Övning 4 - Minneshantering

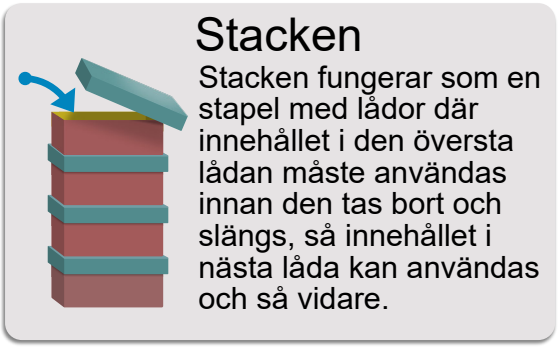
Frågor:

1. Hur fungerar stacken och heapen? Förklara gärna med exempel eller skiss på dess grundläggande funktion.

Svar:
Stacken och heapen handlar om hur minneshanteringen i .NET är uppdelad.

Stacken fungerar lite som en trave med lådor, där den översta lådan hela tiden måste gås igenom innan den kan slängas och nästa låda blir tillgänglig.

Heapen fungerar lite som om alla lådor låg utspridda på golvet. Så länge man vet vad man vill ha kan man sträcka sig efter just den lådan direkt utan att man behöver ta sig förbi några andra hinder innan.



2. Vad är Value Types respektive Reference Types och vad skiljer dem åt?

Svar:
Reference Types är referenser till värden som lagrats någonstans på heapen.Value Types lagras där de deklareras, vilket kan vara både i stacken och på heapen.

3: Följande metoder (se bild nedan) genererar olika svar. Den första returnerar 3, den andra returnerar 4, varför?

Svar:
I första exemplet deklareras int x och int y som value types i en metod. Metoden returnerar ett värde (int x), men inget värde i metoden lagras någonsans och kan därför inte kallas på i efterhand när metoden körts.

I andra exemplet deklareras x och y som klasser.

Här skapas en referens (x) till ett nytt MyInt-objekt.
MyValue i det nya objektet med referens x får värdet 3.

Sen skapas en ny referens (y) till ett nytt MyInt-Objekt.

Sen bestäms att referensen y ska referera till samma objekt som x.

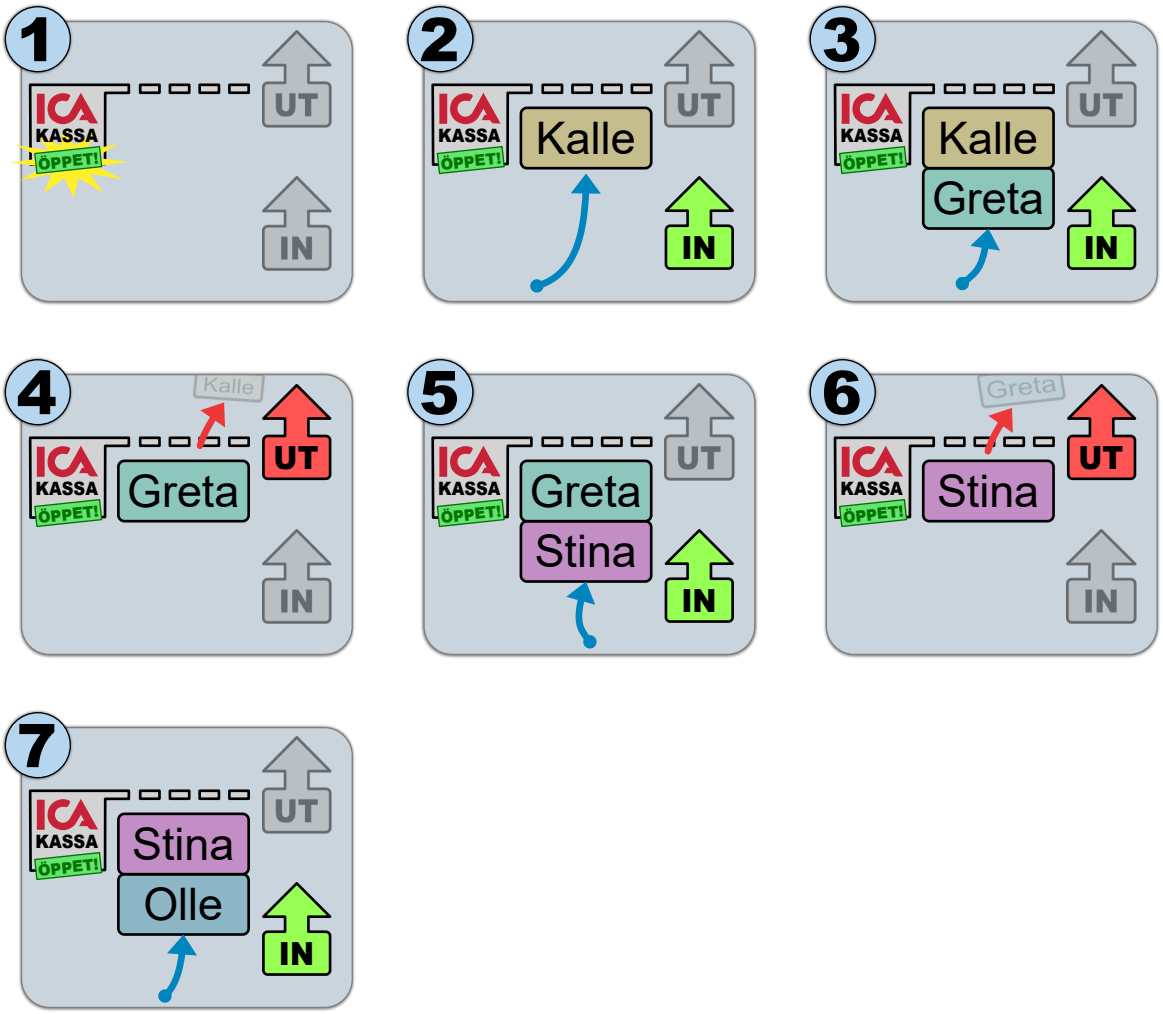
Nu när vi ändrar värdet på y.MyInt kommer även värdet på x.MyInt ändras eftersom båda refererar till samma objekt.

```
public int ReturnValue()
{
    int x = new int();
    x = 3;
    int y = new int();
    y = x;
    y = 4;
    return x;
}

0 references
public int ReturnValue2()
{
    MyInt x = new MyInt();
    x.MyValue = 3;
    MyInt y = new MyInt();
    y = x;
    y.MyValue = 4;
    return x.MyValue;
}
```

Övning 2: ExamineQueue()

1. Simulera följande kö på papper:
   a. ICA öppnar och kön till kassan är tom
   b. Kalle ställer sig i kön
   c. Greta ställer sig i kön
   d. Kalle blir expedierad och lämnar kön
   e. Stina ställer sig i kön
   f. Greta blir expedierad och lämnar kön
   g. Olle ställer sig i kön
   h. ...



Övning 3: ExamineStack()

1. Simulera ännu en gång ICA-kön på papper. Denna gång med en stack. Varför är det inte så smart att använda en stack i det här fallet?

Svar:
Om kön på Ica skulle fungera som en Stack så skulle Kalle, som ställde sig först i kön, aldrig komma därifrån så länge flera personer befinner sig i kön.