1. What is C#?

It is an object-oriented programming language created by Microsoft that runs on the .NET Framework.

2.Can we use keywords as an identifier? Why?

You cannot use keywords as identifiers; they are reserved for special use.

3.CREATE PROGRAM TO TAKE 5 STUDENTS DETAILS USING STRUCTURE.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace rakesh

{

class student

{

public int roll\_no;

public string name;

}

class Program

{

static void Main(string[] args)

{

List<student> s = new List<student>();

s.Add(new student { roll\_no = 1, name= "rakesh" });

s.Add(new student { roll\_no = 2, name = "jaydeep" });

s.Add(new student { roll\_no = 1, name = "jenish" });

s.Add(new student { roll\_no = 1, name = "akshay" });

s.Add(new student { roll\_no = 1, name = "shankar" });

foreach(var item in s)

{

Console.WriteLine("roll\_no ={0} name ={1}", item.roll\_no, item.name);

}

Console.Read();

}

}

}

4. Create a program to differentiate explicit and implicit conversation.

* **Implicit Conversion**

int i = 75;

long j = i;

* **Explicit conversions**

class Test

{

static void Main()

{

double x = 1234.7;

int a;

a = (int) x;

System.Console.WriteLine(a);

}

}

5. Create program to sort string in descending order

Namespace array

{

Class Program

{

Static void Main (string [] args)

{

int[] arr = new int[5];

Console.Write("enter the array:");

for(int i = 0; i < arr.Length; i++)

{

arr[i] = Convert.ToInt32(Console.ReadLine());

}

int temp;

for(int i = 0; i < arr.Length; i++)

{ for (int j = i + 1; j < arr.Length; j++)

{

if (arr[i] < arr[j])

{

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

Console.WriteLine("\n -----------short----------");

for (int i = 0; i < arr.Length; i++)

{

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

Console. Read ();

}

}

}

6. Explain any 5 string operation methods

-Clone(): It is used to return a reference to this instance of String.

-Concat(String, String): It is used to concatenate two specified instances of String.

-Copy(String): It is used to create a new instance of String with the same value as a specified String.

-Join(String, String[]): It is used to concatenate all the elements of a string array, using the specified separator between each element.

-Trim(): It is used to remove all leading and trailing white-space characters from the current String object.

7. Create program to take 2 numbers from user and show maximum number.

namespace max

{

class Program

{

static void Main(string[] args)

{

int a, b;

Console.Write("enter the two number:");

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

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

if (a > b)

{

Console.Write("max is :" + a);

}

else

{

Console. Write ("max is:" + b);

}

Console. Read ();

}

}

}

8.Write a program to perform below series and patterns

1+4+9+16+25.

namespace series1

{

class Program

{

Static void Main(string[] args)

{

int n;

Console. Write ("enter the number :");

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

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

{

Console. Write ("+" + (i \* i));

}

Console. Read ();

}

}

}

1+1+2+3+5+8+

Namespace series2

{

Class Program

{

Static void Main (string [] args)

{

int a, b, c, n;

Console. Write ("enter the value :");

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

a = 1;

b = 1;

Console. Write ("" + a);

Console. Write ("+" + b);

For (int i=2;i<n;i++)

{

c = a + b;

Console. Write ("+" + c);

a = b;

b = c;

}

Console. Read ()

}

}

1+5+14+30+55+

namespace series3

{

class Program

{

Static void Main(string[] args)

{

int n;

Console. Write ("enter the number :");

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

int r = 0;

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

{

int rev = r + (i \* i);

Console. Write ("+" + rev);

r = rev;

}

Console. Read ();

}

}

}

9. solve the following patterns :

1

2 3

4 5 6

7 8 9 10

Namespace p1

{

Class Program

{

Static void Main(string[] args)

{

int n,num=1;

Console.Write("enter the number:");

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

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

{

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

{

Console.Write("" + num );

num++;

}

Console.Write("\n");

}

Console.Read()

}

}

}

\*

# #

\*\* \*

####

\*\*\*\*\*

namespace p2

{

class Program

{

static void Main(string[] args)

{

int n;

Console.Write ("enter the number :");

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

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

{

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

{

if (i % 2 != 0)

{

Console.Write("\*");

}else

{

Console.Write("#");

}

}

Console.Write("\n”);

}

Console. Read ();

}

\*

#

# \*

# \* #

# \* # \* #

namespace p3

{

class Program

{

static void Main(string[] args)

{

int n;

Console.Write("enter the number:");

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

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

{

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

{

if (j % 2 != 0)

{

Console.Write("\*");

}else

{

Console.Write("#");

}

}

Console.Write("\n");

}

Console.Read();

}

}

}

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

namespace p4

{

class Program

{

static void Main(string[] args)

{

int n;

Console.Write("enter the number:");

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

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

{

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

{

Console.Write("\*");

}

Console.Write("\n");

}

Console.Read();

}

}

}

1

22

333

4444

55555

namespace p6

{

class Program

{

static void Main(string[] args)

{

int n;

Console.Write("enter the number:");

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

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

{

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

{

Console.Write("" + i );

}

Console.Write("\n");

}

Console.Read();

}

}

}

1

12

123

1234

12345

namespace p7

{

class Program

{

static void Main(string[] args)

{

int n;

Console.Write("enter the number:");

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

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

{

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

{

Console.Write("" + j);

}

Console.Write("\n");

}

Console.Read();

}

}

}

10.

a loop variable is a variable that is set in order to execute some iterations of a "for" loop or other live structure

11.

An iterator, in the context of C#, is a block of code that returns an ordered sequence of values of a collection or array. It is a member function implemented using the iterator block, which contains one or more statements containing the "yield" keyword.

12.

An array is a collection of similar data elements stored at contiguous memory locations. It is the simplest data structure where each data element can be accessed directly by only using its index number.

13.

A jagged array is an array whose elements are arrays, possibly of different sizes. A jagged array is sometimes called an "array of arrays." The following examples show how to declare, initialize, and access jagged arrays.

example: int[][] jaggedArray = new int[3][];

14. What is jagged array? Explain with example?

A jagged array is an array whose elements are arrays, possibly of different sizes. A jagged array is sometimes called an "array of arrays.".

A jagged array is an array of array. Jagged arrays store arrays instead of literal values.

A jagged array is initialized with two square brackets [][]. The first bracket specifies the size of an array, and the second bracket specifies the dimensions of the array which is going to be stored.

The following example declares jagged arrays.

15. Create program to iterate string variable using foreach loop ?

namespace forloop

{

class Program

{

static void Main(string[] args)

{

ArrayList li = new ArrayList();

li.Add("chita");

li.Add(20000);

li.Add('m');

foreach(var item in li)

{

Console.WriteLine(item);

}

Console. Read ();

} } }

16. Write a program to call class method?

namespace class

{

class student

{

public string name;

public int rollno;

internal void get()

{

name = "chita";

rollno = 17;

}

internal void show()

{

Console.WriteLine("student name:" + name);

Console.WriteLine("student rollno:" + rollno);

}

}

class Program

{

static void Main(string[] args)

{

student s = new student();

s.get();

s.show();

Console.Read();

} } }

17. Write a program to calculate arithmetic operations using class and object.

namespace calculator

{

class arithmatic

{

internal int a, b, c;

internal void get()

{

Console.Write("enter the number:");

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

Console.Write("enter the number:");

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

}

internal void chi()

{

c = a + b;

Console.WriteLine("addition of two number: " + c);

c = a - b;

Console.WriteLine("substraction of two number : " + c);

c = a \* b;

Console.WriteLine("multification of two number: " + c);

c = a / b;

Console.WriteLine("division of two number : " + c);

}

}

class Program

{

static void Main(string[] args)

{

arithmatic a = new arithmatic();

a.get();

a.chi();

Console.Read();

}

}

18. Write a program to call method of parent class?

Namespace classs

{

Class student

{

Protected string name;

Protected int age;

}

Class test : student

{

Int mark;

Internal void get ()

{

Name = "sankar";

age = ;

mark = 500;

}

internal void show()

{

Console.WriteLine("student name:" + name);

Console.WriteLine("student age:" + age);

Console.WriteLine("student mark:" + mark);

}

}

Class Program

{

static void Main(string[] args)

{

test t = new test();

t.get();

t.show();

Console. Read();

}

}

}

19. Write a program to get three subject marks details and then show average and sum.

namespace results

{

class student

{

internal int sub1;

internal int sub2;

internal int sub3;

internal void get()

{

sub1 = 23;

sub2 = 45;

sub3 = 24;

}

internal void show()

{

int tol = sub1 + sub2 + sub3;

int avg = tol / 3;

Console.WriteLine("totalmark :" + tol);

Console.WriteLine("avgmark:" + avg);

}

}

class Program

{

static void Main(string[] args)

{

student s = new student();

s.get();

s.show();

Console.Read();

}

}

}

20. Write a program to calculate factorial number of user defined value using class.

namespace fact

{

class factorial

{

int n;

Internal void get ()

{

Console.Write ("enter the number :");

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

}

Internal void show ()

{

int fact = 1;

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

{

fact = fact \* i;

}

Console.Write ("factorial is:" + fact);

}

}

Class Program

{

Static void Main (string[] args)

{

Factorial f = new factorial ();

f.get();

f.show();

Console.Read();

}

}

}

21. Write a program to perform an example of data hiding.

namespace datas

{

abstract class student

{

virtual internal void display()

{

}

}

class result:student

{

override internal void display()

{

Console.WriteLine("hello");

}

}

class Program

{

Static void Main(string[] args)

{

Student s = new result ();

s.display();

Console. Read ();

}

}

}

22. How can we manage runtime errors?

Runtime errors occur during execution of the program that is why we also called runtime exceptions. If you does not create logic in proper way, then it create runtime exception.

Using exception handling, we can handle exception in a proper way and show the accurate result as per user understanding. It is a mechanism to detect and handle run time errors. We can achieve this using Try-Catch-Finally blocks.

23. What is abstract class?

An abstract class is a special type of class that cannot be instantiated. An abstract class is designed to be inherited by subclasses that either implement or override its methods.

24. What is thread?

A thread is a path of execution within a process. A process can contain multiple threads.

25 What is dll?

Dynamic Link Library (DLL) is Microsoft's implementation of the shared library concept. A DLL file contains code and data that can be used by multiple programs at the same time, hence it promotes code reuse and modularization. This brief tutorial provides an overview of Windows DLL along with its usage.

26. What is namespace?

A namespace is designed for providing a way to keep one set of names separate from another. The class names declared in one namespace does not conflict with the same class names declared in another.

27.create program to replace specific word from string.

using System;

public class Demo {

   public static void Main() {

      string str = "Demo text!";

      Console.WriteLine(str);

      string res = str.Replace("Demo ", "New ");

      Console.WriteLine("After replacing...");

      Console.WriteLine(res);

   }

28. Create program to take 3 numbers from user and show maximum and minimum number?

Namespace I2

{

Class Program

{

static void Main(string[] args)

{

int a, b, c;

Console.WriteLine ("enter the 1st number :");

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

Console.WriteLine ("enter the 2nd number :");

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

Console.WriteLine ("enter the 3rd number :");

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

if (a > b)

{

if(a>c)

{

Console.WriteLine (" max is:" + a);

}

}

Else if (b > c)

{

Console.WriteLine ("max is:" + b);

}

Else

{

Console.WriteLine ("max is:"+c);

}

if (a < b)

{

if (a < c)

{

Console.WriteLine (" small is:" + a);

}

}

Else if (b < c)

{

Console.WriteLine ("small is:" + b);

}

Else

{

Console.WriteLine ("small is:" + c);

}

Console.Read ();

}

}

}

29. What is difference between else if ladder and switch case?.

A In else if ladder, the control goes through the every else if statement until it finds true value of the statement or it comes to the end of the else if ladder. In case of switch case, as per the value of the switch, the control jumps to the corresponding case.

The switch case is more compact than lot of nested else if. So, switch is considered to be more readable

The use of break statement in switch is essential but there is no need of use of break in else if ladder.

The variable data type that can be used in expression of switch is integer only where as in else if ladder accepts integer type as well as character.

Switch case statement work on the basis of equality operator whereas else if ladder works on the basis of true false( zero/non-zero) basis.

30. What will occur if we not write break statement in switch case ?

If we do not use break statement at the end of each case, program will execute all consecutive case statements until it finds next break statement or till the end of switch case block.

31. What is difference between entry loop and exit loop? Explain with example?

The Key Difference Between Entry Control and Exit Control Loop is that in Entry Control Loop the test condition is checked first and if that condition is true then the block of the statement will be executed, While in Exit control loop first executes the body of the loop and checks condition at last. Example entry control loop while loop , exit control loop do while loop.

32. What do you mean by multi-dimention array?

A multidimensional array in MATLAB® is an array with more than two dimensions. In a matrix, the two dimensions are represented by rows and columns. ... Multidimensional arrays are an extension of 2-D matrices and use additional subscripts for indexing. A 3-D array, for example, uses three subscripts.

33. Explain 5 method of array class with example?

indexed arrays,

multidimensional arrays,

associative arrays,

34. Get 5 values from user and store in array and show all elements and sum of elements?

namespace I9

{

class Program

{

static void Main(string[] args)

{

int[] arr = new int[5];

Console.Write("enter the element :");

for(int i = 0; i < arr.Length; i++)

{

arr[i] = Convert.ToInt32(Console.ReadLine());

}

Console.WriteLine("show element:");

for(int i = 0; i < arr.Length; i++)

{

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

}

int sum = 0;

for(int i = 0; i < arr.Length; i++)

{

sum = sum + arr[i];

}

Console.WriteLine("sum of element:" + sum);

Console.Read();

}

}

}

35. Create program to make transform of two matrices

namespace ConsoleApplication20

{

class Program

static void Main(string[] args)

{

int i ,j;

int[,] name = new int[2, 3];

Console.WriteLine("1st matrix :");

for(i=0;i<2;i++)

{

for(j=0;j<3;j++)

{

name[i, j] = Convert.ToInt32(Console.ReadLine());

}

}

//1st matrix output //

Console.WriteLine("1st magtrix output shown below ");

Console.WriteLine();

for (i = 0; i < 2; i++)

{

for (j = 0; j < 3; j++)

{

Console.Write(" " + name[i, j]);

}

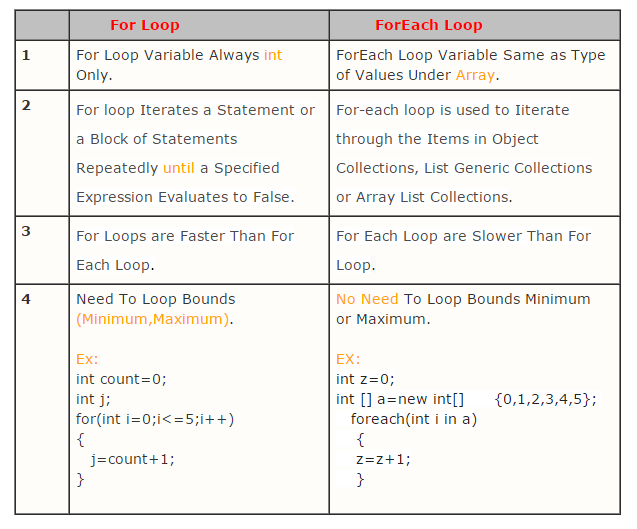
Console.WriteLine();

}

}

}

36. difference between for loop and foreach loop?



37. Write a program to call static method using class.

namespace I12

{

static class student

{

public static string name;

public static int rollno;

}

class Program

{

static void Main(string[] args)

{

student s = new student();

student.name = "sankar";

student.rollno = 17;

Console.Read(); }

} }

38. Write a program to implement multilevel inheritance.

namespace inheritance

{

class student

{

protected int rollno;

}

class test:student

{

protected int sub1;

protected int sub2;

}

class result :test

{

internal int totalmark;

internal void getdata()

{

rollno = 1;

sub1 = 57;

sub2 = 23;

totalmark = sub1 + sub2;

}

internal void show()

{

Console.WriteLine("student rollno:" + rollno);

Console.WriteLine("student sub1mark:" + sub1);

Console.WriteLine("student sub2mark::" + sub2);

Console.WriteLine("student totalmark:" +totalmark);

}

}

class Program

{

static void Main(string[] args)

{

result r = new result();

r.getdata();

r.show();

Console.Read();

}

} }

40. write a program to implement multiple inheritance

**class** ParentA

  {

**public** **void** MethodA()

      {

          Console.WriteLine("MethodA from ParentA called");

      }

**public** **void** MethodB()

      {

          Console.WriteLine("MethodB from ParentA called");

      }

  }

**class** ParentB

  {

**public** **void** MethodA()

      {

          Console.WriteLine("MethodA from ParentB called");

      }

**public** **void** MethodB()

      {

          Console.WriteLine("MethodB from ParentB called");

      }

  }

**class** ParentWrapper

  {

      ParentA objA = **new** ParentA();

      ParentB objB = **new** ParentB();

**public** **void** ParentWrapperAMethodA()

      {

          objA.MethodA();

      }

**public** **void** ParentWrapperAMethodB()

      {

          objA.MethodB();

      }

**public** **void** ParentWrapperBMethodA()

      {

          objB.MethodA();

      }

**public** **void** ParentWrapperBMethodB()

      {

          objB.MethodB();

      }

  }

**class** Child : ParentWrapper

  {

  }

Child objChild = **new** Child();

objChild.ParentWrapperAMethodA();

objChild.ParentWrapperBMethodB();

41. Write a program to redefine method logic in child class. (Overloading)

namespace I16

{

class student

{

virtual internal void display()

{

}

}

class result : student

{

internal override void display()

{

Console.WriteLine("hello");

}

}

class Program

{

static void Main(string[] args)

{

student s = new result();

s.display();

Console.WriteLine();

}

}

}

42. Write a program to access private variables outside of class.

namespace arraycomman

{

class privatevariable

{

private int a = 10;

public void display()

{

Console.WriteLine("private variable :{0}", a);

}

}

class Program

{

static void Main(string[] args)

{

privatevariable a = new privatevariable();

a.display();

Console.ReadKey();

}

} }

43. Write a program to divide class definition into multiple parts. (Partial Class)

partial class a

{

public int c=10;

public void display()

{

Console.WriteLine("\n\tprivate variable :{0}", c);

}

}

partial class a

{

public void view()

{

Console.WriteLine("\thello world");

}

}

partial class a

{

public void show()

{

Console.WriteLine("\tshow me ");

}

}

class Program

{

static void Main(string[] args)

{

a b = new a();

b.display();

b.view();

b.show();

Console.ReadKey();

}

}

}

44. What is dictionary? Advantages of Dictionary?

In C#, Dictionary is a generic collection which is generally used to store key/value pairs. The working of Dictionary is quite similar to the non-generic has table. The advantage of Dictionary is, it is generic type. Dictionary is defined under System

45. What is multithread?

Multithreading is a feature provided by the operating system that enables your application to have more than one execution path at the same time. Technically, multithreaded programming requires a multitasking operating system.

46. How to prevent class to be instantiate?

Those ways are:

1. Abstract
2. Static Class
3. Private and protected constructor.

47 Create program to replace specific character from string

namespace arraycomman

{

class Program

{

static void Main(string[] args)

{

String str = "hello world ";

Console.WriteLine("OldString : " + str);

// replace the character 's' with 'G'

Console.WriteLine("NewString: " + str.Replace('e', '$'));

Console.ReadKey();

}

}

}

48. What is mutable and immutable string?

String Builder is used to represent a mutable string of characters. Mutable means the string which can be changed. So String objects are immutable but String Builder is the mutable string type. It will not create a new modified instance of the current string object but do the modifications in the existing string object. The complete functionality of String Builder is provided by String Builder class which is present in System. Text namespace.

49. Write a program to find factorial of user defined number

Namespace a3

{

Class Program

{

Static void Main(string[] args)

{

int a, fact=1;

Console.WriteLine ("enter the number :");

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

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

{

Fact = fact \* i;

}

Console.WriteLine ("factorial is "+fact);

Console. Read ();

}

}

}

50. Write a program to sort a numeric array without using array methods.

namespace array

{

class Program

{

static void Main(string[] args)

{

int[] arr = new int[5];

Console.Write("enter the array:");

for(int i = 0; i < arr.Length; i++)

{

arr[i] = Convert.ToInt32(Console.ReadLine());

}

int temp;

for(int i = 0; i < arr.Length; i++)

{ for (int j = i + 1; j < arr.Length; j++)

{

if (arr[i] < arr[j])

{

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

Console.WriteLine("\n -----------short----------");

for(int i = 0; i < arr.Length; i++)

{

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

}

Console.Read();

}

}

}

51. What is difference between array and list?

ARRAY

* An Array is strongly-typed. We can store only the same type of data.
* Array stores a fixed number of elements.
* Array provides better performance than Array List.
* Array belongs to namespace System.
* The Array cannot accept null.

LIST

* Array List is a non-generic collection type. Array List's internal Array is of the object type. So, we can store multiple types of data in Array List.
* Array List is dynamic in term of capacity. If the number of element exceeds, Array List will increase to double its current size.
* If we are using a large number of Array List then it degrades performance because of boxing and unboxing.
* Array List belongs to namespace System.Collection.
* An Array can accept null.

52. Create one collection where can be store all data types by indexing?

namespace dictonary

{

class Program

{

static void Main(string[] args)

{

Dictionary<string, object> d = new Dictionary<string, object>();

d.Add("id", 10);

d.Add("name", "sankar");

d.Add("roll no:", 25);

d.Add("std", 12);

foreach(var iteam in d)

{

Console.WriteLine(iteam.Key + ":" + iteam.Value);

}

Console.Read();

}

}

53. Get 5 values from user and show maximum value from array

Namespace I2

{

Class Program

{

Static void Main (string [] args)

{

int a, b, c;

Console.WriteLine("enter the 1st number:");

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

Console.WriteLine("enter the 2nd number :");

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

Console.WriteLine("enter the 3rd number:");

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

if (a > b)

{

if(a>c)

{

Console.WriteLine(" max is :" + a);

}

}

else if (b > c)

{

Console.WriteLine("max is :" + b);

}

else

{

Console.WriteLine ("max is:"+c);

}

Console.Read ();

}

}

}

54. Write a program to define base class with fixed method signature and implement them into derived class. (Abstract Class)

namespace ConsoleApplication13

{

abstract class mcn

{

public int add(int a, int b)

{

return (a + b);

}

}

class mcn1 : mcn

{

public int mul(int a, int b)

{

return a \* b;

}

}

class test

{

static void Main(string[] args)

{

mcn1 ob = new mcn1();

int result = ob.add(5, 10);

Console.WriteLine("the result is {0}", result);

Console.Read();

}

}

}

55. Write a program of abstract class with implemented methods and declared methods.

namespace @abstract

{

public abstract class hello

{

public void display()

{

Console.WriteLine("hello world");

}

}

class hi:hello

{

public void view()

{

Console.WriteLine(" how are you");

}

}

Class Program

{

static void Main(string[] args)

{

hi h = new hi();

h.view();

h.display ();

Console.Read ();

}

}

}

57. Write a program to Get 10 random numbers using parent class's constructor and implement one method in derived class to show them in ascending manner.

namespace ConsoleApplication13

{

class parent

{

public int i;

public void display()

{

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

{

Console.WriteLine(+i);

}

}

}

class child:parent

{

public int j;

public void view()

{

for(j=0;j<10;j++)

{

Console.WriteLine("\n\t{0}",j \* j);

}

}

}

class test

{

static void Main(string[] args)

{

child c = new child();

Console.WriteLine("\n\t10 values in asscending order");

c.view();

Console.Read();

}

}

}

58. Write a program to prevent class from being instantiated.

namespace ConsoleApplication13

{

sealed class SealedClass

{

public int Add(int a, int b)

{

return a + b;

}

}

class Program

{

static void Main(string[] args)

{

SealedClass slc = new SealedClass();

int total = slc.Add(6, 4);

Console.WriteLine("Total = " + total.ToString());

Console.Read();

}

}

}

59 .Write a program to implement virtual method in derived class.

namespace ConsoleApplication13

{

class math

{

public int addition(int a, int b)

{

return a + b;

}

}

class SealedClass:math

{

public virtual int Add(int a, int b)

{

return a + b;

}

public virtual int mull(int a , int b )

{

return a \* b;

}

}

class Program

{

static void Main(string[] args)

{

SealedClass slc = new SealedClass();

int totaladd = slc.Add(6, 4);

Console.WriteLine("Total = " + totaladd.ToString());

int totalmull = slc.mull(10, 5);

Console.WriteLine("Total = " + totalmull.ToString());

Console.Read();

}

}

}

61. Create a program to store student’s details using Dictionary.

namespace dictonary

{

class Program

{

static void Main(string[] args)

{

Dictionary<string, object> d = new Dictionary<string, object>();

d.Add("id", 10);

d.Add("name", "sankar");

d.Add("roll no:", 25);

d.Add("std", 12);

foreach (var iteam in d)

{

Console.WriteLine(iteam.Key + ":" + iteam.Value);

}

Console.Read();

}

}

}

62. What is generics? Explain with example

Generic means the general form, not specific. In C#,

C# allows you to define generic classes, operators using the [type parameter](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/generics/generic-type-parameters) and without the specific data type. A type parameter is a placeholder for a particular type specified when creating an instance of the generic type.

A generic type is declared by specifying a type parameter in an angle brackets after a type name, e.g. TypeName<T> where T is a type parameter.

namespace generictype1

{

class hello<type1, type2 >

{

type1 id;

type2 name;

public void display(type1 name, type2 id)

{

Console.WriteLine("name :{0}", name);

Console.WriteLine("id :{0}", id);

}

}

class Program

{

static void Main(string[] args)

{

hello<string, int> my = new hello<string, int>();

my.display("sankar", 95);

my.display("sagar", 85);

hello<int, int> me = new hello<int, int>();

my.display("95", 25);

Console.Read();

}

}

}

63. What is use of method overriding?

If derived class defines same method as defined in its base class, it is known as method overriding in C#. It is used to achieve runtime polymorphism. It enables you to provide specific implementation of the method which is already provided by its base class.