個数が変わるデータをまとめて扱おう

● 配列の要素数を変更するには？

入門編第4号で配列の解説をしました。ここで簡単に配列の復習をしてみましょう。配列は、複数のデータをまとめて扱うのに便利でしたよね？　配列の宣言時にはその要素数を指定しました。そのため、配列を利用する場合、事前に扱うデータ数が確定しているのなら問題はありません。しかし、処理の途中でデータを追加、削除する場合はどうでしょうか？　例えば、データの追加、削除の度に新たな配列を用意して、データを移すことも可能ですが、この方法だと配列の要素が増えるほどデータを移す処理が多くなり、処理が遅くなってしまいます。

そこで、処理中にデータが増減する場合は、効率よく必要な量だけ追加、削除できる**ArrayList**(アレイリスト)と**List**(リスト)を利用します。

● ArrayList を利用してみよう

まずは、**ArrayList**の宣言をしてみましょう。**ArrayList**の宣言は以下のように記述します。

```
ArrayList 変数名 = new ArrayList();
```

では、実際に宣言例を挙げてみます。ただし、**ArrayList**を使うには下のような準備が必要です。

```
 1 using System;
 2 using System.Collections.Generic;
 3 using System.Linq;
 4 using System.Text;
 5 using System.Threading.Tasks;
 6
 7 namespace データ追加方法_ArrayList
 8 {
 9     class Program
10     {
11         static void Main(string[] args)
12         {
13         }
14     }
15 }
```

```
 1 using System;
 2 using System.Collections.Generic;
 3 using System.Linq;
 4 using System.Text;
 5 using System.Threading.Tasks;
 6 using System.Collections;
 7
 8 namespace データ追加方法_ArrayList
 9 {
10     class Program
11     {
12         static void Main(string[] args)
13         {
14             // ArrayListの宣言
15             ArrayList arrayList = new ArrayList();
16         }
17     }
18 }
```

using System.Collections;を追加します

ArrayListは配列と違い要素数を指定しません。これは、要素数が一定ではなく、必要に応じて変更できるからです。**ArrayList**に新しいデータを追加したければ、追加する処理をプログラムに書くだけです。そうすれば、**ArrayList**の末尾にデータを追加してくれます。また、配列と同様にデータはインデックスで取り出せます。

データを追加するには以下のように記述して、Addメソッドを呼び出します。

変数名.Add（追加するデータ）；

このメソッドに引数として渡す「追加するデータ」には、数値や文字列といったデータを直接書くことができます。また、「追加するデータ」のところに変数名を書くことで、その変数に入っているデータを追加することもできます。

プログラム例を挙げてみます。

```
// ArrayListの宣言
ArrayList arrayList = new ArrayList();

// int型変数の宣言と初期化
int number1 = 1;
int number2 = 2;

// arrayListにデータを追加
arrayList.Add(number1);
arrayList.Add(number2);
arrayList.Add(3);
arrayList.Add(4);
arrayList.Add(5);
```

上記のように**ArrayList**にデータを追加した場合、arrayList [0]には「1」、arrayList [1]には「2」、arrayList [2]には「3」、arrayList [3]には「4」、arrayList [4]には「5」が入ります。

● ArrayList に異なるデータ型のデータを追加してみよう

配列は、宣言時にデータ型を指定しました。そのため、1つのデータ型のデータしか扱うことができません。一方、**ArrayList**の宣言時にはデータ型を書きません。これは、**ArrayList**が扱うデータ型は『オブジェクト型』というあらゆるデータ型を代入できる万能の型だからです。そのため、異なるデータ型でも気にせず追加することができます。ただし、データを取り出す時には、取り出すデータのデータ型でキャストする必要があります。

キャストは以前入門編第4号で扱いましたが、キャストについて復習してみましょう。キャストとは他の型にみなす機能です。例えば、int型変数をdouble型変数にみなして計算をしたり、逆にdouble型変数をint型変数にみなして計算をしたりすることを可能にしてくれます。

プログラム例を挙げてみます。

```
// int型変数の宣言と初期化
int number = 125;
int display = 0;

// double型変数の宣言と初期化
double total = number * 1.05;

// double型変数をint型変数にキャストして代入
display = (int)total;
```

このキャストを利用して、**ArrayList**からデータを取り出します。

プログラム例を挙げてみます。

```
// ArrayListの宣言
ArrayList arrayList = new ArrayList();

// arrayListにデータを追加
arrayList.Add(1);
arrayList.Add(3.1);
arrayList.Add("名前");

// キャストして代入
int number1 = (int)arrayList[0];
double number2 = (double)arrayList[1];
string name = (string)arrayList[2];
```

では、コンソール アプリケーションで実際に**ArrayList**を利用してみましょう。

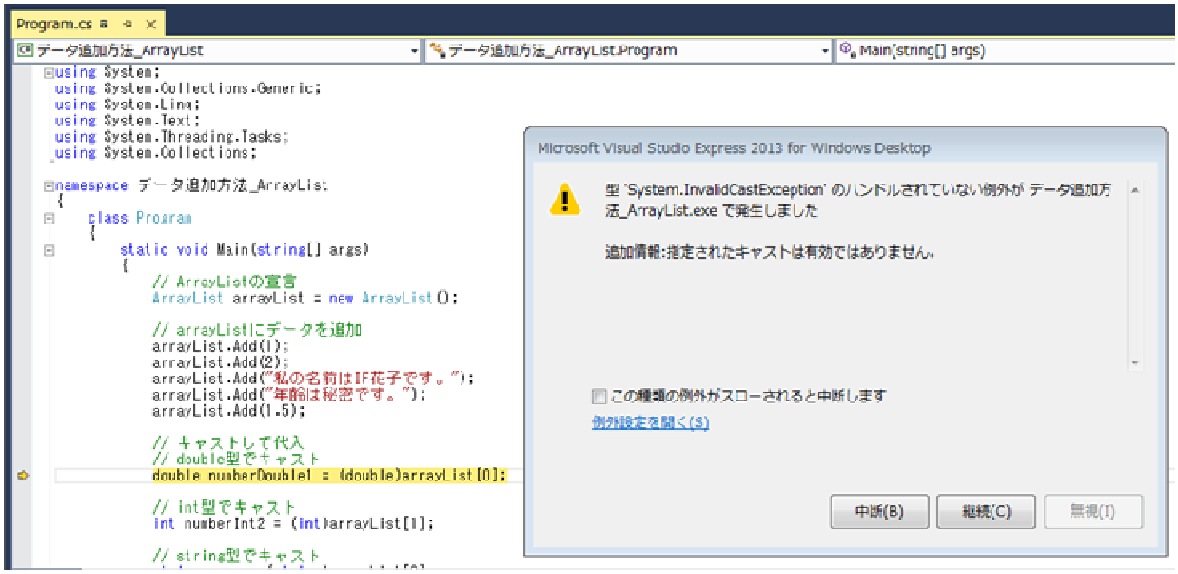
ArrayListに数字と文字列を合計5個追加し、それらを**ArrayList**から取り出して、画面に表示するプログラムを作成します。作成が終わったら実行してみましょう。

```
using System.Collections;

namespace データ追加方法_ArrayList
{
    class Program
    {
        static void Main(string[] args)
        {
            // ArrayListの宣言
            ArrayList arrayList = new ArrayList();

            // arrayListにデータを追加
            arrayList.Add(1);
            arrayList.Add(2);
            arrayList.Add("私の名前はIF花子です。");
            arrayList.Add("年齢は秘密です。");
            arrayList.Add(1.5);

            // キャストして代入
            // int型でキャスト
            int numberInt1 = (int)arrayList[0];

            // int型でキャスト
            int numberInt2 = (int)arrayList[1];

            // string型でキャスト
            string name = (string)arrayList[2];

            // string型でキャスト
            string age = (string)arrayList[3];

            // double型でキャスト
            double numberDouble = (double)arrayList[4];

            // 画面に表示
            Console.WriteLine(numberInt1.ToString());
            Console.WriteLine(numberInt2.ToString());
            Console.WriteLine(name);
            Console.WriteLine(age);
            Console.WriteLine(numberDouble.ToString());

            // Enterキーが押されるまで待機
            Console.ReadLine();
        }
    }
}
```

実行結果

```
1
2
私の名前はIF花子です。
年齢は秘密です。
1.5
```

もし入っているデータ型と異なるデータ型でキャストしてデータを取り出そうとした場合は、実行時に下記のようなエラーが出てしまうので注意しましょう。下記では、int型のデータをdouble型でキャストして取り出そうとしています。



練習1

以下のようなプロジェクトを作成してください。

ソリューション名：数値表示_ArrayList
プロジェクト名：数値表示_ArrayList
プロジェクトテンプレート：コンソール アプリケーション

ArrayList型変数を用意し、繰り返し文を使って1～10を追加してください。入れ終わったら、今度は**ArrayList**型変数に入っているint型のデータを取り出して、すべてのデータを画面に表示するプログラムを作成してください。

実行結果