

Визуал програмчлал

Ү.ИТ203

Лекц-3

Агуулга

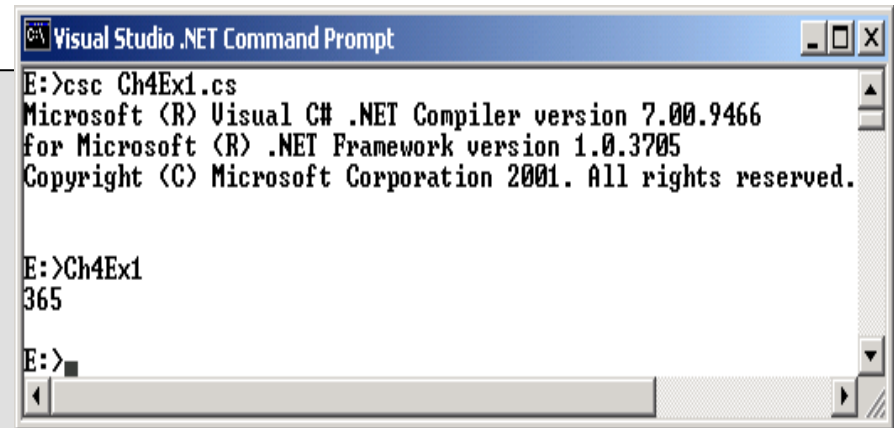
- Байгуулагч ба устгагч функц
- Полиморфизм
- Abstract класс
- Interfaces
- Exception боловсруулалт

Байгуулагч функц

- Байгуулагч функц нь объектод анхны утга олгоход хэрэглэдэг.
- Байгуулагч функцийн нэр классын нэртэй ижил байна.
- Байгуулагч функц утга буцаадаггүй

Байгуулагч функц

```
using System;
public class DaysInYear
{
    private int days;
    public DaysInYear()
    {
        days = 365;
    }
    static void Main(String[] args)
    {
        DaysInYear newDaysInYear = new DaysInYear();
        Console.WriteLine (newDaysInYear.days);
    }
}
```



Visual Studio .NET Command Prompt

```
E:>csc Ch4Ex1.cs
Microsoft (R) Visual C# .NET Compiler version 7.00.9466
for Microsoft (R) .NET Framework version 1.0.3705
Copyright (C) Microsoft Corporation 2001. All rights reserved.

E:>Ch4Ex1
365

E:>
```

Параметртай байгуулагч функц

...

```
public class DaysInYear
{
    private int days;
    public DaysInYear()
    {
        days = 365;
    }
    public DaysInYear(int day)
    {
        days = day;
    }
}
```

```
public DaysInYear(String dayOne)
{
    days
=Convert.ToInt32(dayOne);
}
public void setDay(int newDays)
{
    days = newDays;
}
...
}
```

Устгагч функц

- Устгагч функц нь санах ойд бий болсон объектыг чөлөөлхөд хэрэглэдэг.
- Устгагч функц нь классынхаа нэртэй адил боловч өмнөө (`~`) тэмдэгтэй хамт хэрэглэгддэг.

```
...  
~DaysInYear()  
{  
    //Destructor Implementation  
}  
...
```

Base түлхүүр үг

- Суурь классын байгуулагчийг дэд классын байгуулагчтай хамт дуудахдаа **base** түлхүүр үгийг ашигладаг. Энэ нь дэд классын байгуулагчууд суурь классаар бэлтгэгддэг дундын нөөцийг ашиглах тохиолдолд ашигтай юм.

Полиморфизм

- Полиморфизм нь суурь классаас удамшисан классын методыг дуудахад ашиглагддаг.
- Энэ тохиолдолд үндсэн класс дотрох метод нь `virtual` –аар зарлагдсан байх хэрэгтэй
- Харин удамшисан классын метод нь `override` байх ёстой.

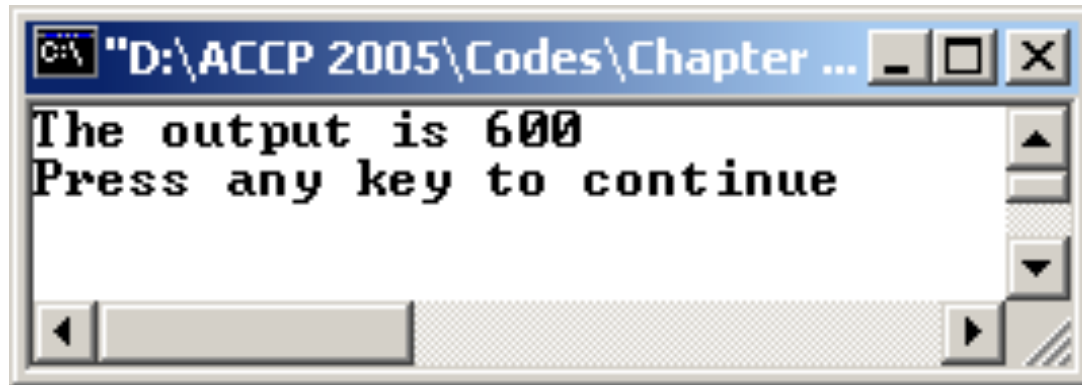
Полиморфизм – жишээ

```
using System;
class Parent
{
    public int MethodA()
    {
        return (MethodB()*MethodC());
    }
    public virtual int MethodB()
    {
        return(10);
    }
    public int MethodC()
    {
        return(20);
    }
}
```

Полиморфизм - жишээ

```
class Child : Parent
{
    public override int MethodB()
    {
        return(30);
    }
}
class PolymorphDemo
{
    public static void Main()
    {
        Child ObjChild = new Child();
        Console.WriteLine("The output is "
+ObjChild.MethodA());
    }
}
```

Үр дүн



Abstract класс

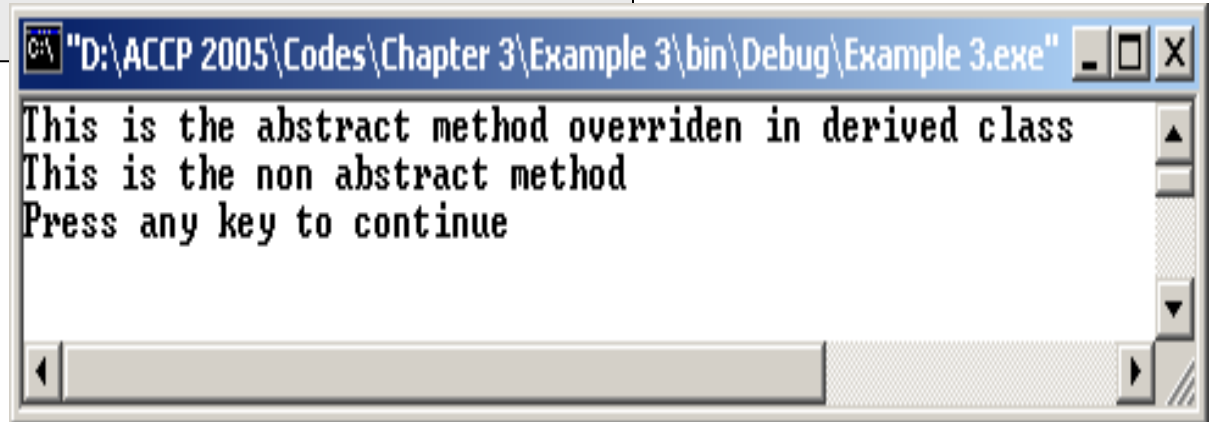
- Хийсвэр класс нь зөвхөн бусад классын суурь болж хэрэглэгдэнэ. Өөрөөр хэлбэл ийм классаас объект шууд үүсгэж болохгүй гэсэн үг. Хийсвэр класс нь хийсвэр болон хийсвэр бус дүрэм, шинж чанарыг агуулж болно.

Abstract класс - жишээ

```
using System;
abstract class BaseClass
{
    public abstract void MethodA();
    public void MethodB()
    {
        Console.WriteLine ("This is the non abstract method"); }
}
class DerivedClass : BaseClass
{
    public override void MethodA()
    {
        Console.WriteLine ("This is the abstract method
overridden in derived class");
    }
}
```

Abstract класс - жишээ

```
class AbstractDemo
{
    public static void Main()
    {
        DerivedClass objDerived = new
            DerivedClass();
        BaseClass objBase = objDerived;
        objBase.MethodA();
        objDerived.MethodB();
    }
}
```

A screenshot of a Windows command prompt window. The title bar shows the file path "D:\ACCP 2005\Codes\Chapter 3\Example 3\bin\Debug\Example 3.exe". The window contains three lines of text: "This is the abstract method overridden in derived class", "This is the non abstract method", and "Press any key to continue". The text is displayed in a monospaced font. The window has standard Windows XP-style window controls (minimize, maximize, close) in the top right corner and a scrollbar on the right side.

"D:\ACCP 2005\Codes\Chapter 3\Example 3\bin\Debug\Example 3.exe"

This is the abstract method overridden in derived class
This is the non abstract method
Press any key to continue

Interface

- Interface нь abstract классын илүү сайжруулсан хэлбэр юм.
- Interface нь зөвхөн abstract методыг агуулсан байдаг боловч методын хийх үйлдлийг тодорхойлж өгдөггүй.
- Бусад классууд түүнүүс удимшиж болдог.

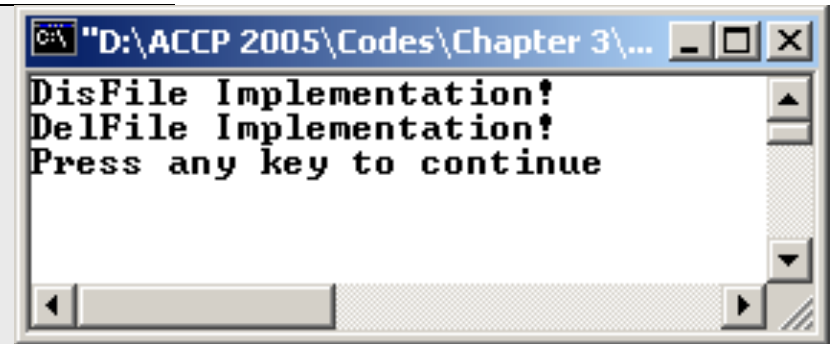
```
public interface IFile
{
    int delFile();
    void disFile();
}
```

Interfaces - Жишээ

```
public interface IFile
{
    int delFile();
    void disFile();
}
public class MyFile : IFile
{
    public int delFile()
    {
        System.Console.WriteLine ("DelFile Implementation!");
        return(0);
    }
    public void disFile()
    {
        System.Console.WriteLine ("DisFile Implementation!");
    }
}
```


Interfaces – Үр дүн:

```
class InterfaceDemo
{
    public static void Main()
    {
        MyFile objMyFile = new MyFile();
        objMyFile.disFile();
        int retValue = objMyFile.delFile();
    }
}
```



Interfaces – Удамшил

```
public interface IFile
{
    int delFile();
    void disFile();
}
public class BaseforInterface
{
    public void open()
    {
        System.Console.WriteLine ("This is the open method of BaseforInterface");
    }
}
```

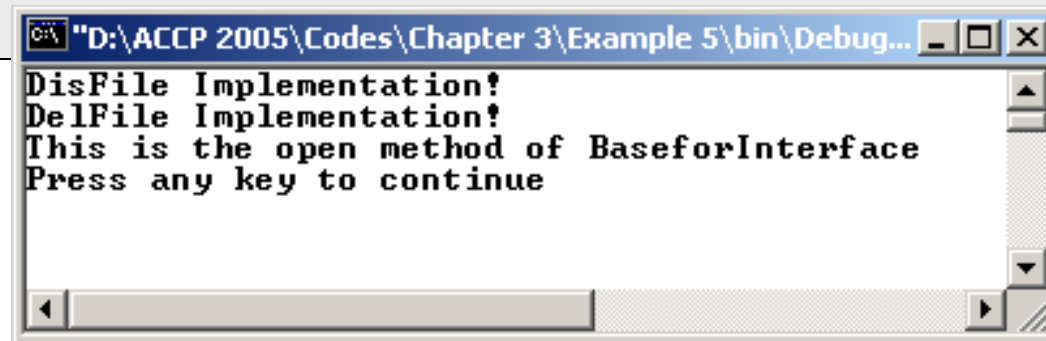
Interfaces – Классын удамшил

```
public class MyFile : BaseforInterface, IFile
{
    public int delFile()
    {
        System.Console.WriteLine ("DelFile Implementation!");
        return(0);
    }
    public void disFile()
    {
        System.Console.WriteLine ("DisFile Implementation!");
    }
}
```

Interface - Классын удамшил

Үр дүн:

```
class Test
{
    static void Main()
    {
        MyFile objMyFile = new MyFile();
        objMyFile.disFile();
        int retValue = objMyFile.delFile();
        objMyFile.open();
    }
}
```



```
C:\D:\ACCP 2005\Codes\Chapter 3\Example 5\bin\Debug...
DisFile Implementation!
DelFile Implementation!
This is the open method of BaseforInterface
Press any key to continue
```

Exception боловсруулалт

- Exception гэж юу вэ?
- Exception тохиолдоход юу болох вэ?
- System.Exception
- Try, catch, finally

Exception гэж юу вэ?

- Програм хэвийн ажиллаж байх/run-time / үед ямар нэгэн алдаа гарвал exception дуудагддаг.
- Exception-ээр алдааг засдаг.
- C# хэлэнд exception нь кодоор эсвэл CLR-ээр үүсгэгдэнэ.
- Бүх стандарт exception класс нь Exception стандарт класаас удамших ба классын нэрэнд exception үг агуулагдсан байдаг.
- Хэрэглэгч exception бичихэд нэрэндээ exception үг заавал оруулах шаардлагатай биш. Ерөнхийдөө нэг функцэд хоёр блок л ордог.
- Жишээ нь:

try + finally,
try + catch

Try, catch, finally

```
try {  
    //алдаа үүсэж болзошгүй кодын хэсэг  
} catch() {  
    //алдааг боловсруулах I хэсэг  
} catch() {  
    //алдааг боловсруулах II хэсэг  
} finally {  
    //үргэлж ажиллах кодын хэсэг  
}
```

Catch блок

```
class Class1{  
    static void Main(string[] args) {  
        int huvaagdagch = 0, huvaagch = 0, hariu = 0;  
        try {  
            hariu = huvaagdagch / huvaagch;  
        }  
        catch (System.DivideByZeroException ex) {  
            Console.WriteLine (“{0} exception ajillav”, ex);  
        }  
        Console.ReadLine ();  
    }  
}
```

- catch блок дотор exception-ы үед ажиллах код байна.
- try блокт хэд хэдэн catch блок байж болно. Огт байхгүй байж болно.

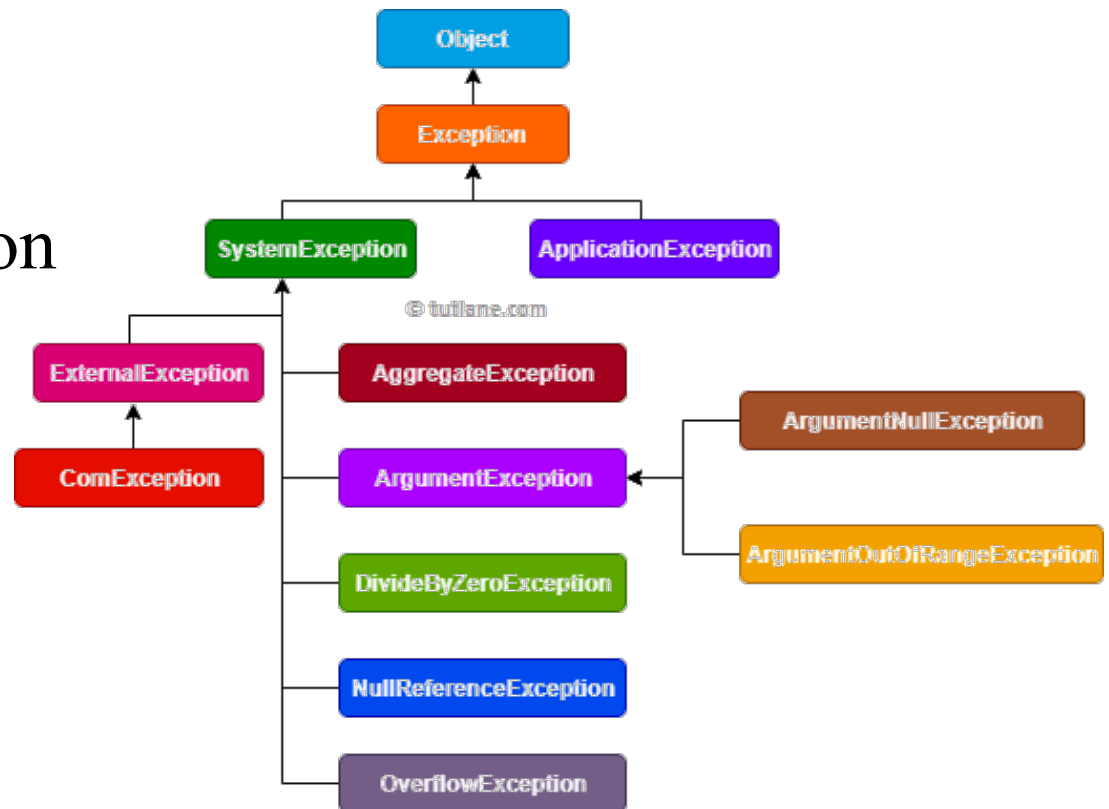
Finally блок

- Энэ ихэвчлэн try блокийн үйлдлийн үр дүнд хийгдэх санах ой чөлөөлөх үйлдэл бичигдэнэ. Жнь:

```
void ReadData(String pathname){  
    FileStream fs = null;  
    try {  
        fs = new FileStream(pathname, FileMode.Open);  
    } catch(OverflowException) {  
    }  
    finally{  
        if(fs != null) fs.Close();  
    }  
}
```

Exception төрөл

- DataException
- FormatException
- IOException
- Arithmetic Exception
- Exception
- Хэрэглэгчийн Exception



System Exception жишээ

The screenshot displays the Visual Studio IDE with a C# project named 'ExceptionTutorial'. The code file 'HelloException.cs' is open, showing a class 'HelloException' with a static method 'Main'. The code prints 'Three', 'Two', and 'One' before attempting a division by zero, which triggers an exception. The exception details window is open, showing the error type 'DivideByZeroException' and the message 'An unhandled exception of type 'System.DivideByZeroException' occurred in ExceptionTutorial.exe'. It also provides troubleshooting tips and actions like 'View Detail...', 'Copy exception detail to the clipboard', and 'Open exception settings'. A console window in the bottom right shows the output: 'Three', 'Two', and 'One'.

```
namespace ExceptionTutorial
{
    0 references
    class HelloException
    {
        0 references
        public static void Main(string[] args)
        {
            Console.WriteLine("Three");

            int value = 10 / 2;

            Console.WriteLine("Two");

            value = 10 / 1;

            Console.WriteLine("One");

            int d = 0;

            value = 10 / d;

            Console.WriteLine("Let's go!");

            Console.Read();
        }
    }
}
```

DivideByZeroException was unhandled

An unhandled exception of type 'System.DivideByZeroException' occurred in ExceptionTutorial.exe

Additional information: Attempted to divide by zero.

Troubleshooting tips:

- [Make sure the value of the denominator is not zero before performing a division operation.](#)
- [Get general help for this exception.](#)

[Search for more Help Online...](#)

Exception settings:

- ☐ Break when this exception type is thrown

Actions:

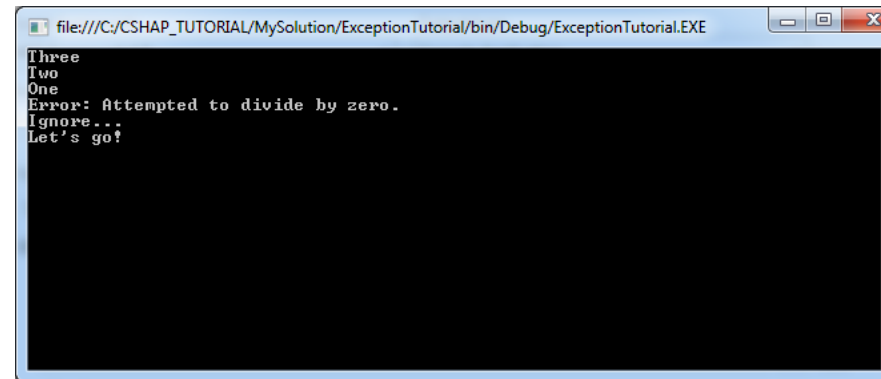
- [View Detail...](#)
- [Copy exception detail to the clipboard](#)
- [Open exception settings](#)

file:///C:/CSHAP_TUTORIAL/MySolution/ExceptionTutorial/bin/Debug/ExceptionTutorial.EXE

Three
Two
One

System Exception жишээ

```
namespace ExceptionTutorial{  
    class HelloCatchException {  
        public static void Main(string[] args) {  
            Console.WriteLine("Three");  
            int value = 10 / 2;  
            Console.WriteLine("Two");  
            value = 10 / 1;  
            Console.WriteLine("One");  
            int d = 0;  
            try {  
                value = 10 / d;  
                Console.WriteLine("Value =" + value);  
            }  
            catch (DivideByZeroException e) {  
                Console.WriteLine("Error: " + e.Message);  
                Console.WriteLine("Ignore...");  
            }  
            Console.WriteLine("Let's go!");  
            Console.Read();  
        }  
    }  
}
```



```
file:///C:/CSHAP_TUTORIAL/MySolution/ExceptionTutorial/bin/Debug/ExceptionTutorial.EXE  
Three  
Two  
One  
Error: Attempted to divide by zero.  
Ignore...  
Let's go!
```

Анхаарал тавьсанд баярлалаа.