#Multi Threading

Mutli-many ,threading=task

If our application involves complicated and time consuming operation then we use multithreading .

Using System;

Using System.threading;

T1{

For(int i=0;i<1000;i++){

}

}

T2{

For(int i=0;i<1000;i++){

}

}

Thread obj =Thread.CurrentThread;

obj.Name = "Hello this is main thread";

Console.WriteLine("this is main "+obj.Name);

Console.ReadLine();

Example

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

namespace threading

{

internal class Program

{

public static void threadchild()

{

for(int i = 0; i < 100; i++)

{

Console.WriteLine("thread child method = "+i);

}

}

public static void abc()

{

for (int i = 0; i < 100; i++)

{

Console.WriteLine("abc method = "+i);

}

}

static void Main(string[] args)

{

ThreadStart obj =new ThreadStart(threadchild);

ThreadStart objabc = new ThreadStart(abc);

Console.WriteLine("In Main thread");

Thread obj1 = new Thread(obj);

Thread obj2 = new Thread(objabc);

obj1.Start();

obj2.Start();

Console.ReadLine();

}

}

}

Example

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

namespace threading

{

internal class Program

{

public static void threadchild()

{

for(int i = 0; i < 10000; i++)

{

//Thread.Sleep(500);

Console.WriteLine("thread child method = "+i);

}

}

public static void abc()

{

for (int i = 0; i < 10000; i++)

{

Console.WriteLine("abc method = "+i);

}

}

static void Main(string[] args)

{

ThreadStart obj =new ThreadStart(threadchild);

ThreadStart objabc = new ThreadStart(abc);

Console.WriteLine("In Main thread");

Thread obj1 = new Thread(obj);

Thread obj2 = new Thread(objabc);

obj1.Start();

Thread.Sleep(2000);

obj2.Start();

Console.ReadLine();

}

}

}

Example-

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

namespace threading

{

internal class Program

{

public static void abc1()

{

for(int i = 0; i < 1000; i++)

{

//Thread.Sleep(500);

Console.WriteLine("abc1 = "+i);

}

}

public static void abc2()

{

for (int i = 0; i < 1000; i++)

{

//Thread.Sleep(500);

Console.WriteLine("abc2 = " + i);

}

}

public static void abc3()

{

for (int i = 0; i < 1000; i++)

{

Console.WriteLine("abc3 = "+i);

}

}

static void Main(string[] args)

{

ThreadStart objabc1 =new ThreadStart(abc1);

ThreadStart objabc2 = new ThreadStart(abc2);

ThreadStart objabc3 = new ThreadStart(abc3);

Console.WriteLine("In Main thread");

Thread obj1 = new Thread(objabc1);

Thread obj2 = new Thread(objabc2);

Thread obj3 = new Thread(objabc3);

obj1.Start();

obj2.Start();

obj3.Start();

obj2.Abort();

Console.ReadLine();

}

}

}

Example-

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

namespace threading

{

internal class Program

{

public static void abc3()

{

try

{

Console.WriteLine("abc3 thrad start");

int len = 10;

int[] arr = { 1, 2, 3, 4, 5 };

for (int i = 0; i <= 5; i++)

{

Thread.Sleep(500);

Console.WriteLine("abc3 = " + arr[i]);

}

}

catch (Exception ex)

{

Console.WriteLine("Thread abort bcz of expection" +ex);

}

}

public static void abc2()

{

for (int i = 0; i <= 5; i++)

{

//Thread.Sleep(500);

Console.WriteLine("abc2 = " +i);

}

}

static void Main(string[] args)

{

ThreadStart obj1 = new ThreadStart(abc3);

ThreadStart obj2 = new ThreadStart(abc2);

Thread objcall = new Thread(obj1);

Thread objcall1 = new Thread(obj2);

objcall.Start();

objcall1.Start();

Console.ReadLine();

}

}

}

Example

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

namespace threading

{

internal class Program

{

public static void abc()

{

Console.WriteLine("abc thread");

int sleepfor = 10000;

Console.WriteLine("abc thread is paused or sleep for "+sleepfor/1000);

Thread.Sleep(sleepfor);

Console.WriteLine("abc thread resume");

for(int i = 0; i < 10; i++)

{

Console.WriteLine(i);

}

}

static void Main(string[] args)

{

ThreadStart obj1 = new ThreadStart(abc);

Thread objcall = new Thread(obj1);

objcall.Start();

Console.ReadLine();

}

}

}

# Collection=>

Using System.Collections;

Designed to store, manage and manipulate similar data more efficiently.

Data manipulation includes adding,removing,finding and inserting data in the collection

Functionality:

1)adding and inserting items

2)removing item from collection

3) finding,sorting,searching items,

4)Replacing item

5)copy and clone

6)count

Types of collection:

Generic and non generic

1)generic🡺List,dictionary,sortedlist,stack,queue

2)Non-generic🡺ArrayList,hashtable, stack,queue

1)ArrayList=> array list is a colleaction .

We can store any type of data in arraylist/

Arraylist is of variable size.

In array list we can store any number element.

//syntax🡺

ArrayList obj= new ArrayList();

//adding data in arraylist

Obj.Add(12);

Obj.Add(“ram”);

Obj.Add(12.22);

Foreach(object x in obj ){

Console….

}

Obj.clear()//to clear evertiteam

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

internal class Program

{

static void Main(string[] args)

{

ArrayList obj = new ArrayList();

obj.Add(10);//0

obj.Add(10);//1

obj.Add("hello");//2

obj.Add("12.12");//3

obj.Add(10);

Console.WriteLine("before");

foreach (object o in obj)

{

Console.WriteLine(o);

}

Console.WriteLine("after");

//obj.Remove("Hello");

//obj.RemoveAt(4);

//obj.Insert(3,"hi");

obj.Reverse();

Console.WriteLine("capacity = "+obj.Count);

foreach (object o in obj)

{

Console.WriteLine(o);

}

Console.ReadLine();

}

}

}

#hash table🡺

Same like arrayList but in hash table we store element in key,value pair

Syntax🡺

Hashtable obj =new Hashtable();

Obj.Add(“1”,”ram”);

Obj.Add(“2”,”sham”);

Obj.Add(“html”,”hyper text …”);

Dictionary

(K,v)

For(DictionaryEntry d in obj){

Console.WriteLine(d.key+ “ ”+d.value);

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

internal class Program

{

static void Main(string[] args)

{

/\*

ArrayList obj = new ArrayList();

obj.Add(10);//0

obj.Add(10);//1

obj.Add("hello");//2

obj.Add("12.12");//3

obj.Add(10);

Console.WriteLine("before");

foreach (object o in obj)

{

Console.WriteLine(o);

}

Console.WriteLine("after");

//obj.Remove("Hello");

//obj.RemoveAt(4);

//obj.Insert(3,"hi");

obj.Reverse();

Console.WriteLine("capacity = "+obj.Count);

foreach (object o in obj)

{

Console.WriteLine(o);

}

\*

\*/

Hashtable ht = new Hashtable();

ht.Add(1, "ram");

ht.Add(2, "ram");

ht.Add(3, 122);

ht.Add(4, "visual basic");

foreach (DictionaryEntry o in ht)

{

Console.WriteLine(o.Key+ " "+ o.Value);

}

//ht.Remove(1);

//ht.Clear();

Hashtable obj=new Hashtable();

obj = (Hashtable)ht.Clone();

Console.WriteLine("------------------");

foreach (object o in obj.Keys)

{

Console.WriteLine(o + " " + obj[o]);

}

Console.ReadLine();

}

}

}

#stack

LIFO(Last in First out)

Syntax🡺

Stack obj = new Stack();

Push 🡺to add item

Pop🡺to remove item

Obj.Push(“hello”);

Obj.Push(11);

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

internal class Program

{

static void Main(string[] args)

{

Stack obj =new Stack();

obj.Push(1);

obj.Push(2);

obj.Push("hello");

foreach(object x in obj)

{

Console.WriteLine(x);

}

obj.Pop ();

obj.Clear ();

Console.WriteLine("count= "+ obj.Count);

Console.WriteLine("after=---------");

foreach (object x in obj)

{

Console.WriteLine(x);

}

Console.ReadLine();

}

}

}

#Queue🡺

Work on fifo (first in first out)

Syntax🡺

Queue obj=new Queue();

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

internal class Program

{

static void Main(string[] args)

{

Queue obj =new Queue();

obj.Enqueue(1);

obj.Enqueue("hello");

foreach(object x in obj)

{

Console.WriteLine(x);

}

obj.Dequeue();

bool isContain = obj.Contains("hi");

if (isContain)

{

Console.Write("hello is their");

}

else

{

Console.Write("hello is not their");

}

Console.WriteLine(obj.Contains("hello"));

//Console.WriteLine("count=);

Console.WriteLine("after=---------");

foreach (object x in obj)

{

Console.WriteLine(x);

}

Console.ReadLine();

}

}

}

2)Generic Collection

Array size is not fixed

We can add/remove item

Specific type

#List🡺

Syntax🡺

List< int> obj= new List<int>();

List<string> obj1= new List<int>();

Obj.Add(100);  
obj.Add(200);

Foreach (object x in obj){

Console.WriteLine(x);

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

internal class Program

{

static void Main(string[] args)

{

List<int> obj = new List<int>();

obj.Add(100);

obj.Add(200);

obj.Add(300);

obj.Add(200);

foreach (object x in obj){

Console.WriteLine(x);

}

Console.WriteLine("--------------------");

obj.Remove(200);

foreach (object x in obj)

{

Console.WriteLine(x);

}

Console.ReadLine();

}

}

}

Example

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

internal class Program

{

static void Main(string[] args)

{

/\*

List<int> obj = new List<int>();

Console.WriteLine("Enter the integer values in List ->");

for(int i = 0; i <= 5; i++)

{

obj.Add(Convert.ToInt32(Console.ReadLine()));

}

Console.WriteLine("--------------------------------------");

int count = 0;

foreach (object i in obj)

{

int x=Convert.ToInt32(i);

if (x % 2 == 0)

{

count++;

}

}

Console.WriteLine("Total number divisible by 2 ="+count);

\*/

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

list.Add("A");

list.Add("1");

list.Add("B");

list.Add("2");

//list.Remove("1");

foreach (object i in list)

{

Console.WriteLine(i);

}

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

{

if(list[i]=="1"|| list[i] == "2")

{

list.RemoveAt(i);

}

}

Console.WriteLine("--------------------------------------");

foreach (object i in list)

{

Console.WriteLine(i);

}

Console.ReadLine();

}

}

}

#dictionary🡺

(Key,Value)

Syntax 🡺

Dictionary <int,string> obj = new Dictionary<int , sting>();

Obj.Add(1,”hello”);

Obj.Add(2,”hi”);

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

internal class Program

{

static void Main(string[] args)

{

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

Obj.Add(1,"hello");

Obj.Add(2,"hi");

Console.WriteLine("--------------------------------------");

foreach(KeyValuePair<int, string> i in Obj)

{

Console.WriteLine(i.Key + " " + i.Value);

}

Console.ReadLine();

}

}

}

Example

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

internal class Program

{

static void Main(string[] args)

{

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

Obj.Add(1,"hello");

Obj.Add(2,"hi");

Console.Write("Pls enter the key which you want to remove = ");

int rem= Convert.ToInt32(Console.ReadLine());

foreach (KeyValuePair<int, string> i in Obj)

{

Console.WriteLine(i.Key + " " + i.Value);

}

Console.WriteLine("--------------------------------------");

foreach (KeyValuePair<int, string> i in Obj)

{

if(i.Key == rem)

{

Obj.Remove(rem);

}

else

{

Console.WriteLine("key not present");

break;

}

}

Console.WriteLine("--------------------------------------");

foreach(KeyValuePair<int, string> i in Obj)

{

Console.WriteLine(i.Key + " " + i.Value);

}

Console.ReadLine();

}

}

}

#sortedList==string==key

String==value

Syntax 🡺

SortedList<string,string> obj = new SortedList<string,string>();

Obj.Add(“HP”,”Himachal pradesh”);

Obj.Add(“UP”,”Uttar Pradesh”);

Obj.Remove(“HP”);

SortedList<string, string> obj = new SortedList<string, string>();

obj.Add("HP", "Himachal Pradesh");

obj.Add("UP", "uttar pradesh");

Console.WriteLine("--------------------------------------");

foreach(KeyValuePair<string, string> i in obj)

{

Console.WriteLine(i.Key + " " + i.Value);

}

Console.ReadLine();

Example🡺

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

internal class Program

{

static void Main(string[] args)

{

SortedList<string, string> obj = new SortedList<string, string>();

obj.Add("HP", "Himachal Pradesh");

obj.Add("UP", "uttar pradesh");

Console.WriteLine("--------------------------------------");

for(int i = 0; i < 5; i++)

{

Console.Write("Pls enter the key = ");

string skey = Console.ReadLine();

Console.Write("Pls enter the value = ");

string svalue = Console.ReadLine();

obj.Add(skey, svalue);

}

foreach (KeyValuePair<string, string> i in obj)

{

Console.WriteLine(i.Key + " " + i.Value);

}

Console.ReadLine();

}

}

}

Structure

Structure can have method,fields,property

We can’t make constructor.

Structure is value type.

It is used for low amount of data.

Structure we can’t inherit

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

using System.Collections;

namespace threading

{

struct abc

{

public int Id;

public string Name;

public string course;

public void display()

{

Console.WriteLine(Id);

Console.WriteLine(Name);

Console.WriteLine(course);

}

}

internal class Program

{

static void Main(string[] args)

{

abc obj = new abc();

obj.Id = 1;

obj.Name = "test";

obj.course = "Btech";

obj.display();

Console.ReadLine();

}

}

}