

การเขียนโปรแกรม ด้วยภาษา C#

Class Inheritance – Part 2

Virtual and Override Methods

Virtual and Override Methods

- ในหัวข้อที่ผ่านมา เราสามารถใช้ reference ของ derived class เรียกไปยัง method ของ base class ได้ โดยใช้คำสั่ง `base.<methodName(>`
- ในทางกลับกัน เราสามารถใช้ reference ของ base class เรียกไปยัง method ของ derived class ได้
 - ใช้ virtual method ใน base class
 - ใช้ override method ใน base class
 - Virtual และ override method ต้องมี signature และ return type ตรงกัน

รูปแบบการเขียน Virtual - Override Methods

Virtual

```
class BaseClass
{
    [virtual] public void Print()
    {
        ...
    }
}
```

Virtual อยู่ใน Base Class

Override

```
class DerivedClass : BaseClass
{
    [override] public void Print()
    {
        ...
    }
}
```

Override อยู่ใน Derived Class
(ลำดับสุดท้าย)

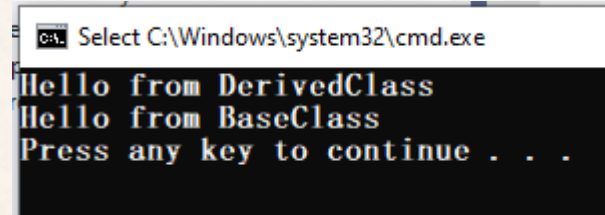
ตัวอย่าง

```
class BaseClass
{
    virtual public void Print()
    {
        Console.WriteLine("Hello from BaseClass");
    }
}
```

```
class DerivedClass : BaseClass
{
    override public void Print()
    {
        Console.WriteLine("Hello from DerivedClass");
    }
}
```

```
using System;
class Program
{
    static void Main()
    {
        DerivedClass derivedRef = new DerivedClass();
        BaseClass baseRef = new BaseClass();

        derivedRef.Print();
        baseRef.Print();
    }
}
```



```
cmd - Select C:\Windows\system32\cmd.exe
Hello from DerivedClass
Hello from BaseClass
Press any key to continue . . .
```

ตัวอย่างการเขียน Virtual - Override Methods

```
class BaseClass
{
    virtual public void Print()
    {
        Console.WriteLine("Hello from BaseClass");
    }
}
```

```
class DerivedClass : BaseClass
{
    override public void Print()
    {
        Console.WriteLine("Hello from DerivedClass");
    }
}
```

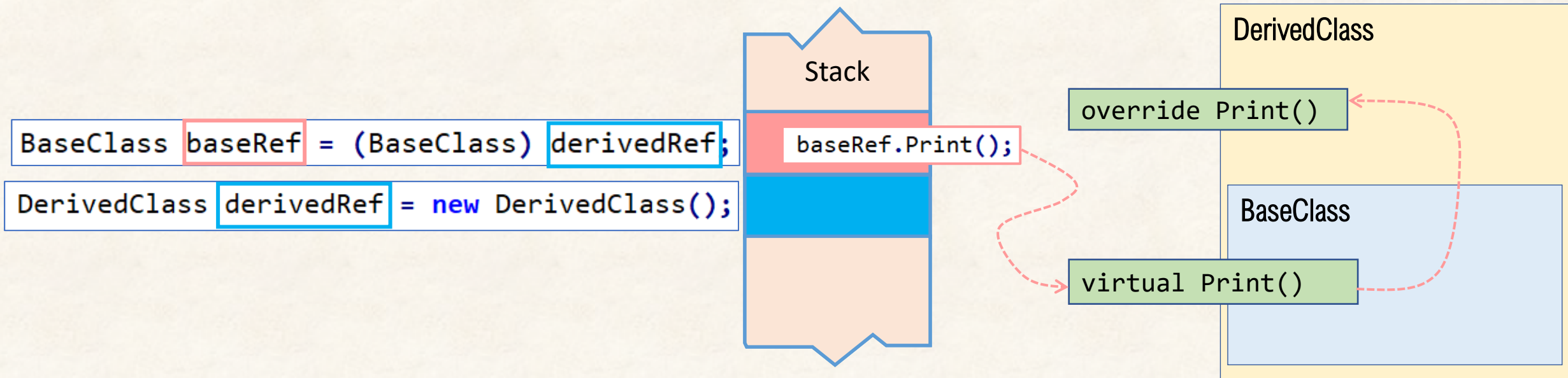
```
using System;
class Program
{
    static void Main()
    {
        DerivedClass derivedRef = new DerivedClass();
        BaseClass baseRef = (BaseClass) derivedRef;

        derivedRef.Print();

        baseRef.Print();
    }
}
```

Result = ??????

การเข้าถึงสมาชิกโดย virtual และ override



ตัวอย่างการเขียน Virtual - Override Methods

```
class BaseClass
{
    virtual public void Print()
    {
        Console.WriteLine("Hello from BaseClass");
    }
}

class DerivedClass : BaseClass
{
    override public void Print()
    {
        Console.WriteLine("Hello from DerivedClass");
    }
}
```

```
using System;
class Program
{
    static void Main()
    {
        DerivedClass derivedRef = new DerivedClass();
        BaseClass baseRef = (BaseClass) derivedRef;

        derivedRef.Print();
        baseRef.Print();
    }
}
```


การ Override Method ที่ผ่านการ overridden

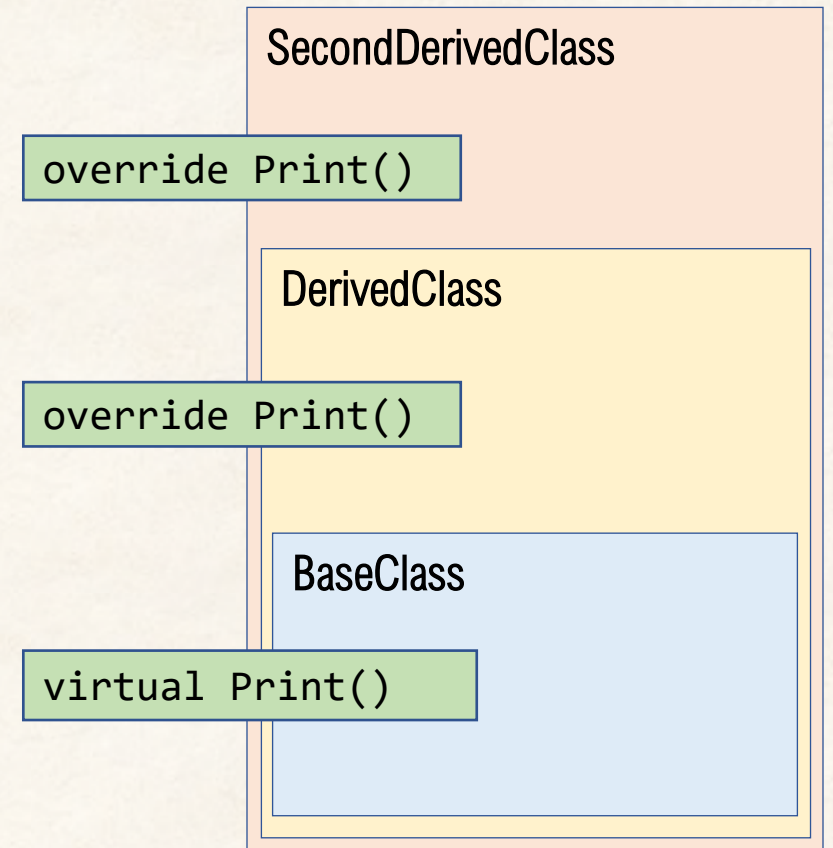
- Methods สามารถทำ overridden ได้หลายระดับ
 - Method ในห่วงโซ่ของการ override จะถูกเรียกไปจนคลาสลำดับสุดท้ายในการสืบทอด
- ถ้า method ของ derived class มี signature และ return type ตรงกับของ base class แต่ไม่มีการระบุ override ก็จะไม่เข้าข่ายกระบวนการ virtual-override
 - Compiler จะแจ้งเตือนว่ามีการทับการกระทำโดย derived class
 - แต่ไม่ถือเป็น error

ตัวอย่าง

```
class BaseClass
{
    virtual public void Print()
    {
        Console.WriteLine("Hello from BaseClass");
    }
}

class DerivedClass : BaseClass
{
    override public void Print()
    {
        Console.WriteLine("Hello from DerivedClass");
    }
}

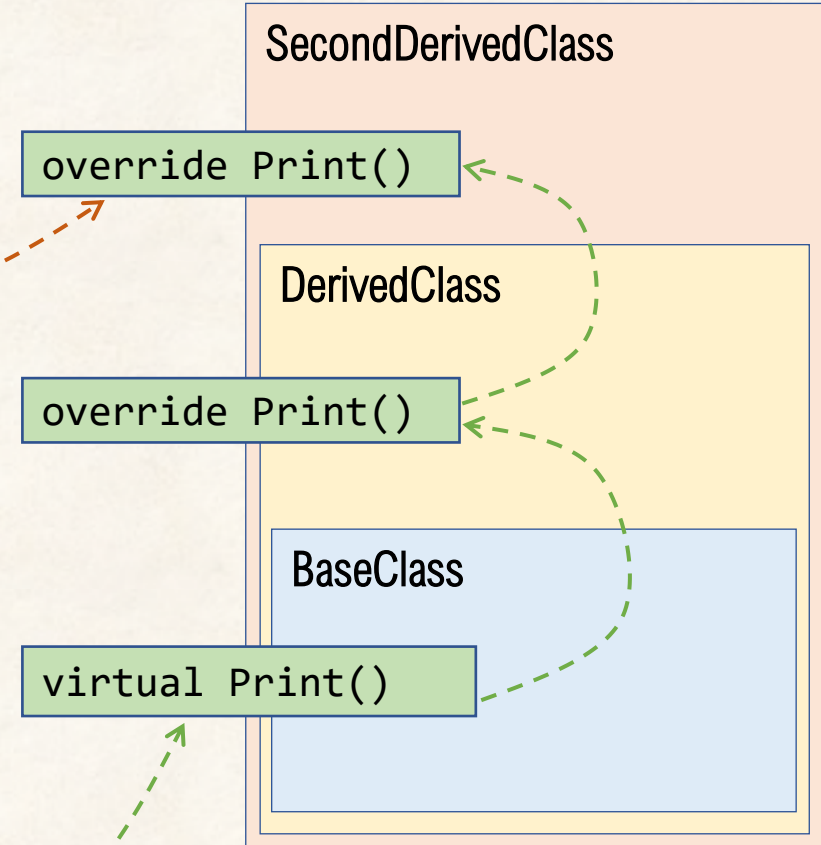
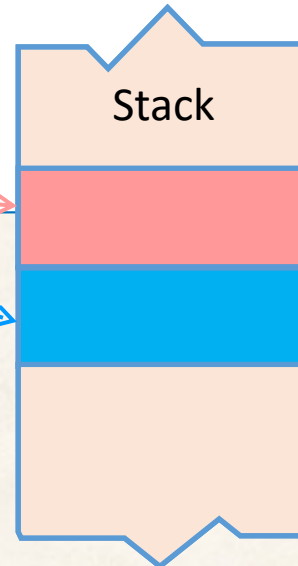
class SecondDerivedClass : DerivedClass
{
    override public void Print()
    {
        Console.WriteLine("Hello from SecondDerivedClass");
    }
}
```



ตัวอย่าง

```
using System;
class Program
{
    static void Main()
    {
        SecondDerivedClass sdRef = new SecondDerivedClass();
        BaseClass baseRef = (BaseClass) sdRef;

        sdRef.Print();
        baseRef.Print();
    }
}
```



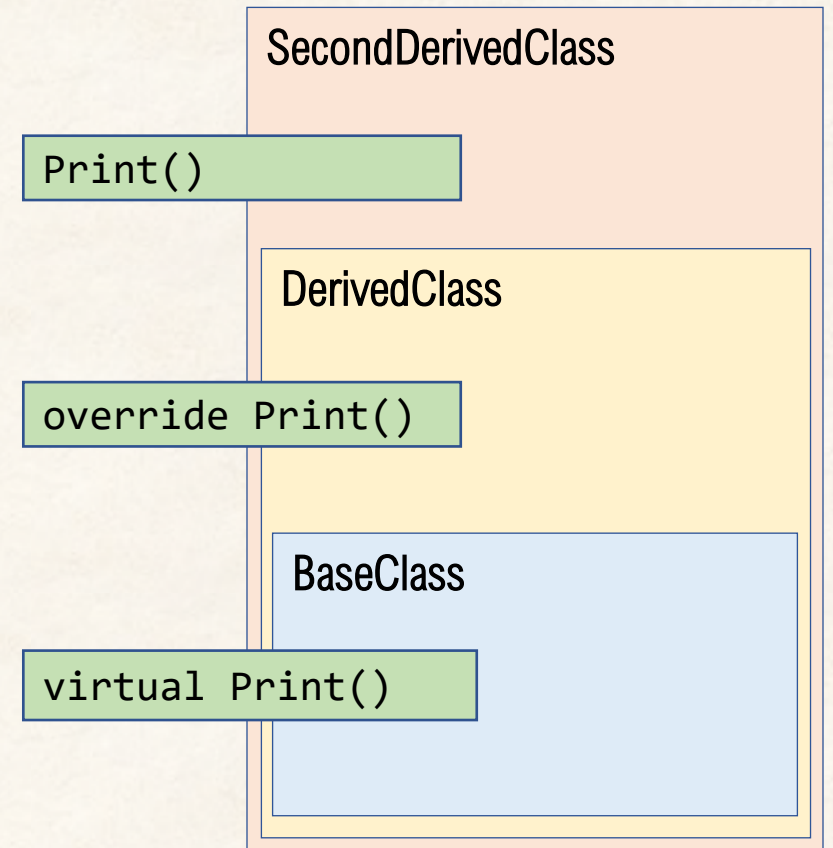
C:\Windows\system32\cmd.exe
Hello from SecondDerivedClass
Hello from SecondDerivedClass
Press any key to continue . . .

ตัวอย่าง

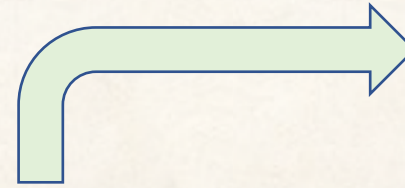
```
class BaseClass
{
    virtual public void Print()
    {
        Console.WriteLine("Hello from BaseClass");
    }
}

class DerivedClass : BaseClass
{
    override public void Print()
    {
        Console.WriteLine("Hello from DerivedClass");
    }
}

class SecondDerivedClass : DerivedClass
{
    public void Print()
    {
        Console.WriteLine("Hello from SecondDerivedClass");
    }
}
```



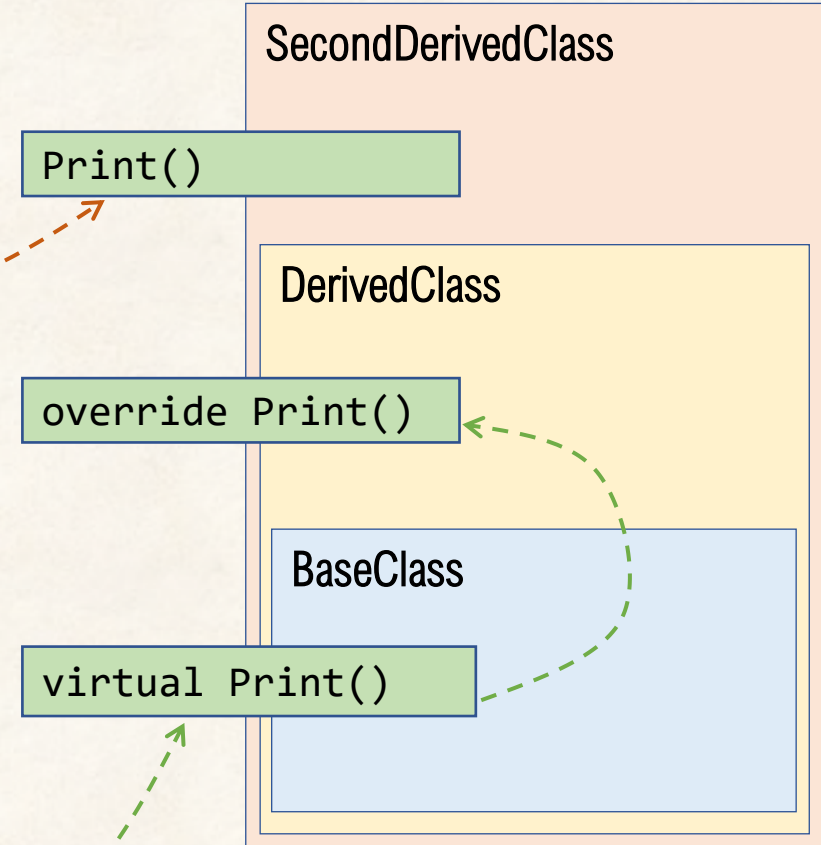
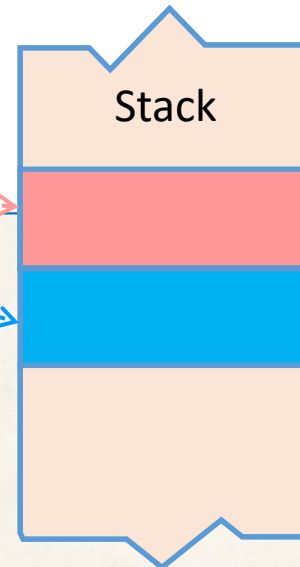
ตัวอย่าง



```
C:\Windows\system32\cmd.exe
Hello from SecondDerivedClass
Hello from DerivedClass
Press any key to continue . . .
```

```
using System;
class Program
{
    static void Main()
    {
        SecondDerivedClass sdRef = new SecondDerivedClass();
        BaseClass baseRef = (BaseClass) sdRef;

        sdRef.Print();
        baseRef.Print();
    }
}
```



การ override ด้วยคำสั่ง new

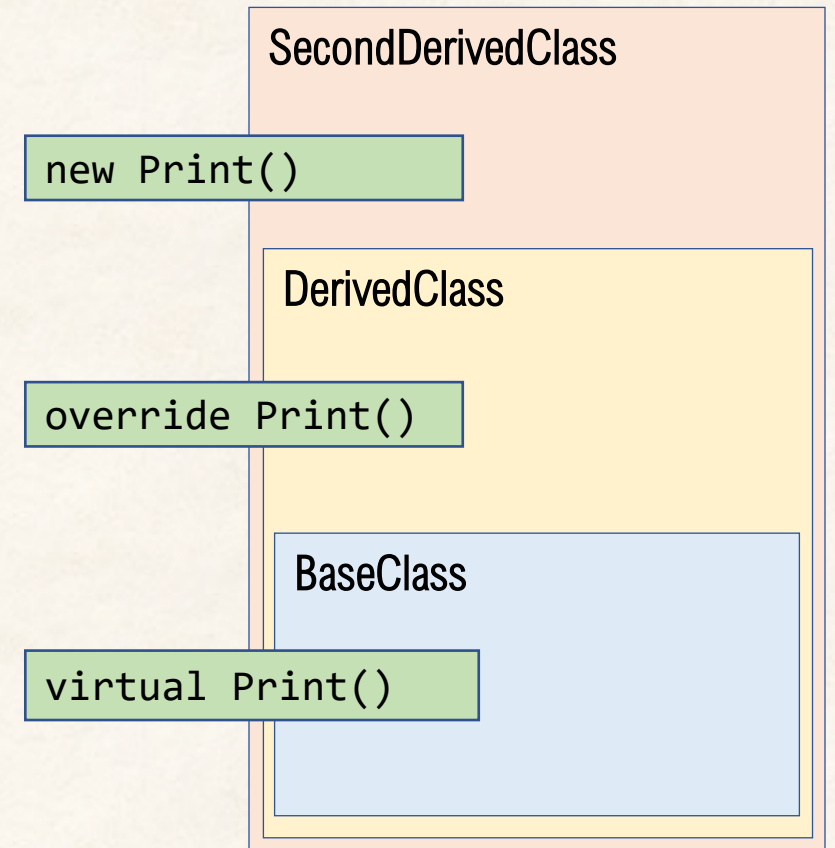
- การสืบทอด Method ในห่วงโซ่ของการ override จะเรียก method ที่มี signature ตรงกันตลอดไปจนคลาสลำดับสุดท้ายในการสืบทอด
- เราสามารถใช้คำสั่ง new แทน override
 - Compiler จะไม่แจ้งเตือนว่ามีการทับการกระทำโดย derived class
 - การเรียก method ในห่วงโซ่ของการสืบทอดจะสิ้นสุดในระดับก่อนคลาสที่ override ด้วยคำสั่ง new
 - (เทียบได้กับการตั้งนามสกุลใหม่ที่จะมีผลต่อรุ่นลูกหลาน แต่ไม่ส่งผลถึงบรรพบุรุษ)

ตัวอย่าง

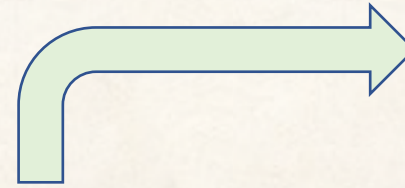
```
class BaseClass
{
    virtual public void Print()
    {
        Console.WriteLine("Hello from BaseClass");
    }
}

class DerivedClass : BaseClass
{
    override public void Print()
    {
        Console.WriteLine("Hello from DerivedClass");
    }
}

class SecondDerivedClass : DerivedClass
{
    new public void Print()
    {
        Console.WriteLine("Hello from SecondDerivedClass");
    }
}
```



ตัวอย่าง



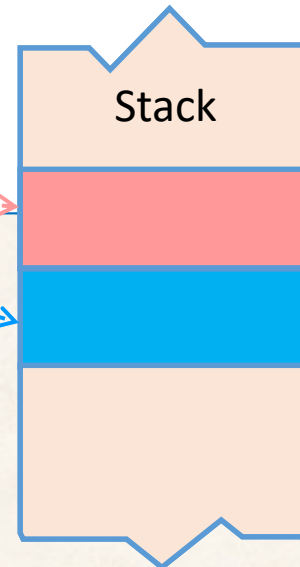
```
C:\Windows\system32\cmd.exe
Hello from SecondDerivedClass
Hello from DerivedClass
Press any key to continue . . .
```

```
using System;
class Program
{
    static void Main()
    {
        SecondDerivedClass sdRef = new SecondDerivedClass();
        BaseClass baseRef = (BaseClass) sdRef;

        sdRef.Print();
        baseRef.Print();
    }
}
```

sdRef.Print();

baseRef.Print();



SecondDerivedClass

new Print()

DerivedClass

override Print()

BaseClass

virtual Print()

