

第44章 抽象メソッドと抽象クラス



派生クラスで必ずオーバーライドしてもらう必要があるメソッドを宣言することができます。

メソッドにabstract修飾子をつけるとそのメソッドは抽象メソッドといい、中身を持ちません。

abstract データ型 メソッド名 (パラメータリスト);

抽象メソッドはstaticにはできません。また、抽象メソッドを持つクラスのインスタンスを生成することもできません。

一つでも抽象メソッドを有するクラスを「抽象クラス」といいます。抽象クラスは、修飾子abstractをつける必要があります。

抽象クラスから、派生したクラスは抽象メソッドを全部オーバーライドするか、もしくはそのクラスも抽象クラスにしないてはいけません。

最も簡単な抽象クラスの例を示します。

```
// abstract01.cs

using System;

abstract class MyClassA
{
    public abstract void show();
}

class MyClass : MyClassA
{
    public override void show()
    {
        Console.WriteLine("showメソッドです");
    }
}

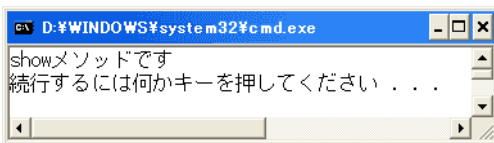
class abstract01
{
    public static void Main()
    {
        MyClass mc = new MyClass();

        mc.show();
    }
}
```

抽象クラスMyClassAには、抽象メソッドshowがあります。当然中身はありません。

MyClassAを継承したMyClassでは、showメソッドをオーバーライドしています。

実行結果は次のようになります。



次の例では、抽象クラスCalcOrgをCalcが継承してメソッドをオーバーライドしています。CalcOrgクラスは、計算をするクラスのひな形とも考えられます。そして実際に計算をするクラスがCalcクラスです。

```
// abstract02.cs

using System;

abstract class MyCalcOrg
{
    public abstract void puls(int x);
    public abstract int ans();
    public abstract void minus(int x);
}

class MyCalc : MyCalcOrg
{
    int sum;

    public override void puls(int x)
```

```

    {
        sum += x;
    }
    public override int ans()
    {
        return sum;
    }
    public override void minus(int x)
    {
        sum -= x;
    }
    public void setZero()
    {
        sum = 0;
    }
    public MyCalc()
    {
        setZero();
    }
}

class abstract02
{
    public static void Main()
    {
        int num = 0;

        MyCalc m = new MyCalc();

        while (true)
        {
            Console.Write("整数値(*で終了,=で計算結果)--- ");
            string strnum = Console.ReadLine();
            if (strnum == "*")
                break;
            else if (strnum == "=")
            {
                Console.WriteLine("計算結果-- {0}", m.ans());
                m.setZero();
                continue;
            }
            else
            {
                num = Int32.Parse(strnum);
                Console.Write("記号(+,-)--- ");
                string kigo = Console.ReadLine();

                switch (kigo)
                {
                    case "+":
                        m.puls(num);
                        break;
                    case "-":
                        m.minus(num);
                        break;
                    default:
                        Console.WriteLine("入力エラー");
                        return;
                }
            }
        }
    }
}

```

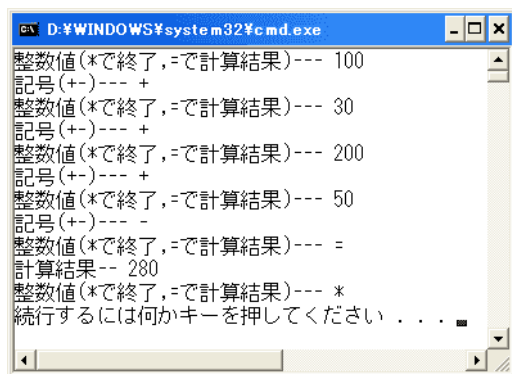
ごく簡単な、加減計算機です。整数値しか扱えませ^ん。

100 + 200 の計算は、100を入力、+を入力、200を入力、+を入力、=を入力で行います。

100 - 30 の計算は、100,+,30,-,=の順に入力します。

100 + 30 + 200 - 50 の計算は、100,+,30,+,200,+,50,-,=と入力する必要があります。

では、実行結果を見てみましょう。



```

D:\WINDOWS\system32\cmd.exe
整数値(*で終了,=で計算結果)--- 100
記号(+)---- +
整数値(*で終了,=で計算結果)--- 30
記号(+)---- +
整数値(*で終了,=で計算結果)--- 200
記号(+)---- +
整数値(*で終了,=で計算結果)--- 50
記号(+)---- +
整数値(*で終了,=で計算結果)--- =
計算結果-- 280
整数値(*で終了,=で計算結果)--- *
続行するには何かキーを押してください . . .

```

[\[C# Index\]](#) [\[総合Index\]](#) [\[Previous Chapter\]](#) [\[Next Chapter\]](#)

Update 19/Sep/2006 By Y.Kumei

当ホームページの一部または全部を無断で複写、複製、転載あるいはコンピュータ等のファイルに保存することを禁じます。