What is Polymorphism in C#?

Polymorphism is a OOPs concept where one name can have many forms.

For example, you have a smartphone for communication. The communication mode you choose could be anything. It can be a call, a text message, a picture message, mail, etc. So, the goal is common that is communication, but their approach is different. This is called Polymorphism.

Contoh Polymorphism dengan fungsi logic :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ConsoleApplication11
{
    public class BeratBadanIDeal
    {
        public static void Main(string[] args)
        {
            BeratBadan bb = new BeratBadan();
            try
            {
                bb.showBeratBadan();
            }
            catch (TidakIdealException e)
            {
                Console.WriteLine("TidakIdealException: {0}", e.Message);
            }
            Console.ReadLine();
        }
    }
}

public class TidakIdealException : Exception
{
    public TidakIdealException(string message)
        : base(message)
    {

    }
}

public class BeratBadan
{
    float beratbadan = 55;
    float tinggibadan = 170;
    public void showBeratBadan()
    {
        float ideal = ((tinggibadan - 100) - ((tinggibadan - 100) / 100));
        if (beratbadan != ideal)
        {
```

```
            throw (new TidakIdealException("Berat Badan Anda Tidak Ideal,
Berat badan ideal anda adalah :" + ideal));
        }
        else
        {
            Console.WriteLine("Berat badan anda ideal: {0}", beratbadan);
        }
    }
}
```

Polimorphism dengan Konsep Overloading

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ConsoleApplication11
{
  // Polimorphism dengan konsep Overloading
  // Overloading : adalah pemakaian beberapa methods ataupun properties
dengan nama yang sama,
  // tetapi memiliki daftar parameter/argument yang berbeda. Perbedaan yang
dimaksud adalah beda
  // jumlah parameter, beda tipe data, atau beda keduanya (jumlah parameter
dan tipe data).
  // Methods ataupun properties yang hanya beda return value (nilai balik)
tidak bisa dikatakan
  // sebagai overloading.

  public class KelilingSegi
    {
        public double sisiA;
        public double sisiB;
        public double sisiC;

        //Constuctor satu
        public void Keliling (double a){
        this.sisiA=a;
        this.sisiB=1;
        this.sisiC=1;

        }
        //Constuctor dua
        public void Keliling (double a, double b){
        this.sisiA=a;
        this.sisiB=b;
        this.sisiC=1;
        }
        //Constuctor tiga
        public void Keliling (double a, double b, double c){
        this.sisiA=a;
        this.sisiB=b;
```

```
        this.sisiC=c;
        }
    }
 class Program
 {
     public static void Main(string[] args)
     {
         KelilingSegi tampil = new KelilingSegi();
         KelilingSegi tampil_dua = new KelilingSegi();
         KelilingSegi tampil_tiga = new KelilingSegi();

         Console.WriteLine("Tampil Constructor Pertama");
         tampil.Keliling(10);
         Console.WriteLine("Sisi A = "+tampil.sisiA);
         Console.WriteLine("Sisi B = "+tampil.sisiB);
         Console.WriteLine("Sisi C = "+tampil.sisiC);

         Console.WriteLine("Tampil Constructor Kedua");
         tampil_dua.Keliling(10, 11);
         Console.WriteLine("Sisi A = "+tampil_dua.sisiA);
         Console.WriteLine("Sisi B = "+tampil_dua.sisiB);
         Console.WriteLine("Sisi C = "+tampil_dua.sisiC);

         Console.WriteLine("Tampil Constructor Ketiga");
         tampil_tiga.Keliling(10, 11, 12);
         Console.WriteLine("Sisi A = "+tampil_tiga.sisiA);
         Console.WriteLine("Sisi B = "+tampil_tiga.sisiB);
         Console.WriteLine("Sisi C = "+tampil_tiga.sisiC);

         Console.Write("Press any key to continue . . . ");
         Console.ReadKey(true);
     }
 }
}
```


Polimorphism dengan Konsep Overriding

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ConsoleApplication11
{
    // Polimorphism dengan konsep Orverriding
```

```
    // Overriding : kemampuan class turunan untuk memodifikasi methods atau
properties dari class induk.
    // Dengan overriding kita bisa membuat implementasi baru pada methods
atau properties di class
    // turunan yang berbeda dengan methods atau properties yang ada pada
class induk.

    class first
    {
        public void show()
        {
            Console.WriteLine("Keterangan ini ditampilkan oleh class first");
        }
    }

    class second : first
    {
        public void show1()
        {
            Console.WriteLine("Keterangan ini ditampilkan oleh class dua");
        }
    }

    class Program
    {
        public static void Main(string[] args)
        {
            first satu = new first();
            satu.show();
            second dua = new second();
            dua.show1();
            Console.Write("Press any key to continue . . . ");
            Console.ReadKey(true);
        }
    }
}
```