```
/***********************************Part 13***********************************/

namespace BreakNCont
//break and continue are two tools within loops.  Break is used to break out of a
loop if a condition is met.
//continue is used to skip one iteration in a loop if a certain condition occurs

{
    class BreakContClass
    {
        static void BreakUpNContinue()
        {
            //When this runs try to determine WHY there are NO FOURS and WHY there
is only ONE TEN.

            for (int i = 0; i < 15; i++)
            {
                if (i == 4)
                {
                    continue;
                }

                Console.WriteLine(i);
                if (i == 10)
                {
                    break;
                }
                Console.WriteLine(i);

            }

        }
        static void Main(string[] args)
        {
            BreakUpNContinue();
        }

    }

}

/***********************************Part 14***********************************/

namespace Arrays
{
    class ArraysClass
    {
        static void GoArrayGo()
        {
            //Arrays are a way to strore loads of data
            //You can create arrays of any data type
            //************************
            //need to add system.Linq;  to use to min,max,sort funstions at the end
            //************************

            string[] hockeyTeams = { "Flames", "Canucks", "Leafs", "Oilers" };
            Console.WriteLine(hockeyTeams[2]);
            int[] pieNumbers = { 3, 1, 4, 1, 5, 9 };
            int amountOfPie = pieNumbers.Length;
            Console.WriteLine("You have: "+ amountOfPie+" digits of pie.");
            for(int i=0; i<amountOfPie; i ++)
            {
                Console.WriteLine(pieNumbers[i]);
            }
            //can exchange values
            pieNumbers[0] = 8;
            Console.WriteLine("Now pie starts with an 8.");
            for (int i = 0; i<amountOfPie; i++)
            {
                Console.WriteLine(pieNumbers[i]);
            }
            //some other array tools
            Array.Sort(pieNumbers);
            Console.WriteLine("Let's sort the digits of pie:");
            foreach (int i in pieNumbers)
            {
                Console.WriteLine(i);
            }
            //need to add system.Linq;
            Console.WriteLine("bigest "+pieNumbers.Max());  // returns the largest
value
            Console.WriteLine("littlist "+pieNumbers.Min());  // returns the
smallest value
            Console.WriteLine("sum "+pieNumbers.Sum());  // returns the sum of
elements

        }
        static void Main(string[] args)
        {
            GoArrayGo();
        }

    }

}

/***********************************Part 15***********************************/

namespace MultiDimensionalArrays

{
    class MultiArraysClass
    {
        static void GoMultiArrayGo()
        {
            int[,] counting = { {1,3,5,7 },{2,4,6,8 } };
            int[,] tripArray = { { 1, 3, 5, 7 }, { 2, 4, 6, 8 }, { 3, 6, 9, 12 } };
            // when acessing an array it goes [row,coloumn]
            Console.WriteLine(tripArray[0, 2]);
            Console.WriteLine(tripArray[2, 3]);

            Console.WriteLine(counting[0, 2]);
            counting[0, 2] = 8;
            Console.WriteLine(counting[0, 2]);
            Console.WriteLine("LineBreak");
            //display all the elements in an array going though the rows one by one
            foreach (int i in counting)
            {
                Console.WriteLine(i);
            }
            foreach (int i in tripArray)
            {
                Console.WriteLine(i);
            }
            Console.WriteLine("NewLoop");
            //kind of odd to put into multidimensional array if you will print out 
like this
            //Becasue its an array need to use getlength instead of length
            for (int i = 0; i < counting.GetLength(0); i++)
            {
                for (int j = 0; j < counting.GetLength(1); j++)
                {
                    Console.WriteLine(counting[i, j]);
                }
            }
        }
        static void Main(string[] args)
        {
            GoMultiArrayGo();
        }

    }

}

/***********************************Part 16***********************************/

namespace MakingLists

{
    class ListIt
    {
        static void ToDoList()
        {
            //A list stores values like an array, but elements can be added or 
removed at will.
            //An array can only hold a fixed number of values
            //List give you greater flexability

            List<int> userAgeList = new List<int>();
            List<int> numbersList = new List<int> { 3, 1, 4, 1 };

            Console.WriteLine(numbersList[0]);

            numbersList.Add(5);
            numbersList.Add(9);

            Console.WriteLine("This list now has " + numbersList.Count() + " 
numbers.");

            numbersList.Insert(2, 8);

            foreach (int i in numbersList)
            {
                Console.WriteLine(i);
            }

            numbersList.Remove(8);

            Console.WriteLine("After removal");

            foreach (int i in numbersList)
            {
                Console.WriteLine(i);
            }

            //or do it by index

            numbersList.Insert(2, 8);
            numbersList.RemoveAt(2);

            Console.WriteLine("add and take away again");

            foreach (int i in numbersList)
            {
                Console.WriteLine(i);
            }

            Console.WriteLine(numbersList.Contains(4));

            numbersList.Clear();
            Console.WriteLine("Empty now? " + numbersList.Count());

        }
        static void Main(string[] args)
        {
            ToDoList();
        }

    }

}

/***********************************Part 17***********************************/

namespace BasicErrorHandling

{
    class NoErrors
    {
        static void ErrorHandler()
        {
            /*            The try-catch-finally statement controls how
                        the program proceeds when an error occurs. The syntax is as
follows:

                        try
                        {
                            do something
                        }
                        catch (type of error)
                        {
                            do something else when an error occurs
                        }
                        finally
                        {
                            do this regardless of whether the try or catch condition
is
                            met.
                        }*/

            int numerator, denominator; //can define two ints at once

            try
            {
                Console.Write("Please enter the numerator: ");
                numerator = Convert.ToInt32(Console.ReadLine());
                Console.Write("Please enter the denominator: ");
                denominator = Convert.ToInt32(Console.ReadLine());
                try
                {
                    Console.WriteLine("The result is {0}.",
                    numerator/denominator);
                }
                catch (Exception e)
                {
                    Console.WriteLine(e.Message);
                }
            }
            catch (Exception e)
            {
                Console.WriteLine(e.Message);
            }
            finally
            {
                Console.WriteLine("----End of Error Handling Example----");
            }

            Console.WriteLine("One more example... wooohooo");
            //You can also have very specific exceptions... there are loads of them

            int choice = 0;
            int[] numbers = { 10, 11, 12, 13, 14, 15 };
            Console.Write("Please enter the index of the array: ");
            try
            {
                choice = Convert.ToInt32(Console.ReadLine());
                //New
                Console.WriteLine("numbers[{0}] = {1}" , choice , numbers[choice]);
            }
            catch (IndexOutOfRangeException)
            {
                Console.WriteLine("Error: Index should be from 0 to 5.");
            }
            catch (FormatException)
            {
                Console.WriteLine("Error: You did not enter an integer.");
            }
            catch (Exception e)
            {
                Console.WriteLine(e.Message);
            }

        }
        static void Main(string[] args)
        {
            ErrorHandler();
        }

    }

}
```