using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication10

{

class Program

{

static void Main(string[] args)

{

List<string> products = new List<string>()

{

"Sony TV","Sony ABC","Samsung Mobile","Hp LapTop","LG Washing Machine"

};

products.Where(p => p.StartsWith("Sony")).ToList().ForEach(p => Console.WriteLine(p));

//LINQ=language integrated query

var result = from p in products

where p.StartsWith("Sony")

orderby p descending

select p;

foreach (var r in result)

Console.WriteLine(r);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication10

{

class Product

{

public int id;

public string name;

public int price;

}

class Program

{

static void Main(string[] args)

{

List<Product> products = new List<Product>()

{

new Product() {id=1,name="Sony tv",price=40000 },

new Product() {id=2,name="Samsung tv",price=36000 },

new Product() {id=3,name="Sony mobile",price=14000 },

new Product() {id=4,name="Sony laptop",price=66000 },

new Product() {id=5,name="motorola mobile",price=9000 },

new Product() {id=1,name="Sony tv",price=40000 },

};

//LINQ query to print Sony products

var result1 = from q in products

where q.name.StartsWith("Sony")

select q;

foreach (var r in result1)

Console.WriteLine(r.name);

//linq query to print all products price btween 10000 and 40000

var result2=from a in products

where (a.price>10000 && a.price<40000)

select a;

foreach (var r in result2)

Console.WriteLine("{0},{1}",r.name,r.price);

//linq query to print all products in descending order of price

var result3 = from s in products

orderby s.price descending

select s;

foreach (var s in result3)

Console.WriteLine("{0},{1}",s.name,s.price);

//print name, price of all laptops

var result4 = from b in products

where b.name.Contains("laptop")

select b;

foreach (var b in result4)

Console.WriteLine("{0},{1}", b.name, b.price);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication13

{

class Product

{

public int id;

public string name;

public int price;

}

class Program

{

static void Main(string[] args)

{

List<Product> products = new List<Product>()

{

new Product() {id=1,name="Sony Tv",price=13000 },

new Product() {id=2,name="Samsung Tv",price=12000 },

new Product() {id=3,name="Sony Laptop",price=46000 },

new Product() {id=4,name="Sony Mobile",price=9000 }

};

//tsk-1:write linq query to print name,price of product details

//which are greater than 10000 and belong to Sony brand

//in the descending order

var result = from p in products

where ((p.price) > 10000) && (p.name.StartsWith("Sony"))

orderby p.price descending

select p;

foreach(var q in result)

{

Console.WriteLine("{0},{1}",q.name,q.price);

}

//lamda expressions

products.Where(p => p.price > 10000 && p.name.StartsWith("Sony")).ToList().ForEach(p => Console.WriteLine("{0},{1}", p.name, p.price));

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication13

{

class PetShop

{

public string ShopName { get; set; }

public List<string> Pets { get; set; }

}

class Program

{

static void Main(string[] args)

{

List<PetShop> petshops = new List<PetShop>()

{

new PetShop() {ShopName="ABC",Pets=new List<string>() {"Parrot","Tortoise","Dog" } },

new PetShop() {ShopName="XYZ",Pets=new List<string>() {"Cats" } },

new PetShop() {ShopName="MNR",Pets=new List<string>() {"Pegion","Rabbit" } }

};

//write lambda expressions to print the shop name which has max number of pets

Console.WriteLine(petshops.OrderByDescending(P => P.Pets.Count).First().ShopName);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

List<string> names = new List<string>()

{

"Bharath","Shankar","Bhargav","Raghu","Harish"

};

//linq query to print names starting with B

var result1 = from w in names

where w.StartsWith("B")

select w;

foreach (var x in result1)

Console.WriteLine(x);

//linq query to print names whose second character is h

var result2 = from y in names

where y.IndexOf("h")==1

select y;

foreach (var z in result2)

Console.WriteLine(z);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Employee

{

public int id;

public string name;

public string Company

{

get

{

return "Capgemini";

}

}

}

class Program

{

static void Main(string[] args)

{

Employee emp = new Employee();

Console.WriteLine(emp.Company);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Maths

{

public int input;

public int InputFactorial

{

get

{

int i = 1, fact = 1;

for (i = 1; i <= input; i++)

fact = fact \* i;

return fact;

}

}

public int InputSquare

{

get

{

return input \* input;

}

}

}

class Program

{

static void Main(string[] args)

{

Maths obj = new Maths();

Console.WriteLine("Enter a input: ");

obj.input = int.Parse(Console.ReadLine());

Console.WriteLine(obj.InputFactorial);

Console.WriteLine(obj.InputSquare);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Date

{

private int month;

public int Month

{

set

{

if (value <= 12 && value >= 1)

month = value;

else

month = value % 12;

}

get

{

return month;

}

}

}

class Program

{

static void Main(string[] args)

{

Date obj = new Date();

obj.Month = 12;

Console.WriteLine(obj.Month);

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void PrintHi()

{

Console.WriteLine("hiii");

}

static void PrintHello()

{

Console.WriteLine("hlooo");

}

static void PrintGM()

{

Console.WriteLine("good morning");

}

public delegate void MyDelegate();

static void Main(string[] args)

{

MyDelegate obj = new MyDelegate(PrintHi);

obj += PrintHello;

obj += PrintGM;

obj();

obj -= PrintHello;

obj();

Console.ReadLine();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

public static int Factorial(int n)

{

int result=1;

for (int i = 1; i <= n; i++)

result = result \* i;

return result;

}

public static int nCr(int n,int r)

{

return Factorial(n) / (Factorial(n - r) \* Factorial(r));

}

static void Main(string[] args)

{

int input;

Console.WriteLine("Enter the no of rows: ");

input = int.Parse(Console.ReadLine());

int n = input - 1;

for(int r=0;r<=n;r++)

{

Console.Write("{0} ", nCr(n, r));

}

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

Stack data = new Stack();

data.Push(34);

data.Push(87);

data.Push(67);

Console.WriteLine(data.Pop());

Console.WriteLine(data.Pop());

Console.WriteLine(data.Pop());

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

Queue data = new Queue();

data.Enqueue(34);

data.Enqueue(87);

data.Enqueue(67);

Console.WriteLine(data.Dequeue());

Console.WriteLine(data.Dequeue());

Console.WriteLine(data.Dequeue());

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

Queue data = new Queue();

data.Enqueue(34);

data.Enqueue(87);

data.Enqueue(67);

foreach(var p in data)

{

Console.WriteLine(p);

}

Console.WriteLine("Size: " + data.Count);

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

Queue<int> data = new Queue<int>();

data.Enqueue(34);

data.Enqueue(87);

data.Enqueue(67);

foreach(var p in data)

{

Console.WriteLine(p);

}

Console.WriteLine("Size: " + data.Count);

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

Dictionary<int, string> Employee = new Dictionary<int, string>();

Employee.Add(1,"jeevitha");

Employee.Add(2, "sai");

Employee.Add(3, "ravi");

Employee.Add(4, "prasad");

//print employee with id=1

Console.WriteLine(Employee[1]);

//print all employee ids

foreach (var k in Employee.Keys)

{

Console.WriteLine(k);

}

//print all employee names

foreach(var name in Employee.Values)

{

Console.WriteLine(name);

}

//1.jeevitha

//2.ravi

foreach (var k in Employee.Keys)

{

Console.WriteLine("{0}.{1}",k,Employee[k]);

}

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

SortedDictionary<int, string> Employee = new SortedDictionary<int, string>();

Employee.Add(9,"jeevitha");

Employee.Add(2, "sai");

Employee.Add(6, "ravi");

Employee.Add(4, "prasad");

//print employee with id=1

Console.WriteLine(Employee[6]);

//print all employee ids

foreach (var k in Employee.Keys)

{

Console.WriteLine(k);

}

//print all employee names

foreach(var name in Employee.Values)

{

Console.WriteLine(name);

}

//1.jeevitha

//2.ravi

foreach (var k in Employee.Keys)

{

Console.WriteLine("{0}.{1}",k,Employee[k]);

}

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

Dictionary<int, string> Employee = new Dictionary<int, string>();

Employee.Add(9,"jeevitha");

Employee.Add(2, "sai");

Employee.Add(6, "ravi");

Employee.Add(4, "prasad");

//print some code here to print employee ids and respective names in ascending order

//method 1

Employee.Keys.OrderBy(p => p).ToList().ForEach(p => Console.WriteLine("{0}.{1}", p, Employee[p]));

//method 2

var result = Employee.Keys.ToList();

result.Sort();

foreach(var k in result)

{

Console.WriteLine("{0}.{1}",k,Employee[k]);

}

//employee nmaes whose last but one character is a

//9.avathar

//2.sai

foreach(var p in Employee.Keys)

{

if(Employee[p].LastIndexOf('a')==Employee[p].Length-2)

{

Console.WriteLine("{0}.{1}", p, Employee[p]);

}

}

Console.ReadLine();

}

}

}

Fileeee

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

foreach(var line in File.ReadLines("D:\\Csharp\\hii.txt.txt"))

{

Console.WriteLine(line);

}

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

foreach(var directory in Directory.GetDirectories("D:\\"))

{

DirectoryInfo di = new DirectoryInfo(directory);

Console.WriteLine(di.Name+"\t\t"+di.CreationTime);

}

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

StreamReader obj = new StreamReader("D:\\Csharp\\hii.txt");

while(!obj.EndOfStream)

{

Console.Write((char)obj.Read());

}

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

//#region "StreamReader"

//StreamReader obj = new StreamReader("D:\\Csharp\\hii.txt");

//while(!obj.EndOfStream)

//{

// Console.Write((char)obj.Read());

//}

//#endregion

string name;

Console.WriteLine("enter your name");

name = Console.ReadLine();

StreamWriter sw = new StreamWriter("D:\\Csharp\\hii.txt", true);

sw.WriteLine(name);

Console.WriteLine("\n\n your name is saved");

sw.Close();//closing the stream

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

foreach(var filename in Directory.GetFiles("D:\\Csharp"))

{

FileInfo obj = new FileInfo(filename);

Console.WriteLine("{0}\t{1}\t{2}", obj.Name, obj.LastAccessTime, obj.Length);

}

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

class Program

{

static void Main(string[] args)

{

try

{

string path;

Console.WriteLine("enter directory path: ");

path = Console.ReadLine();

//write code to print list of all directories and file names + count

//to print directories

Console.WriteLine("LIST OF DIRECTORIES:\n");

foreach (var direc in Directory.GetDirectories(path))

{

Console.WriteLine((new DirectoryInfo(direc)).Name);

}

//to print files

Console.WriteLine("LIST OF FILES:\n");

foreach (var file in Directory.GetFiles(path))

{

Console.WriteLine(new FileInfo(file).Name);

}

//toprint count of directories and files

Console.WriteLine("Directory Count : {0}", Directory.GetDirectories(path).Count());

Console.WriteLine("File Count : {0}", Directory.GetFiles(path).Count());

}

catch(DirectoryNotFoundException Ex)

{

Console.WriteLine("enter valid path,it is a DirectoryNotFoundException ");

}

catch(Exception Ex)

{

Console.WriteLine("file not found");

}

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.Serialization.Formatters.Binary;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

[Serializable]

class Employee

{

public int id;

public string name;

public int salary;

}

class Program

{

static void Main(string[] args)

{

////SERIALIZATION

Employee emp = new Employee();

//emp.id = 10;

//emp.name = "abc";

//emp.salary = 10000;

//Stream s = File.Open("D:\\CSharp\\hii.txt", FileMode.Create, FileAccess.ReadWrite);

//BinaryFormatter b = new BinaryFormatter();

//b.Serialize(s, emp);

//s.Close();

//[Deserialization]

Stream s = File.Open("D:\\CSharp\\hii.txt", FileMode.Open, FileAccess.ReadWrite);

BinaryFormatter b = new BinaryFormatter();

emp = (Employee)b.Deserialize(s);

Console.WriteLine(emp.id);

Console.WriteLine(emp.name);

Console.WriteLine(emp.salary);

s.Close();

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.Serialization.Formatters.Binary;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

[Serializable]

class Employee

{

public int id;

public string name;

[NonSerialized]

public int salary;

}

class Program

{

static void Main(string[] args)

{

////SERIALIZATION

Employee emp = new Employee();

//emp.id = 10;

//emp.name = "abc";

//emp.salary = 10000;

//Stream s = File.Open("D:\\CSharp\\hii.txt", FileMode.Create, FileAccess.ReadWrite);

//BinaryFormatter b = new BinaryFormatter();

//b.Serialize(s, emp);

//s.Close();

//[Deserialization]

Stream s = File.Open("D:\\CSharp\\hii.txt", FileMode.Open, FileAccess.ReadWrite);

BinaryFormatter b = new BinaryFormatter();

emp = (Employee)b.Deserialize(s);

Console.WriteLine(emp.id);

Console.WriteLine(emp.name);

Console.WriteLine(emp.salary);

s.Close();

Console.ReadLine();

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.Serialization.Formatters.Binary;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApplication11

{

[Serializable]

class Employee

{

public int id;

public string name;

[NonSerialized]

public int salary;

}

class Program

{

static void Main(string[] args)

{

int i, j, rows, k = 1;

Console.Write("\n\n");

Console.Write("Display the pattern like right angle triangle with number increased by 1:\n");

Console.Write("---------------------------------------------------------------------------");

Console.Write("\n\n");

Console.Write("Input number of rows : ");

rows = Convert.ToInt32(Console.ReadLine());

for (i = 1; i <= rows; i++)

{

for (j = 1; j <= i; j++)

Console.Write("{0} ", k++);

Console.Write("\n");

}

Console.ReadLine();

}

}

}