C# cheat sheet

* on top of the entity class,

[Serializable] , then getter and setter methods

* GetXmlFile (Xml file should be in bin/debug folder if want to read)

OpenFileDialog openFileDialog = new OpenFileDialog();

openFileDialog.Filter = "XML Files (\*.xml)|\*.xml";

openFileDialog.Title = "Select a XML File";

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

{ add try catch statement

StreamReader transactionFile = new StreamReader(openFileDialog.FileName);

XmlSerializer rentalHouserSerializer = new XmlSerializer(typeof(List<NewWestminsterRentalHousing>));

rentalHousingList = rentalHouserSerializer.Deserialize(rentalHouseFile) as List<NewWestminsterRentalHousing>;

rentalHouseFile.Close();

* InitializeDataGridViewRentalHousing

DataGridViewTextBoxColumn[] columns = new DataGridViewTextBoxColumn[] {

new DataGridViewTextBoxColumn() { Name = "BuildingId" },

new DataGridViewTextBoxColumn() { Name = "Address" }, …

dataGridViewRentalHousing.Columns.AddRange(columns);

* Add specific fileds to list using lamda expression

var neighborhoods = from house in rentalHousingList

orderby house.Neighborhood

select house.Neighborhood.ToUpper();

listBoxNeighborhoods.Items.AddRange(neighborhoods.Distinct().ToArray());Register event handler

buttonBuildingNameSearch.Click += ButtonBuildingNameSearch\_Click

* Reset to default

listBoxNeighborhoods.SelectedIndexChanged -= ListBoxNeighborhoods\_SelectedIndexChanged;

textBoxBuildingNameSearch.Clear();

for (int i = 0; i < listBoxNeighborhoods.Items.Count; i++) // select all neighborhoods

listBoxNeighborhoods.SetSelected(i, true);

DisplayRentalHousing();

listBoxNeighborhoods.SelectedIndexChanged += ListBoxNeighborhoods\_SelectedIndexChanged;

-Displaying

dataGridViewRentalHousing.Rows.Clear();

List<string> selectedItems = new List<string>();

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

{

selectedItems.Add(listBoxNeighborhoods.SelectedItems[i].ToString());

}

string userInput = textBoxBuildingNameSearch.Text.ToUpper();

Expression<Func<NewWestminsterRentalHousing, bool>> expression = p => selectedItems.Contains(p.Neighborhood) &&

((p.BuildingName == userInput) || !isSearchClicked)

((p.Price <= maxPrice && p.Price >= minPrice) || !checkBoxSearchPrice.Checked);

var items = rentalHousingList.Where(expression.Compile()).OrderBy(p => p.Neighborhood).ThenBy(p => p.BuildingName);

foreach (NewWestminsterRentalHousing house in items)

{

dataGridViewRentalHousing.Rows.Add(house.BuildingId, house.Address, house.BuildingName, house.NumberOfResidences, house.Neighborhood);

TotalNumberOfRegistence += house.NumberOfResidences;

}

Format averagePrice.ToString("C2");

numberOfTransaction = selectedHouses.Count();

if (numberOfTransaction > 0)

averagePrice = selectedHouses.Average(p => p.Price);

else

averagePrice = 0;

 var query2 = students.GroupBy(s => s.Name[1]).OrderByDescending(s => s.Key);  
 var query2Sql = from student in students  
                 group student by student.Name[1] into studentGroup  
                 orderby studentGroup descending  
                 select studentGroup;

 var querySql = from student in students  
                select new  
               {  
                Name = student.Name,

  AverageScore = student.Scores.Average(),  
                 MaxScore = student.Scores.Max(),  
                MinScore = student.Scores.Min()  
                           };  
  
var query1 = students.Select(s =>   
                            new { Name = s.Name,   
                                  Average = s.Scores.Average(),  
                                  MaxScore = s.Scores.Max(),  
                                  MinScore = s.Scores.Min() });