings.Columns.Add("Amount per Month", 110);

}

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

{

int earned = 0;

int year18 = Convert.ToInt32(txtSelect.Text);

int Counter = 0;

year18 -= 18;

int daynow = 1;

int monthnow = 1;

int[] ages = new int[25];

do

{

DateTime born = new DateTime((year18 + 18), monthnow, daynow);

DateTime adult = new DateTime(year18, monthnow, daynow);

int adults = DateTime.Now.Subtract(adult).Days;

int baby = DateTime.Now.Subtract(born).Days;

ages[monthnow] = adults;

ages[monthnow + 12] = baby;

monthnow++;

} while (monthnow <= 12);

int monthearn = 0;

//Calculate Welfare

for (int i = 0; i < McCaves.Count; i++)

{

int days2;

days2 = dob(McCaves[i].DOB);

for (int intrnl = 0; intrnl < McCaves.Count; intrnl++)

{

int days3 = dob(McCaves[intrnl].DOB);

if (intrnl != i)

{

if (days2 == days3)

{

McCaves[i].twins++;

}

}

}

}

for (int z = 1; z <= 12; z++)

{

monthearn = 0;

for (int i = 0; i < McCaves.Count; i++)

{

int days2;

days2 = dob(McCaves[i].DOB);

if (ages[z] > days2 && ages[z + 12] < days2)

{

switch (McCaves[i].twins)

{

case 0:

earned += 140;

monthearn += 140;

break;

case 1:

earned += 210;

monthearn += 210;

break;

case 2:

earned += 280;

monthearn += 280;

break;

default:

break;

}

}

}

add("Month: " + z, "Earned: " + monthearn);

txtTotal.Text = ("€" + earned);

}

Counter++;

if (Counter == 2)

{

this.Close();

}

}

private void add(string month, string amount)

{

string[] row = { month, amount };

ListViewItem item = new ListViewItem(row);

lvMonthlyEarnings.Items.Add(item);

}

private void frmYearlyAllowance\_Load\_1(object sender, EventArgs e)

{

}

}

}

**frmFuture\_Plans.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 TermProject

{

public partial class frmFuture\_Plans : Form

{

List<Daves> McCaves = new List<Daves>();

public static int DOB(string dob)

{

string[] date = new string[3];

date = dob.Split('/');

int day = Int32.Parse(date[0]);

int month = Int32.Parse(date[1]);

int year = Int32.Parse(date[2]);

DateTime newDob = new DateTime(year, month, day);

int dates = DateTime.Now.Subtract(newDob).Days;

return dates;

}

public frmFuture\_Plans()

{

InitializeComponent();

string[] date = new string[3];

int[] birth = new int[3];

string[] row = new string[3];

string[] column = System.IO.File.ReadAllLines(@"C:\Users\Ken\OneDrive - Limerick Institute Of Technology\Programming\2017-2018\Programming C Sharp\Group Project\TermProject\TermProject\list.txt");

string[] sort = new string[5];

foreach (string line in column)

{

row = line.Split(',');

McCaves.Add(new Daves() { name = row[0], DOB = row[1], details = row[2] });

}

listPreSchool.View = View.Details;

listPreSchool.FullRowSelect = true;

listPreSchool.Columns.Add("~~~~~~~~~~~", 100);

listPrimarySchool.View = View.Details;

listPrimarySchool.FullRowSelect = true;

listPrimarySchool.Columns.Add("~~~~~~~~~~~", 100);

listSecondarySchool.View = View.Details;

listSecondarySchool.FullRowSelect = true;

listSecondarySchool.Columns.Add("~~~~~~~~~~~", 100);

listCollege.View = View.Details;

listCollege.FullRowSelect = true;

listCollege.Columns.Add("~~~~~~~~~~~", 100);

listFinished.View = View.Details;

listFinished.FullRowSelect = true;

listFinished.Columns.Add("~~~~~~~~~~~", 100);

}

private void add(string name)

{

string[] row = { name};

ListViewItem info = new ListViewItem(row);

listPreSchool.Items.Add(info);

}

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

{

string year = txtBox\_yearInput.Text;

int preSchool = Convert.ToInt32(year);

preSchool -= 5;

int primarySchool = preSchool - 8;

int secondarySchool = primarySchool - 6;

int college = secondarySchool - 4;

DateTime pre = new DateTime(preSchool, 9, 30);

DateTime prime = new DateTime(primarySchool, 9, 30);

DateTime secondary = new DateTime(secondarySchool, 9, 30);

DateTime third = new DateTime(college, 9, 30);

int inPre = DateTime.Now.Subtract(pre).Days;

int inPrime = DateTime.Now.Subtract(prime).Days;

int inSecond = DateTime.Now.Subtract(secondary).Days;

int inThird = DateTime.Now.Subtract(third).Days;

int age;

for (int i = 0; i < McCaves.Count; i++)

{

age = DOB(McCaves[i].DOB);

if (age <= inPre)

{

listPreSchool.Items.Add(McCaves[i].name);

}

else if (age <= inPrime && age > inPre)

{

listPrimarySchool.Items.Add(McCaves[i].name);

}

else if (age <= inSecond && age > inPrime)

{

listSecondarySchool.Items.Add(McCaves[i].name);

}

else if (age <= inThird && age > inSecond)

{

listCollege.Items.Add(McCaves[i].name);

}

else

{

listFinished.Items.Add(McCaves[i].name);

}

}

}

}

}

**frmDynamicInfographic.cs:**

**Pseudocodes:**

**frmBirthday:**

1. Create all necessary Variables.
2. Read from External File.
3. Consider each lines from the text file as rows and columns.
4. Split each lines into one string each for each row.
5. Split the date into three parts : day , month and year.
6. Convert date into Integer.
7. Use Subtract Method to find out how far apart their birthdays are.
8. Use Subtract Method to find out how many days till/after their birthdays from the present time.
9. See if the date of their birthdays lies on this present week.
10. Output the birthdays that is occurring during the present week.

**frmListAge:**

1. Create all necessary Variables.
2. Read from External File.
3. Consider each lines from the text file as rows and columns.
4. Split each lines into one string each for each row.
5. Split the date into three parts : day , month and year.
6. Convert date into Integer.
7. Use Subtract Method to find out how far apart their birthdays are in years.
8. Sort all by age in ascending order.

**frmLetter:**

1. Create all necessary Variables.
2. Read from External File.
3. Consider each lines from the text file as rows and columns.
4. Split each lines into one string each for each row.
5. Add split strings to a Generic List<>.
6. Sort their names in an ascending order.
7. Duplicate the result of Sorting.
8. Output the duplicate.

**frmNameGen:**

1. Create two external text files that’ll hold a list of names.
2. Create all necessary Variables.
3. Read from External File.
4. Select random names from the external files.
5. Output results.

**frmMultiple:**

1. Create all necessary Variables.
2. Read from External File.
3. Consider each lines from the text file as rows and columns.
4. Split each lines into one string each for each row.
5. Add split strings to a Generic List<>.
6. Convert date into Integer.
7. Use Subtract Method to find out how far apart their birthdays are in years.
8. Sort the results by placing the person who shares the same birthday together with another person.
9. Output the results.

**frmNewChild:**

1. Create all necessary Variables.
2. Read from External File.
3. Allow user to input their Name, D.O.B and Traits.
4. Split each line into one string each.
5. Add split strings to a Generic List<>.
6. Write the strings to an external file and allow to save the file when the program terminates.
7. When the process is done, output a message that indicates if the process was a success or not.

**frmMonthly.cs:**

1. Create all necessary Variables.
2. Read from External File.
3. Consider each lines from the text file as rows and columns.
4. Split each lines into one string each for each row.
5. Add split strings to a Generic List<>.
6. Convert date into Integer.
7. Use Subtract Method to find out how far apart their birthdays are in years.
8. Find how many people who shares the same birthday.
9. If twins have been found or triplets, store these data.
10. Calculate the monthly earnings, with the singles first then to the twins and then to the triplets.
11. Sum it all up and output the results.

**frmYearlyAllowance.cs:**

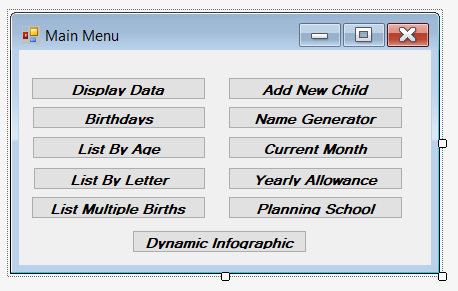
1. Create all necessary Variables.
2. Read from External File.
3. Consider each lines from the text file as rows and columns.
4. Split each lines into one string each for each row.
5. Add split strings to a Generic List<>.
6. Convert date into Integer.
7. Use Subtract Method to find out how far apart their birthdays are in years.
8. Find how many people who shares the same birthday.
9. Find out if a person has turn eighteen on the present month.
10. Loop this cycle for twelve months.
11. If twins have been found or triplets, store these data.
12. Calculate the monthly earnings, with the singles first then to the twins and then to the triplets and then store the data found.
13. Sum it all up and output the results.

**frmFuture\_Plans.cs:**

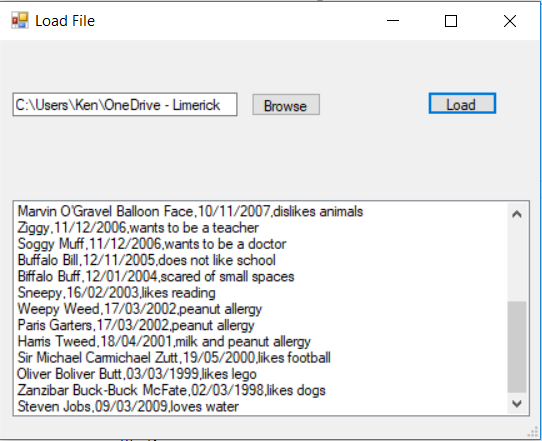
1. Create all necessary Variables.
2. Read from External File.
3. Consider each lines from the text file as rows and columns.
4. Split each lines into one string each for each row.
5. Add split strings to a Generic List<>.
6. Convert date into Integer.
7. Find the normal school start dates.
8. Use Subtract Method to find out how far apart start dates from the current date.
9. Compare results with the D.O.B.
10. Sort results in four columns, which are the four levels education has currently.
11. Output results in these columns.

**GUI Designs:**

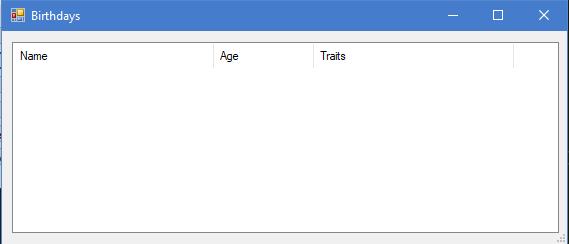
**Main GUI:**



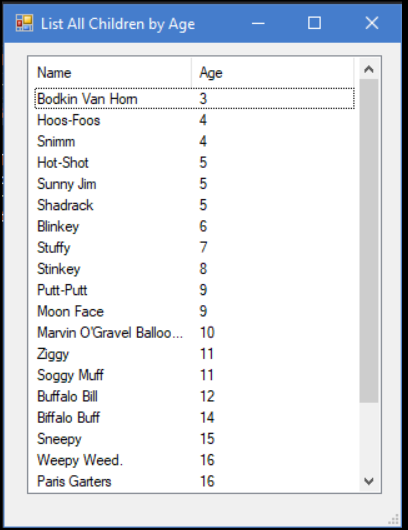
**Display Data:**



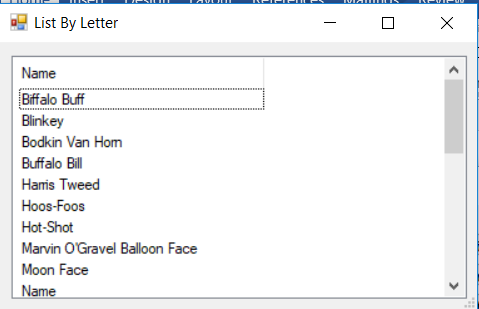
**Birthdays:**



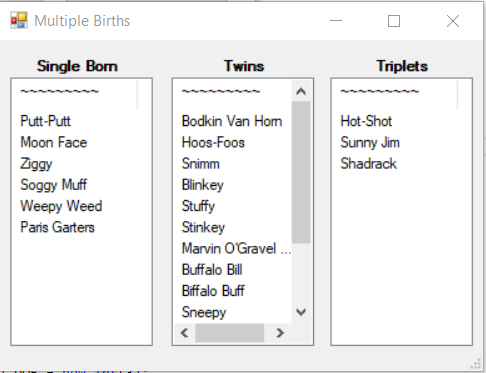
**List By Age:**



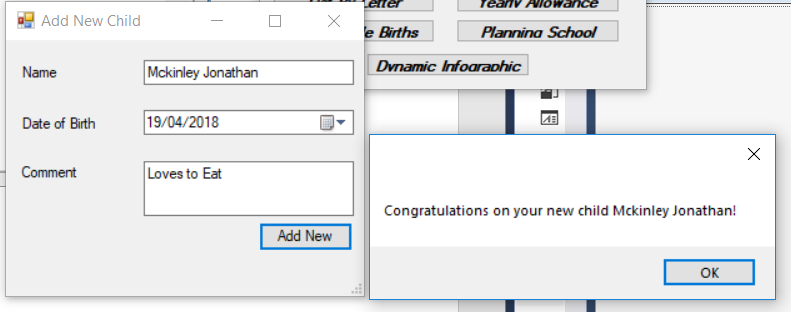
**List By Letter:**



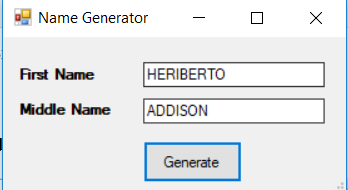
**List Multiple Births:**



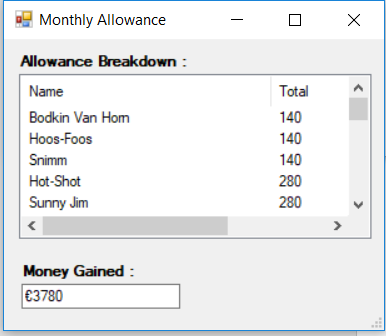
**Add New Child:**



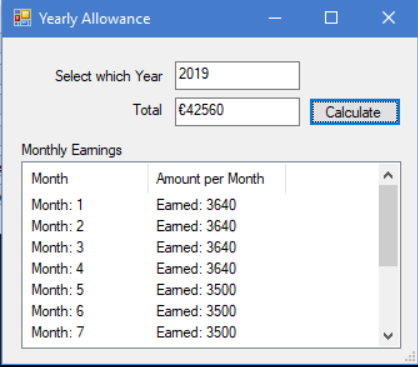
**Name Generator:**



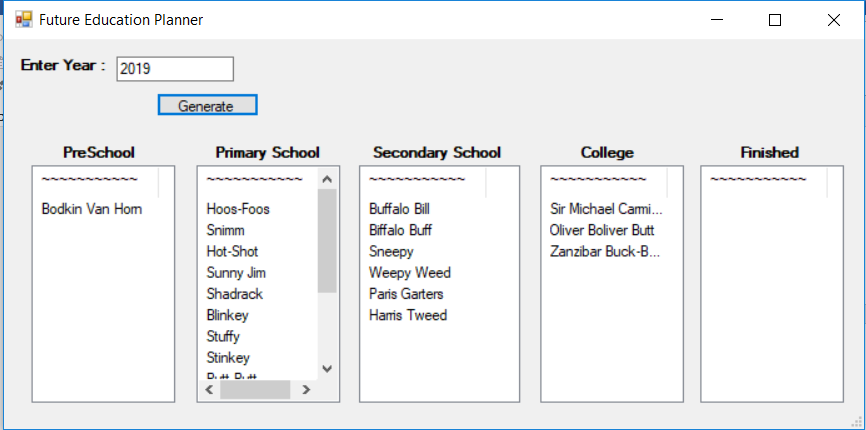
**Current Month:**



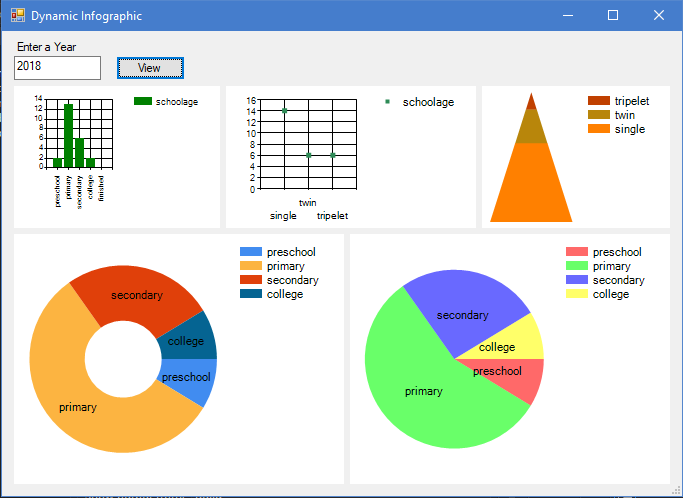
**Yearly Allowance:**



**Planning School:**



**Dynamic Infographic:**



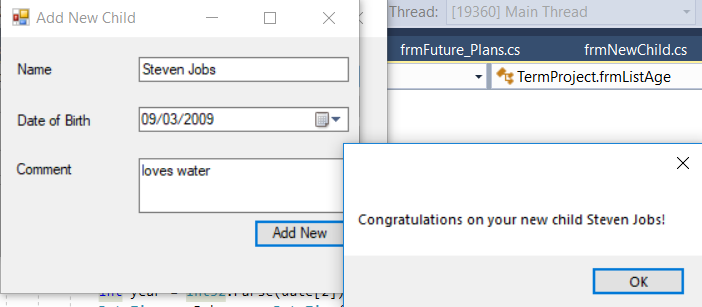
**Test plan:**

* Testing is done by adding a new member to the list of children.
* The Child is to be named Steven Jobs and was born on the 09/03/2009.
* Steven should be 9 years old
* Steven should appear between Soggy Muff and Stinky, in order by name.
* Run random name generator 10 times and check if no duplicates occur.
* Steven should not be a part of the multiple birth group (twins and triplets).
* Check Monthly Allowance before and after adding Steven.
* Check Yearly Allowance and check if it changes for when the children reach 18 years of age
* Steven should be in Primary School for the year 2018.

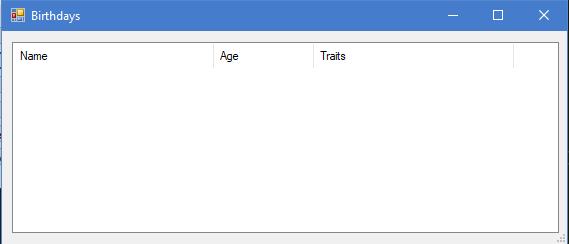
**Test results:**

**Test Results for the 18/04/2018**

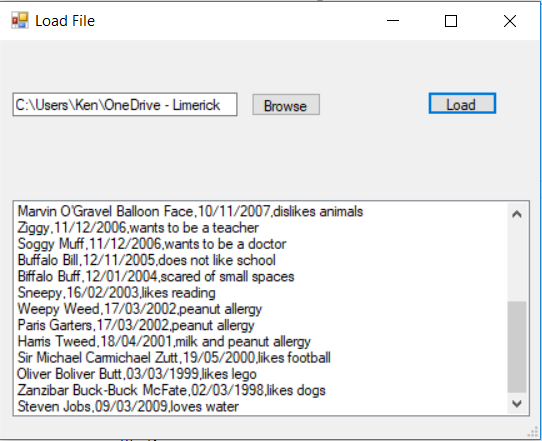
**Add New Child:**



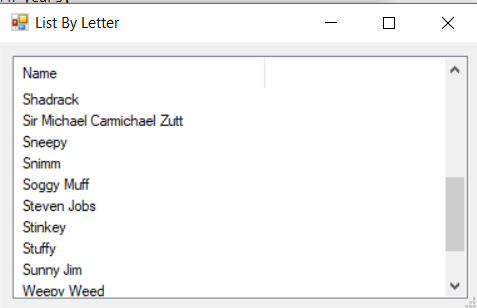
**Birthdays:**



**Display Data:**



**List by Letters:**



**Education:**

