what is Delegate

→ A delegate is a variable that holds the reference to a method or pointer to a method.

→ A delegate can refer to more than one methods of same return type and parameter

when to use delegate?

→ when we need to pass a method as a parameter.

Example:-

```
delegate void calculator (int x, int y)
class program
 {
 public static void Add (int a, int b)
  {
   console.writeLine (a+b);
  }
public static void ~~Main~~ Mult (int a, int b)
  {
   console.writeLine (a*b);
  }
static void Main (string [] args)
  {
  calculator cal = new calculator (Add);
     cal (20, 30); // calling method using delegate
  }
       calc += Mult; (Multicast delegate)

       //output: 50 600
```

what are multicast Delegate?

→ A multicast delegate is a delegate that holds the references of more than one function.

what are Anonymous Delegate?

→ Delegate pointing methods without name are called anoymous delegates.

```
public delegate void calculator (int x, int y);
class program
{
static void Main (string [] args)
{
calculator calcAdd = delegate (int a, int b)
 {
  // Inline content of the method
   console.writeLine (a+b);
  };
  calcAdd (20,30);
 }
}
      //output : 50
```

Func Delegate

→ It is a generic delegate in the System namespace. It has zero or more input parameter and one out parameter. The last parameter is considired as an out parametir

For example:

```
public delegate TResult Func<in T, out TResult>(T args);
```

Type of return value

Type of first input parameter

A func delegate with two input parameters and one out parameter

Type of return value

```
public delegate TResult Func <in T1, in T2, out TResult>
                        (T1 args1, T2 args2);
```

Type of first input parameter

Type of second input parameter

- It can take up to 16 input parameters of any type and has a generic type parameter that represents the return type.

- Func delegate are commonly used when you need to pass a method as a parameter with a return value.

```
class Program
{
static int Sum (int x, int y)
{
return x+y;
}
static void Main (string [] args)
{
Func <int, int, int >add =sum;
   int result = add (10,10);
   console.writeLine ( result);
}
```

Action Delegate

→ It is a predefined delegate type in C# that represents a method with no return value. It can take up to 16 input parameters of any type. Action delegate are commonly used when you need to pass a method as a parameter without a return value.

```
Action <int, int> add = (a, b) => {
     console.writeLine (a +b);
  };
  add (5,3);
```

Predicate Delegate

The predicate dolg delegate is a specialized version of the Func delegate that represents a method that takes a single input parameter and returns a boolean value. It is commonly used for filtering or testing conditions.

```
static bool IsupperCase (String str)
{
  return str.Equals(str.ToUpper ());
}
static void Main (String[] args)
{
  Predicate<String> isupper = IsupperCase;
  bool result = isupper ("hello world");
  console.writeLine(result); //false
 }
```

Lambda Expressions

→ They are a shorthand syntax for defining anonymous methods. They provide a more concise and readable way to write code.

→ They are commonly used with delegates and LINQ queries

Expression Lambda ⇒ input ⇒ expression;

Statement Lambda input ⇒ { statement };