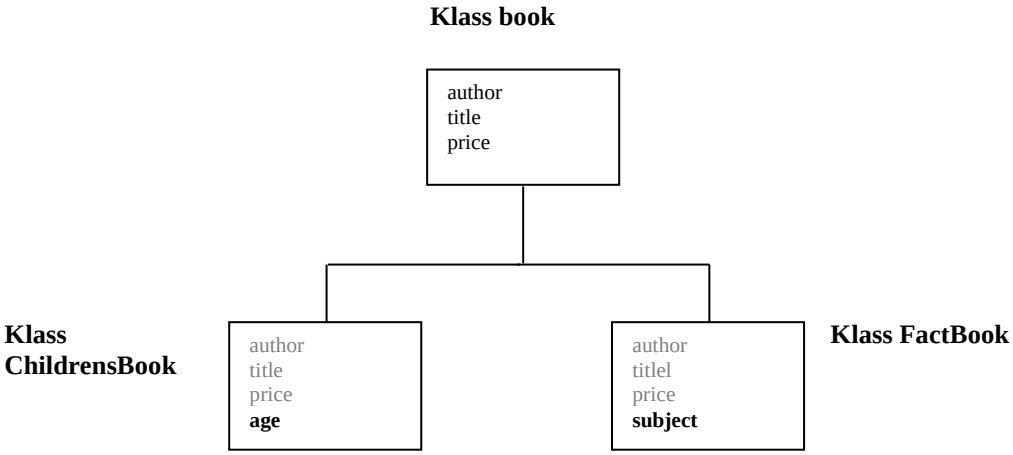
Objektorientering del 3 – Arv

När man gör en ny klass behöver man inte alltid börja från början. Genom arv kan man ärva funktionalitet som finns i en existerande klass.

Ibland säger man föräldraklass eller basklass och barnklass.

Ex.



Eftersom barnbok och faktabok ärver från bok så har de samma funktionalitet. Man kan säga att en barnbok eller faktabok också **är** en bok.

Man kan sedan lägga till funktionalitet i de ärvda klasserna, som ålder för barnbok eller ämne för faktabok.

För att ärva skriver man ett kolon efter klassnamnet och sedan klassen man vill ärva från.

Så här: **class ChildrensBook : book**

Uppgift:

Skapa en ny consol-applikation eller använd den uppgiften när du gjorde
Objektorientering del 2 – konstruktorer.

Om du skapar ett nytt projekt gör du så här:

- Skapa en ny klass och lägg in klassen **book** (Se sida 3).
  - Använder du övningen med konstruktorer har du redan den klassen.
- Sedan skapar du ytterligare en klass. Döp den till **ChildrensBook**. (Se sida 4)
  - Denna ärver från klassen **book.**
    Vi talar om att vi vill ärva från b**ook** genom att sätta kolon (**:**)

```
class ChildrensBook : book
```

- Eftersom vi ärver behöver vi inte deklarera Titel, författare och pris. Vi kan ändå använda dom.
- Ska vi bara skriva ut Titel, författare och pris behöver vi inte göra en ny skrivUt-metod i ChildrensBook. Vi ärver från book och kan använda den.

För att skapa ett objekt av vår nya klass BarnBok skriver vi:

ChildrensBook enBarnBok = new **ChildrensBook**();

Vi kan nu sätta Titel, Författare och Pris trots att vi **inte** har deklarerat dom i ChildrensBook.

```
enBarBok.Title = "Emil i Lönneberga";
enBarnBok.Author = "Astrid Lindgren";
enBarnBok.Price = 300;
```

Vi kan sedan även skriva ut värdena fast vi inte har en skrivUt()-metod.

enBarnBok.SkrivUt();

Testa!

```
class book
{
        private string titel;
        private string author;
        private int price;


        //egenskaper (properties). Används för att förhindra åtkomst till privata variabler.
        public string Titel
        {
            get
            {
                return titel;
            }
            set
            {
                titel = value;
            }
        }

        public string Author
        {
            get
            {
                return author;
            }
            set
            {
                author = value;
            }
        }

        public int Price
        {
            get
            {
                return price;
            }
            set
            {
                price = value;
            }
        }


      //Konstruktor (constructor) Metod som körs när objektet skapas. Denna körs om du har tom parantes
      //Har man konstruktor med parametrar måste man skapa en som är tom också.
        public book()
        {
        }

        //Konstruktor med parametrar
        public book(string titel, string author, int price)
        {
            Author = author;
            Titel = titel;
            Price = price;
        }

        //Metod som skriver ut våra variabler.
        public void skrivUt()
        {
            Console.WriteLine("Författare: {0}", Author);
            Console.WriteLine("Titel: {0}", Titel);
            Console.WriteLine("Pris: {0}", Price);
            Console.WriteLine("");
        }

    }
```

```
class ChildrensBook : book
{
        private string age;

        public string Age
        {
            get
            {
                return age;
            }
            set
            {
                age = value;
            }
        }

        //Konstruktor (constructor) Metod som körs när objektet skapas. Denna är tom.
        public ChildrensBook ()
        {
        }

        public ChildrensBook (string titel, string author, int price, string age)
        {
            Author = author;
            Titel = titel;
            Price = price;
            Age = age;
        }

        public override void skrivUt()
        {
            base.skrivUt();
            Console.WriteLine("Ålder: {0}", Age);
            Console.WriteLine("");
        }
    }
}
```

Vitual och Override

Har man en metod som vår **skrivUt()** och vill använda den för att skriva ut allt som tillhör en bok men även det nya i barnklassen måste man använda två nyckelord: **virtual** och **override.**

Virtual skriver man i metoden som man har i basklassen och **override** skriver man i den metod man har i barnklassen.

Så här:

Basklass:

```
public virtual void skrivUt();
{
        skrivUt(); //Original-metod
}
```

Barnklass:

```
public override void skrivUt()
{
        base.skrivUt(); //Detta anropar skrivUt() i basklass och skriver ut det som alla böcker har gemensamt.

       //Vi kan nu lägga till det som är unikt för barnklassen:

        Console.WriteLine("Ämne: {0}", Subject);
}
```

Uppgift:

Ändra i klasserna enligt ovan.

Skapa ett objekt av barnklassen med sin konstruktor som tar 4 värden. **Titel**, **författare**, **pris** och det unika för barnbok nämligen vilken **ålder** som barnboken passar till.

Skriv sedan ut värdena med skrivUt-metoden.

Mera uppgifter

1. När ni sätter värdet i efterhand skriver ni t.ex. HistoryBook.Titel. Vilken del av er klass är det som ni anropar då?

Titel med litet t är ju satt till private. Vad händer om ni istället försöker anropa variabeln direkt så här: HistoryBook.titel? (Med litet **t**)

Varför?

2. Skapa en ny klass **FaktaBok** där extra information ska vara **ämne** (subject) som är en **string**. Skap en tredje klass som ska heta **KokBok** och extra information ska vara område (area) som ska vara en **string.** Område kan vara för vegetarianer eller något annat.

Varje klass ska ha privata fält och egenskaper som sätter och hämtar värdet. Bokklassen ska ha en konstruktor som tar **tre parametrar** och de övriga en konstruktor som tar **fyra parametrar**.

Alla klasser ska ha en metod som heter **skrivUt**. Barnklasserna ska skriva ut den information som finns i basklassen och **dessutom** sin extra information.

3. Hitta på en egen klass och gör minst två klasser som ärver från den.

4. Vi antar att priset är utan moms. Skriv en metod som räknar ut priset inklusive moms och skriver ut det. Vi kan anta att priset är 25 %.