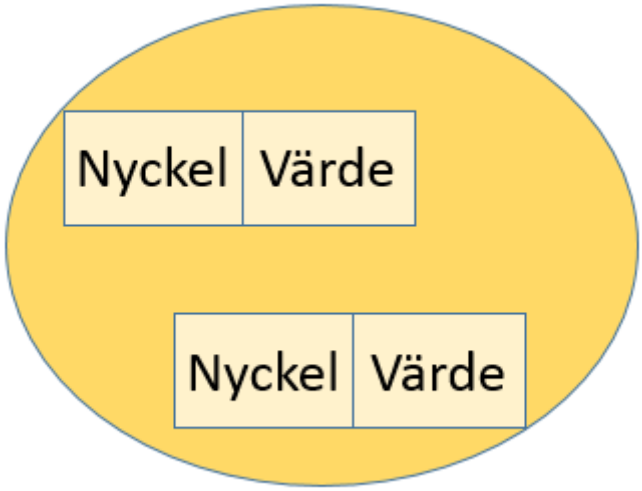
Övning Dictionary

När collection av typen Dictionary används, används nyckel och ett värde som ett par för lagringen. Duplikat av information kan lagras, men nyckel är alltid unik.



Generic collections är definierade i namespace System.Collections.Generic. Det finns tre bastyper av collection (List, HashSet och Dictionary) definierade.

I övningen skall vi bygga ett enkelt banksystem, där tre unika konto skapas, du gör en insättning för en av kunderna och sedan kontrollerar du ingående balans.

I övningen kommer du att göra två stycken klasser, en klass för kontot (Account) och en klass för bank (Bank). Som unik kombination används nyckel i form av ett löpnummer och värde i form av namnet för kund. I metoden Main(), skapas tre konton (även försök att skapa post med samma nyckel), visa balans för de tre kontona, ett konto hämtas för att göra en insättning, därefter visas aktuell balans igen.

Övning 1

Steg 1: På din virtuella maskin, starta Visual Studio.

Steg 2: Klicka på File-New-Project. I den vänstra trädstrukturen klicka på Visual C#. Klicka därefter på Console Application i det mittersta fönstret.

Steg 3: Skriv in DictionaryÖvning1 i rutan till höger om File Name, klicka därefter på OK.

Steg 4: Verifiera att `using System.Collections.Generic;` finns med överst i din applikation.

Steg 5: Skriv in följande kod som är markerad med fet text.

```
namespace DictionaryÖvning1
{
        class Account
        {
                private String name;
                public String Name
                {
                        get { return name; }
                }
                private String dateOfBirth;
                public String DateOfBirth
                {
                        get { return dateOfBirth; }
                }
                private int balance;
                public Account(String name, String
dob)
                {
                        this.name = name;
```

```
                this.dateOfBirth = dob;
                balance = 0;
        }
        public void DepositMoney(int increase)
        {
                balance = balance + increase;
        }
        public override string ToString()
        {
                return ”Namn: ” + name
”\nBalans : ” + balance;
        }
        public override bool Equals(object
obj)
        {
        Account a = (Account)obj;
        return ((name==a.Name) && (dateOfBirth
==a.DateOfBirth));
        }
        public override int GetHashCode()
        {
                return (name +
dateOfBirth).GetHashCode();
        }
        }
Class Bank
```

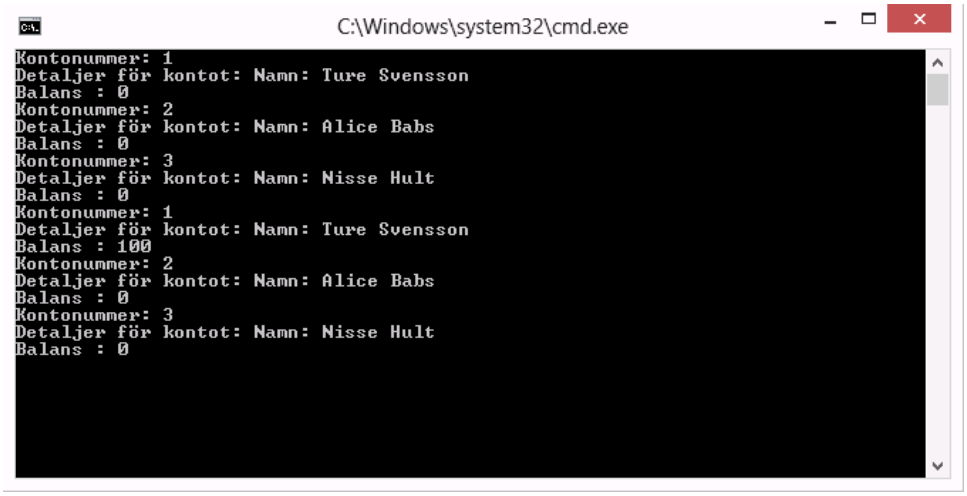
```
        {
                private Dictionary<int,Account>
accounts;
                public Dictionary<int,Account>
Accounts
                {
                get { return accounts; }
                }
                public Bank()
                {
                        accounts = new Dictionary<int,
Account>();
                }

                public void AddAccount(int number,
Account account)
                {
                        try
                        {
                                accounts.Add(number,
account);
                        }
                        catch (Exception)
                        {
                        }
                }
                public Account GetAccount(int number)
```

```
                {
                        Account a;
                        accounts.TryGetValue(number,
out a);
                        return a;
                }
        }
        Class Program
        {
            static void Main(string[] args)
            {
                Bank b = new Bank();
                Account a;
                int an;

                a = new Account(”Ture Svensson”,
”06/12/1963”);
                b.AddAccount(1, a);


                // lägger till duplikat med samma
nyckel
                a = new Account(”Ture Svensson”,
”06/12/1963”);
                b.AddAccount(1, a);



                a = new Account(”Alice Babs”,
”06/06/1928”);
                b.AddAccount(2, a);

                a = new Account(”Nisse Hult”,
”06/06/1945”);
                b.AddAccount(3, a);
```

```
                foreach (KeyValuePair<int, Account>
kvp in b.Accounts)
                {
                        an = kvp.Key;
                        a = kvp.Value;


                        Console.WriteLine("Konto
nummer: " + an +
                        "\nDetaljer för kontot: " +
a.ToString());

                }
                a = b.GetAccount(1);
                a.DepositMoney(100);

                foreach (KeyValuePair<int, Account>
kvp in b.Accounts)


                {
                        an = kvp.Key;
                        a = kvp.Value;

                        Console.WriteLine("Konto
nummer: " + an +
                        "\nDetaljer för kontot: " +
a.ToString());

                }

            }

        }

}
```

Steg 5: Klicka på DEBUG-Start Without Debugging.

Steg 6: Verifiera funktion enligt bild nedan:



Program:

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace DictionaryÖvning1

{

    class Account

    {

        private String name;

        public String Name

        {

            get { return name; }

        }


        private String dateOfBirth;

        public String DateOfBirth

        {

            get { return dateOfBirth; }

        }
```

```
    private int balance;

    public Account(String name, String dob)
    {
        this.name = name;
        this.dateOfBirth = dob;
        balance = 0;
    }

    public void DepositMoney(int increase)
    {
        balance = balance + increase;
    }

    public override String ToString()
    {
        return "Namn: " + name +  "\nBalans : " + balance;
    }

    public override bool Equals(object obj)
    {
                Account a = (Account)obj;
        return ( (name==a.Name) && (dateOfBirth==a.DateOfBirth));
    }

    public override int GetHashCode()
    {
        return (name + dateOfBirth).GetHashCode();
    }
}
```

```
class Bank
{
    private Dictionary<int,Account> accounts;
    public Dictionary<int,Account> Accounts
    {
        get { return accounts;}
    }

    public Bank()
    {
        accounts = new Dictionary<int, Account>();
    }

    public void AddAccount(int number,Account account)
    {
        try
        {
            accounts.Add(number, account);
        }
        catch (Exception)
        {
        }
    }

    public Account GetAccount(int number)
    {
        Account a;
        accounts.TryGetValue(number, out a);
        return a;
    }
```

```
}


class Program
{
    static void Main(string[] args)
    {
        Bank b = new Bank();
        Account a;
        int an;

        a = new Account("Ture Svensson", "06/12/1963");
        b.AddAccount(1, a);
        // lägger till duplikat med samma nyckel

        a = new Account("Ture Svensson", "06/12/1963");
        b.AddAccount(1, a);

        a = new Account("Alice Babs", "06/06/1928");
        b.AddAccount(2, a);

        a = new Account("Nisse Hult", "06/12/1945");
        b.AddAccount(3, a);

        foreach (KeyValuePair<int, Account>kvp in b.Accounts)
        {
            an = kvp.Key;
            a = kvp.Value;

            Console.WriteLine("Kontonummer: " + an +
```

```
                "\nDetaljer för kontot: " + a.ToString());


            }


            a = b.GetAccount(1);
            a.DepositMoney(100);


            foreach (KeyValuePair<int, Account>kvp in b.Accounts)
            {


                an = kvp.Key;
                a = kvp.Value;


                Console.WriteLine("Kontonummer: " + an +
                                  "\nDetaljer för kontot: " + a.ToString());
            }
        }
    }
}
```