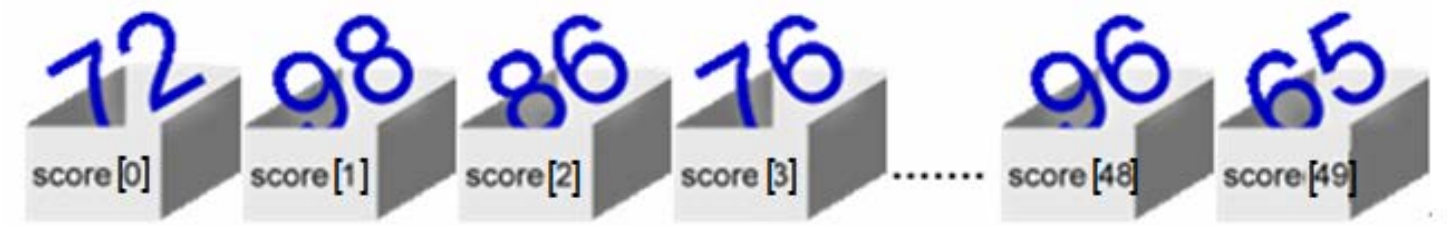
Array and Method

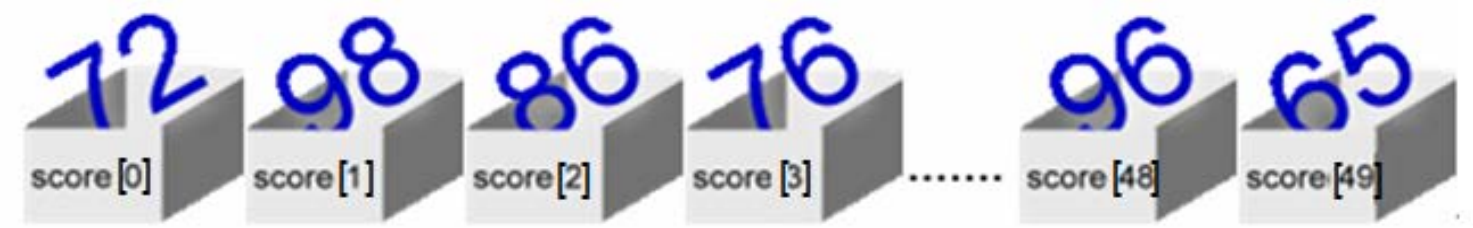
4-1 Array

- Array object can be simply called array
- Store data – like variable
- One variable only stores one data
- Array object stores many data in a series of memory area
- Array object is a set of data which has the same data type。



4-1 Array

- Combined with array elements
- Use array_name[index] to stand for an array element
- An array element is as same as a variable
- Use [] to identify every element
- Assign index, then C# automatically figures out the real position



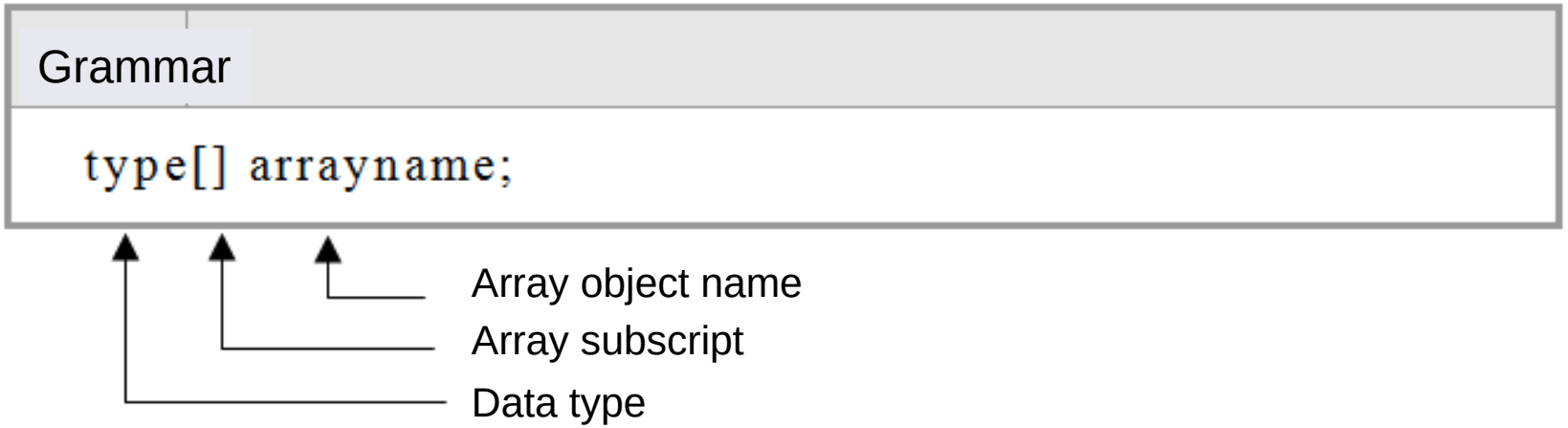
4-2 1-D Array

- Declaration is required before usage
- Target:
  1. Assign array’s name and data type
  2. Determine memory space used by array
  3. Store number of data

4-1.1. Array Declaration

- **1 [] ⇨ 1-D array**
- **2 [,] ⇨ 2-D array**

1. Declaration of 1-D array



Ex: declare myAry as an 1-D array

```
int[] myAry;
```

2. Create an 1-D array object

Grammar
`arrayname = new type[size];` // size is array's length

Ex: create an array object called myAry, this array object contains myAry[0] ~ myAry[4] five array elements:

```
myAry = new int[5];
```

3. C# allows combining declaration and realization into one statement, usage:

Grammar
`type[] arrayname = new type[n];`

Ex: combine declaration and realization of myAry into a single statement:

```
int[] myAry;
myAry = new int[5];
```

} Combine: **int[] myAry = new int[5];**

4-1.2. Initial Values of Array

- Every array element is a variable
- A declared array can use assign operator (=) to assign initial values to array elements
- If no initial value during declaration:
  ① number types: initial value is 0
  ② string types: initial value is empty string

2 ways to set initial values during declaration:

1. Use equal sign to assign initial values of array elements

```
int[] tAry = new int[5];
tAry[0] = 10;   tAry[1] = 20;   tAry[2] =30;   tAry[3] = 40;   tAry[4] = 50;
```

2. Use brackets to assign initial values, separated by comma

```
int []tAry = new int[ ] {10, 20, 30, 40, 50};
```

Ignore array length when assigning initial values

Array Declaration, Realization and Initial Value Assignment

Grammar

1. Declaration

   ```
   type[] arrayname;
   ```

2. Realization

   ```
   arrayname = new type[size];
   ```

3. Combine declaration and realization in one statement

   ```
   type[] arrayname = new type[n];
   ```

4. Combine declaration, realization and initial value in one statement

   ```
   type[] arrayname = new type[] { initiationlist };
   ```

4-1.3. Length Property of Array

Grammar
int varName = arrayName.Length;

【例】① long[] score = new long[4]; ⇦ Declare and realize score long array

int size = score.Length; ⇦ size =4

② double[] price=new double[] {1.2, 3.0, -0.0000005, 55, 1.6E14 };

int size = price.Length; ⇦ size = 5

③ string[] wk = {“Sun”, “Mon”, “Tue”, “Wed” , “Thu ”, “Fri” , “Sat”};

int size = wk.Length; ⇦ size = 7

4-2. Use for Loop

- **Read contents of tAry**

```
// FileName : ary1.sln
    int [] tAry = new int[] { 10, 20, 30, 40, 50 };
    for (int k = 0; k <= tAry.Length-1; k++)
            Console.Write (" tAry[{0}]={1}    ", k , tAry[k]);
    Console.Read();
```

Result:

tAry[0]=10 tAry[1]=20 tAry[2]=30
tAry[3]=40   tAry[4]=50

- **Exchange contents of tAry[1]=20 and tAry[3]=40**
- **Usage:**

```
temp= tAry[1]  ;
 tAry[1]= tAry[3];
 tAry[3] =temp;
```

4-2.2. Use foreach Loop

- **Combined with array name, array data type and the same variable name:**
  ⇨ read array elements one by one
  ⇨ avoid the consideration of index

Grammar

```
foreach (type element in group) {
    [statements]
    [break;]
    [statements]
};
```

Ex: display the values of array elements in the string array continuously on screen

```
string[] msg = new string[4]{"Time", "is", "money",  "."};
foreach (string word in msg)
    Console.Write("Δ{0} ", word);   // Δ : a space
Console.Read();
```

Result: "Time is money ."

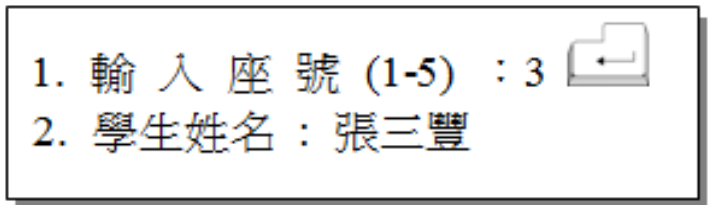
Practice:

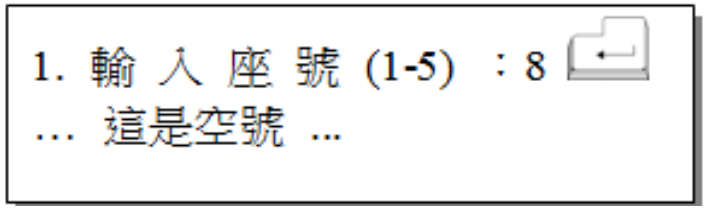
Write a program to store names and serial numbers, as shown in the list. As the serial number is inputted, the related name is shown. If the inputted number is out of range, show ”… This serial number is unavailable …”.

S/N	Name	Array element
1	錢一文	name[0]
2	孫二娘	name[1]
3	張三豐	name[2]
4	李四維	name[3]
5	王五金	name[4]

Hint: the subscript’s difference between serial number and array element is 1

Result:





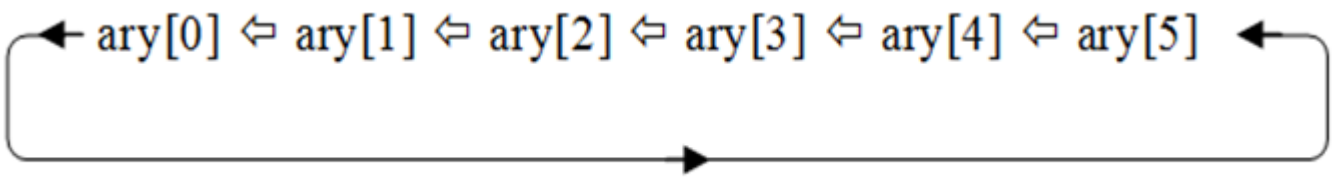
FileName : readAry1.sln

```
05  static void Main(string[] args)
06  {
07          string[] name = new string[5];
09          name[0]="錢一文";  name[1]="孫二娘"; name[2]="張三豐";
10          name[3]="李四維";  name[4]="王五金";
11          int no;
12          Console.Write(" 1. 輸入座號(1-5) ： ");
13          no = int.Parse(Console.ReadLine());
14          if(no>=1 && no<=5)
15              Console.WriteLine(" 2. 學生姓名 : {0} ", name[no-1]);
16          else
17              Console.WriteLine(" ... 這是空號 ... ");
18          Console.Read();
19  }
```

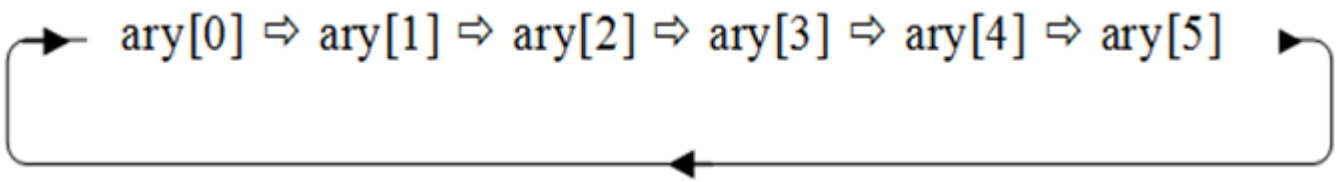
Practice:shift.sln

Write a program. The array has initial values {111, 222, 333, 444, 555, 666}. First display the initial values of the array. Second shift the array to left and display the result. Third shift the array to right and display the result.

1. Shift left, array element rotates to left as shown:



2. Shift right, array element rotates right as shown:



FileName : shift.sln

```
05  static void Main(string[] args)
06  {
07      string str1 = "";
08       int[] ary = new int[] { 111, 222, 333, 444, 555, 666 };
10       Console.Write(" 原始陣列 : ");
11       foreach (int num in ary)
12           str1 += num.ToString() + "  ";
13      Console.WriteLine("{0}", str1);
14       // 陣列左移
15      int temp;
16   temp = ary[0];
17
17      for (int i = 0; i <= ary.Length - 2; i++)   // 6-2 = 4 , 0 ~ 4
18           ary[i] = ary[i + 1];          // ary[0] = ary[1]= 222
19      ary[ary.Length - 1] = temp;  // ary[6-1=5]= temp = 111
```

	0	1	2	3	4	5
Original:	111	222	333	444	555	666
Shift left:	222	333	444	555	666	111
Shift right:	111	222	333	444	555	666

```
20          str1 = "";
21          Console.Write(" 陣列左移 : ");
22          foreach (int num in ary)
23              str1 += num.ToString() + "  ";
24          Console.WriteLine("{0}", str1);
25          // 陣列右移
26          Console.Write(" 陣列右移 : ");
27          temp = ary[ary.Length - 1];          // 6-1 = 5 , temp = 111
28          for (int i = ary.Length - 1; i >= 1; i--) // I = 6-1 = 5
29              ary[i] = ary[i - 1];                  //  ari[5] = ari[5-1=4]= 666
30          ary[0] = temp;                            // ary[0]=temp = 111
31          str1 = "";
32          foreach (int num in ary)
33              str1 += num.ToString() + "  ";
34          Console.WriteLine("{0}", str1);
35          Console.Read();
36      }
37  }
38 }
```

4-3 1-D Array Methods

1. Array.Sort() Method

Grammar

```
Array.Sort (arrayname1 [,arrayname2] );
```

Ex: an integer array called num, use Array.Sort method to increasing sorting.

```
int[] num = {12, 45, 76, -3, 48, 93};
Array.Sort(num);
```

Result: arranged num array: {-3, 12, 45, 48, 76, 93}

- **If first argument is arranged increasingly, second one is also arranged increasingly.**

Ex: try to arrange the following table increasingly

Name (myValue)	Score (myKeys)
mary	90
jack	60
tom	50
david	70
grace	80

排序前:

註標	對應值	索引鍵
k=0	mary :	90
k=1	jack :	60
k=2	tom :	50
k=3	david :	70
k=4	grace :	80

排序後

註標	對應值	索引鍵
k=0	tom :	50
k=1	jack :	60
k=2	david :	70
k=3	grace :	80
k=4	mary :	90

FileName : arySort.sln

```
05    static void Main(string[] args)
06    {
07      int[] myKeys = new int[] { 90, 60, 50, 70, 80 };
08      string[] myValues= new string[] { "mary", "jack", "tom", "david", "grace" };
10      Console.WriteLine(" 排序前: ");
11      Console.WriteLine(" 註標  對應值   索引鍵 ");
12      for (int k = 0; k <= myKeys.Length-1 ; k++)
13          Console.Write(" k={0} \t{1} :\t {2} \n", k, myValues[k],myKeys[k]);
14      Console.WriteLine();
16      Array.Sort(myKeys, myValues);
17      Console.WriteLine(" 排序後: ");
18      Console.WriteLine(" 註標  對應值   索引鍵 ");
19      for (int k = 0; k <= myKeys.Length - 1; k++)
20        Console.Write(" k={0} \t{1} :\t  {2} \n", k, myValues[k], myKeys[k]);
21      Console.Read();
22    }
```

Array.Reverse() Method

- **Reverse the arrange of a 1-D array. In other words, the first element will be the last one**

Grammar
Array. Reverse (arrayname);

Ex: use Array.Sort() method to sort array increaseingly; then use Array.Reverse() method to change the array arranged decreasingly.

```
int[] score = {98, 78, 85, 64};
Array.Sort(score);          ⇦ score = {64, 78, 85, 98}
Array. Reverse (score);     ⇦ score = {98, 85, 78, 64}
```

Array.IndexOf() Method

- **Search from specific index, find matching elements and return the index of the first occurrence**

Grammar
Array.IndexOf(arrayName, variable[, start[, length]]);

Ex:
string[] arr = {“a”, “b”, “c”, “b”};
int pos = Array.IndexOf(arr, “b”); // result: pos = 1

Array.Resize() Method

- **If the length of dynamic array is reset**
  ⇨ contents of array elements are cleared
- **Array.Resize() can be used if we want to reserve array's data and modify the size of array**

Grammar
Array.Resize(ref arrayname, newsize);

Ex: change the length of score array and add an array element.

```
int score = {20, 30, 40, 50, 60};
Array.Resize(ref score, score.Length + 1);
```

Result: score = {20, 30, 40, 50 ,60, 0};

4-4 Multiple-dimension Array

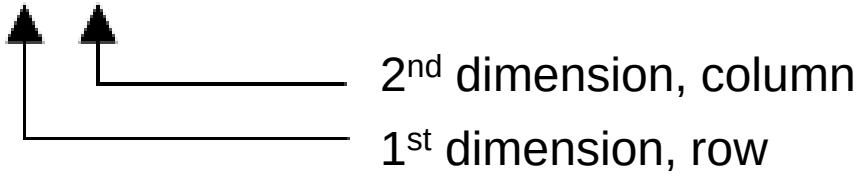
- 2 subscripts – 2-D array
- 3 subscripts – 3-D array
- Application of 2-D array can be seen in matrix, score table, sales achievement, stock quotation table, etc.

Declaration of 2-D Array

- 2-D array uses row and column to represent 2 dimensions
- Dimensions are separated by comma(,)

1. How to declare a 2-D array:
   dataType[,] arrayName;

2. How to realize a 2-D array:
   arrayName = new dataType[n, m];



3. Combine declaration and realization in 1 statement:

dataType[,] arrrayName = new dataType[n, m];

Ex: int[,] ary = new int[3, 4];

declare and realize a 2-D array named ary, the array has 4 columns and 3 rows

	Column 0	Column 1	Column 2	Column 3
Row 0	ary[0, 0]	ary[0, 1]	ary[0, 2]	ary[0, 3]
Row 1	ary[1, 0]	ary[1, 1]	ary[1, 2]	ary[1, 3]
Row 2	ary[2, 0]	ary[2, 1]	ary[2, 2]	ary[2, 3]

4. Combine declaration, realization and initial value assignment:

dataType[,] arrayName = { {initiationList}, {…}, {…} };

dataType[,] arrayName = new dataType[,] { {initiationList}, {…}, {…} };

Ex: create a 2-D array named score with initial values:

	Column 0	Column 1	Column 2	Column 3
Row 0	80	75	60	76
Row 1	65	67	62	80
Row 2	83	82	87	94

Result: `int[,] score = {{80,75,60,76},{65,67,62,80},{83,82,87,94}};`

0th row 1st row 2nd row

or

`int[,] score = new int [,] {{80,75,60,76},{65,67,62,80},{83,82,87,94}};`

0th row 1st row 2nd row

Practice:

Use array to store the following data listed in the table. Show the student's name and scores when the serial number is inputted

		0	1	2
	S/N	Name	Chinese	English	Math
0	1	錢一文	80	75	60
1	2	孫二娘	65	67	62
2	3	張三豐	100	93	91
3	4	李四維	98	25	50
4	5	王五行	83	82	87

Result:

```
輸 入 座 號 (1-5) ：2
1. 姓名 ：孫二娘
2. 國文： 65
3. 英文 ：67
4. 數學 ：62
```

FileName : readAry2.sln

```
05      static void Main(string[] args)
06      {
07          string[] name = {"錢一文","孫二娘","張三豐","李四維","王五行"};
08          int[,] score = {{80,75,60},{65,67,62},{100,93,91},{98,25,50},{83,82,87}};
09          int no;
10          Console.Write(" 輸入座號(1-5)：");
11          no = int.Parse(Console.ReadLine());
12          if (no >= 1 && no <= 5)
13          {
14              Console.WriteLine("1. 姓名 : {0} ", name[no-1]);
15              Console.WriteLine("2. 國文 : {0} ", score[no-1,0]);
16              Console.WriteLine("3. 英文 : {0} ", score[no-1,1]);
17              Console.WriteLine("4. 數學 : {0} ", score[no-1,2]);
18          }
19          else
20              Console.WriteLine(" ... 空號 ... ");
21          Console.Read();
22      }
```

GetLength() Method

- **Can use GetLength() method to get number of elements in specific dimension (over 2-D)**

Grammar
arrayname.GetLength(dimension); Target dimension of array

Ex1:

```
int[,] ary = new int[3, 4]; // used in multiple dimension array
int element;
element = ary.GetLength(0); // return 3
element = ary.GetLength(1); // return 4
```

Ex2:

```
int[] ary = new int[6]; // used in 1-D array
int element;
element = ary.GetLength(0); // return 6
```

GetLowerBound(), GetUpperBound() method

- Get the upper bound and the lower bound of an array (scope)
- C# array index starts from 0

Grammar

```
arrayname.GetLowerBound (dimension);
arrayname.GetUpperBound (dimension);
```

Ex1:

```
int[,,] score = new int[3, 15, 20]; // Used in 3-D array
int index;
index = score.GetUpperBound(0);  ⇦ Return the 1st upper bound: 2
index = score.GetUpperBound(1);  ⇦ Return the 2nd upper bound: 14
index = score.GetUpperBound(2);  ⇦ Return the 3rd upper bound: 19
```

Ex2:

```
int[] score = new int[6];        ⇦ Used in 1-D array
int index;
index = score.GetUpperBound(0);  ⇦ Return the upper bound: 5
```

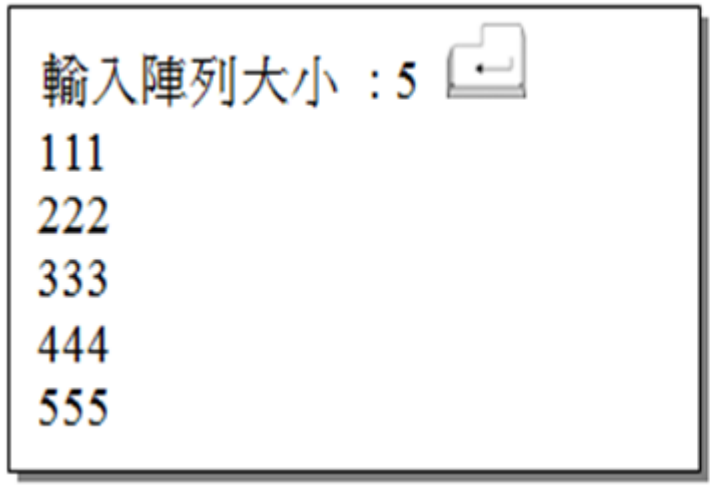
4-5 Dynamic Array

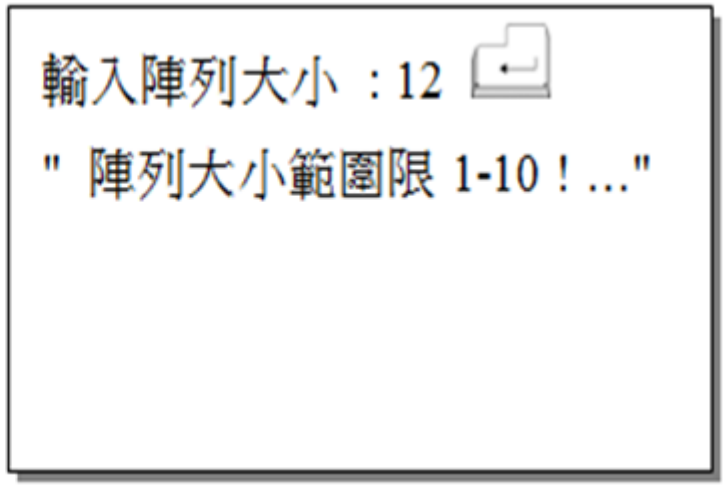
- Size of array need not be defined in advance
- Can be set when the program is running, if needed
- If the size has been defined, re-definition is allowed
- Advantage
  Flexible
- Disadvantage
  Out of memory possibility during running
  ⇨ Cause crash if no special handler and the program cannot continue

Practice: dynamicAry.sln

Try to write a program get input size from keyboard, then construct an array with size elements, the scope is 1 ~ 9

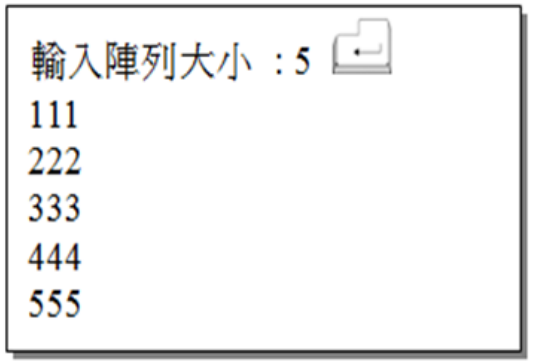
Result:

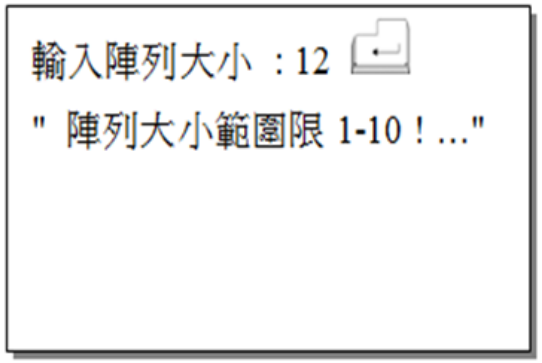




FileName : dynamicAry.sln

```
05      static void Main(string[] args)
06       {
07          int[] data;
08          string str1 = "";
09          int size;
10          Console.Write(" 輸入陣列大小 : ");
11          size = int.Parse(Console.ReadLine());
12          if (size >= 1 && size <= 9)
13           {
14              data = new int[size];
15              for (int i = 0; i < data.Length ; i++)
16                   data[i] = 111 * (i + 1);
17              foreach (int num in data)
18                   str1 += num.ToString() + "\n";
19              Console.WriteLine("{0}", str1);
20           }
21          else
22             Console.WriteLine(" 陣列大小範圍限1-10 ! …");
23          Console.Read();
24        }
```





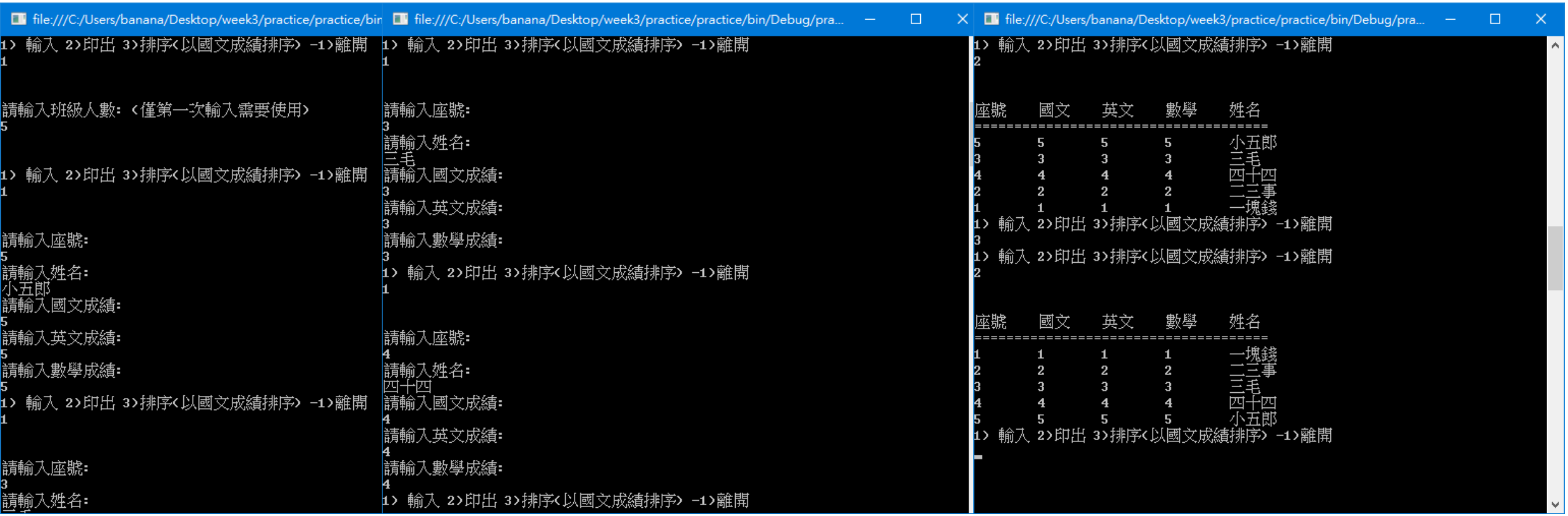
Practice

- Hint :
- 1. For the first time, only put in the numbers of people.

  Start to put in the scores at the second time.
- 2. When printing, print out all the array information below(refer to the next page) . Use 0 to represent the blank spaces.
- 3. To put them in order, the built in C# Sort function is available. Users are able to create their own Sort function as well.

Practice

- Use dynamic array to manage the size of user enter.
- Use sort function to arrange array.
- Use while loop to keep program continually.


file:///C:/Users/banana/Desktop/week3/practice/practice/bir
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開
1
請輸入班級人數: (僅第一次輸入需要使用)
5
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開
1
請輸入座號:
5
請輸入姓名:
小五郎
請輸入國文成績:
5
請輸入英文成績:
5
請輸入數學成績:
5
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開
1
請輸入座號:
3
請輸入姓名:
file:///C:/Users/banana/Desktop/week3/practice/practice/bin/Debug/pra...
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開
1
請輸入座號:
3
請輸入姓名:
三毛
請輸入國文成績:
3
請輸入英文成績:
3
請輸入數學成績:
3
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開
1
請輸入座號:
4
請輸入姓名:
四十四
請輸入國文成績:
4
請輸入英文成績:
4
請輸入數學成績:
4
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開
file:///C:/Users/banana/Desktop/week3/practice/practice/bin/Debug/pra...
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開
2
座號 國文 英文 數學 姓名
5 5 5 5 小五郎
3 3 3 3 三毛
4 4 4 4 四十四
2 2 2 2 二三事
1 1 1 1 一塊錢
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開
3
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開
2
座號 國文 英文 數學 姓名
1 1 1 1 一塊錢
2 2 2 2 二三事
3 3 3 3 三毛
4 4 4 4 四十四
5 5 5 5 小五郎
1) 輸入 2)印出 3)排序(以國文成績排序) -1)離開

The End

Take a Break …

4-6 Method

- **Use method to organize the source code**
  advantage:

1. **Separate a huge program into several small function units**
   ⇨ written from different people, raises the programming efficiency
2. **Program segment with the same function can be written in a function**
   ⇨ can be used in different places
   ⇨ less modification to apply to other programs
3. **Shorten program lines and simplify the logic**
   ⇨ raises readability

2 kinds of method:

1. system integrated
2. programmer designed:
   - ① event handler
   - ② user defined

4-7 System Integrated Random Methods

- Venders integrates math formula, string function and so on in the programming language
- C# offers .NET Framework integrated classes, only give the parameters and get the result
  ⇨ called system integrated methods

Random Methods

Grammar and function	Description
1. Next method Grammar: rnd.Next() Function: return an unsigned integer random between 0 and 2147483647 Grammar: rnd.Next(n) Function: return an unsigned integer random smaller than n Grammar: rnd.Next(n1, n2) Function: return an integer random n1≤n<n2	Ex1: num = rnd.Next(); Result: num = 556146380 Ex2: n = rnd.Next(49); Result: n = 43 // generate 0 ~ 48 random Ex3: n = rnd.Next(-5, 6) Result: n = -3 // generate -5 ~ 5 random

2. NextBytes method
 	
Grammar: rnd.NextBytes(ary);
Function: ary array element is put with integer 0 ~ 255
	Ex:
	Random rnd = new Random();
	byte[] ary = new byte[4];
	rnd.NextBytes(ary);
	Result:
	ary[0] = 136
	ary[1] = 90
	ary[2] = 174
	ary[3] = 41

3. NextDouble method Grammar: rnd.NextDouble(); Function: return the random between 0.0 and 1.0	Ex: Double d = rnd.NextDouble(); Result: d = 0.210550215658988

- Random object has to be realized before using random methods

- Example steps:

1. Declare and realize Random object ranObj

```
Random radObj = new Random();
```

2. ① ranObj object generate random number between 0 and 4 and assign to ranNo:

```
int ranNo = ranObj.Next(5)
```

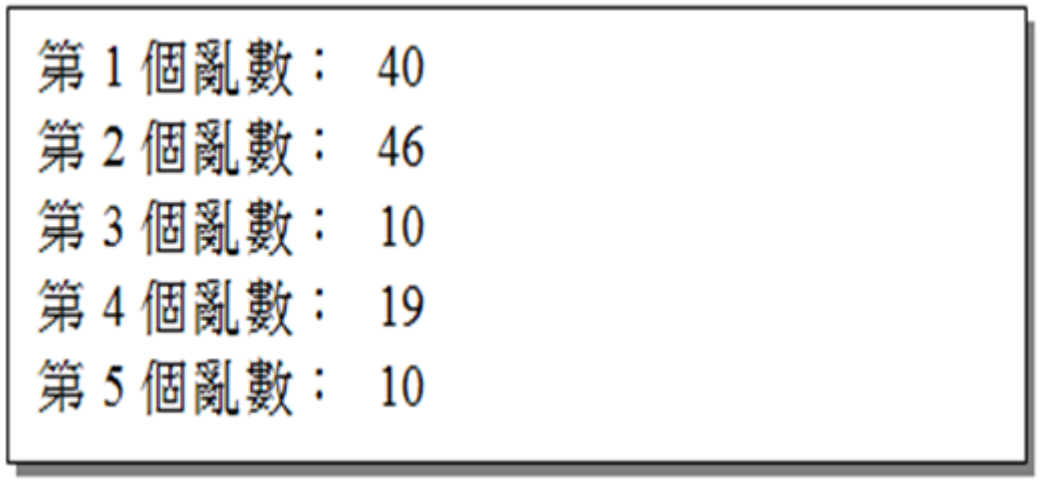
② ranObj object generate random number between 10 and 50 and assign to ranNo:

```
int ranNo = ranObj.Next(10,51)
```

Practice: ranObj.sln

Try to use for loop and Random class to generate 5 random numbers between 10 and 50, the result as shown in figure:

Result:



```
// FileName : ranObj.sln
05   static void Main(string[] args)
06   {
07       Random ranObj = new Random();
08       int ranNo ;
09       for (int i=1 ; i <= 5; i++)
10       {
11            ranNo = ranObj.Next(10, 51);
12            Console.WriteLine("第{0}個亂數 : {1}" ,  i , ranNo);
13       }
14      Console.Read();
15    }
```

4-8 User-defined Method

- **Methods defined by programmers**
- **Have to write new source code**
- **The methods have to be defined before using them**

Grammar

```
[fixWord] [static] returnType methodName([arguments]) {
        [statements;]
        [return expression;]
}
```

Practice:

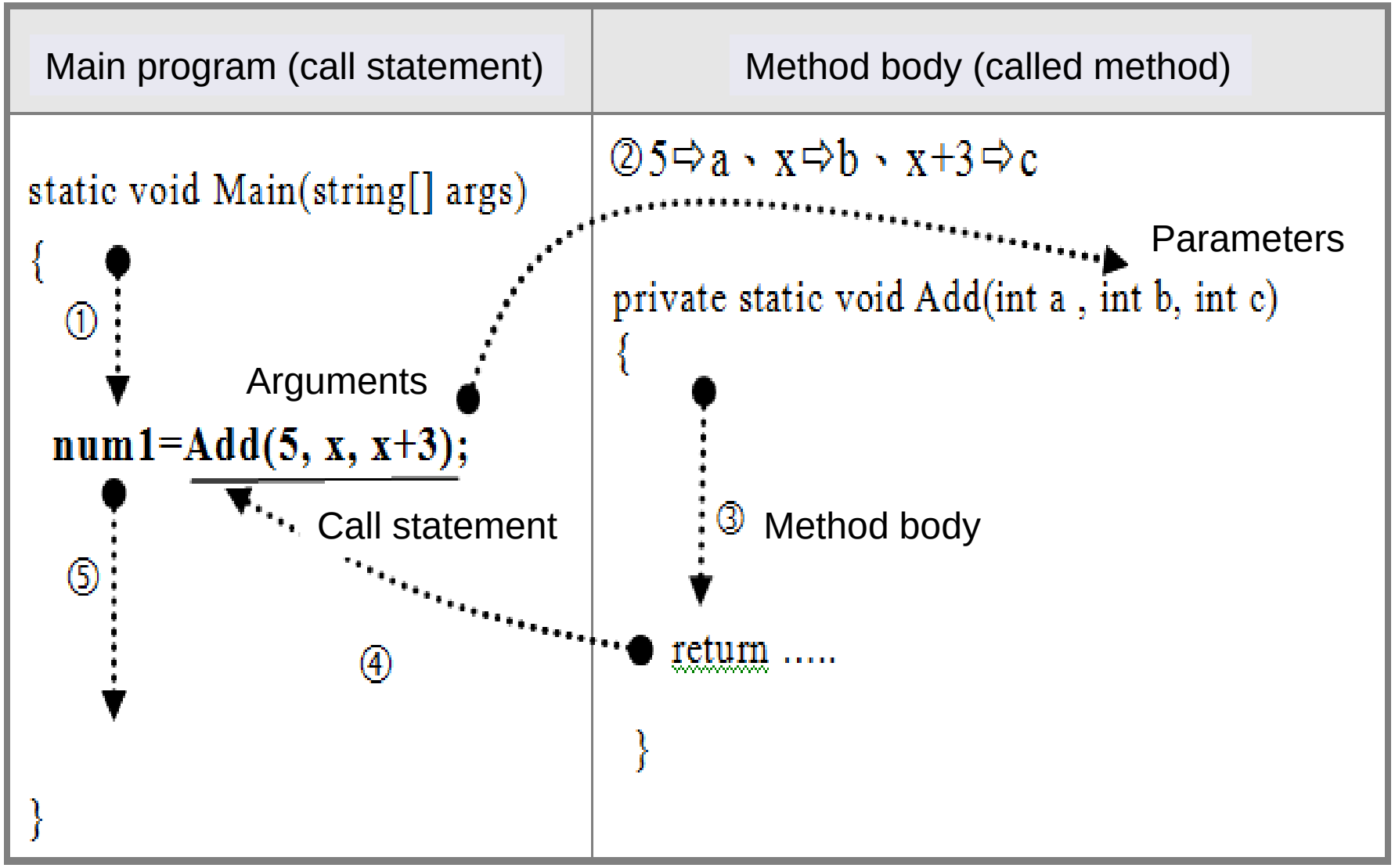
Define a static method called Add(), the parameter is an integer n1. The input value pluses 20 and assign to local variable sum. Finally return sum.

```
private static int Add (int n1) {   // static method, private can be ignored
    int sum;                        // local variable
    sum = n1 +20;
    return sum;                     // return value
}
```

How to Call Method in the Same Class

Grammar

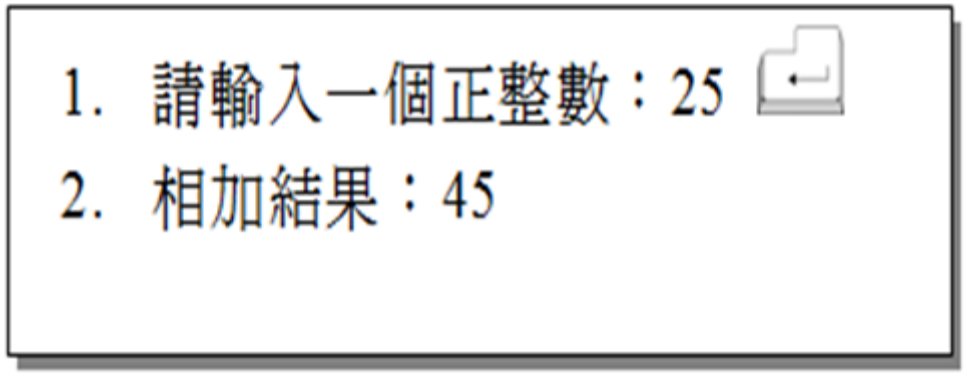
1. [objectName.]methodName([arguments]);
2. variable = [objectName.]methodName([arguments]);


Main program (call statement)
Method body (called method)
static void Main(string[] args)
{
①
num1=Add(5, x, x+3);
⑤
}
Arguments
Call statement
④
②5⇨a、x⇨b、x+3⇨c
Parameters
private static void Add(int a , int b, int c)
{
③ Method body
return .....
}

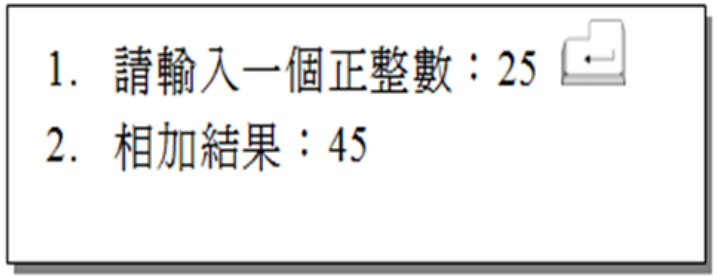
Practice:: privateStatic.sln

Use the Add() static method in the previous practice. The method in the same object can call Add() and get the result. The Main function calls Add() and show the returned result

Result:



```
// FileName : privateStatic.sln
05   private static int Add(int n1)
06   {
07         int sum;          // local variable
08         sum = n1 +20;
09         return sum;       // return value
10   }
11
12   static void Main(string[] args)
13   {
14         Console.Write("1. 請輸入一個正整數：");
15         int num1 = int.Parse(Console.ReadLine());
16         int num2 = Add( num1);
17         Console.WriteLine("2. 相加結果：{0} ", num2);
18         Console.Read();
19   }
```



How to Call Method in Different Class

Grammar

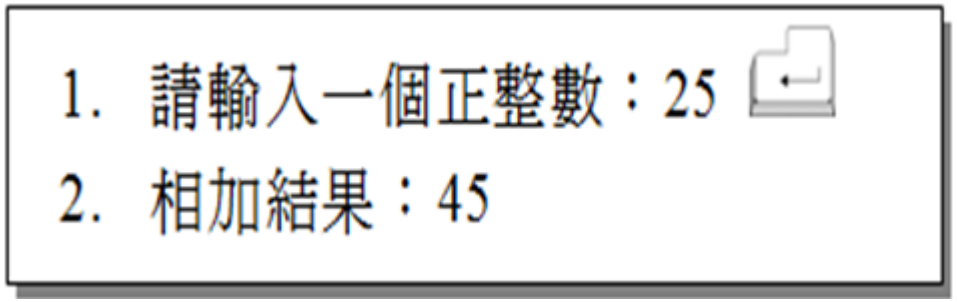
1. className.staticMethodName([arguments]);
   or
   objectName.methodName([arguments]);

2. variable = className.staticMethodName([arguments]);
   or
   variable = objectName.methodName([arguments]);

Practice:classAdd.sln

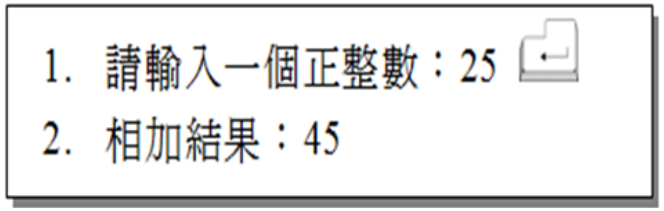
From the previous practice, Main() method calls Add() method in different class by className.methodName()

Result:



// FileName :classAdd.sln

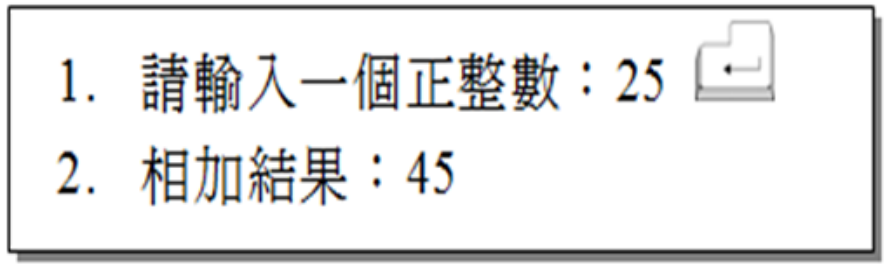
```
01 namespace classAdd
02 {
03     class Program        // Program類別
04     {
05         static void Main(string[] args)
06         {
07             Console.Write("1. 請輸入一個正整數：");
08             int num1 = int.Parse(Console.ReadLine());
09             int num2 = Class1.Add( num1); //有建立 static 所以可以直接哪個 className.methodName()
10             Console.WriteLine("2. 相加結果：{0} ", num2);
11             Console.Read();
12         }
13     }
14     public class Class1      // 建立Class1類別
15     {
16         public static int Add(int n1)  // Class1類別內的Add方法
17         {
18             int sum;
19             sum = n1 + 20;
20             return sum;     // 傳回值
21         }
22     }
23 }
```



Practice: :objAddMethod.sln

From the previous practice, Main() method calls Add() method in different object by className.methodName()

Result:



```
// FileName :objAddMethod.sln
01 namespace objAddMethod
02 {
03    class Program
04    {
05       static void Main(string[] args)
06       {
07          Class1 obj = new Class1();          // 建立Class1類別的物件
08          Console.Write("1. 請輸入一個正整數：");
09          int num1 = int.Parse(Console.ReadLine());
10          int num2 = obj.Add( num1);   // 呼叫obj物件的Add方法
11          Console.WriteLine("2. 相加結果：{0} ", num2);
12          Console.Read();
13       }
14    }
15    // ------------------------------------------------------------------------------
16    public class Class1        // 建立Class1類別
17    {
18       public int Add(int n1)  //未加ststic無法直接呼叫必須先建物件實體才能呼叫
19       {
20          int sum;
21          sum = n1 + 20;
22          return sum;       // 傳回值
23       }
24    }
25 }
```

1. 請輸入一個正整數：25
2. 相加結果：45

Call by Value

- Function parameter gets the duplication of function argument’s value
- Parameter and argument have different memory addresses. Parameter changes but it does not affect the argument value
- When call by value
  argument – constant, variable, expression
  parameter – variable only

Call by Value

Parameter	Memory address	Memory content	Argument
a	100000	10	
b	100004	7	
	100008	10	x
	10000c	7	y

Practice:

Try to write a program to monitor the status of parameters and arguments. In Main(), assume that (a, b) is (1, 10) and call passValue(a, b) by value. In passValue(), the parameter x pluses 5 and the parameter y pluses 10. Show the value of variables before function call, during function call and after function call

Result:

```
====== 傳值呼叫 ======
1.  呼叫前：a = 1   b = 10
2.  呼叫中：x = 6   y = 20
3.  呼叫後：a = 1   b = 10
```

```
// FileName :passValue.sln
05   private static void passValue(int x, int y)
06   {
07          x = x + 5;        // x= 1+6 = 6
08          y = y + 10;       // y= 10+10 = 20
09          Console.WriteLine (" 2. 呼叫中 : x =  {0}  y = {1}" , x, y );
10   }
11   // ----------------------------------------------------------------------
12   static void Main(string[] args)
13   {
14          int a, b;
15          a = 1;
16          b = 10;
17          Console.WriteLine("  ===== 傳值呼叫  === ");
18          Console.WriteLine(" 1. 呼叫前 : a =  {0}  b = {1}  ", a, b);
19          passValue(a,b);
20          Console.WriteLine(" 3. 呼叫後 : a =  {0}  b = {1}  ", a, b);
21          Console.Read();
22   }
```

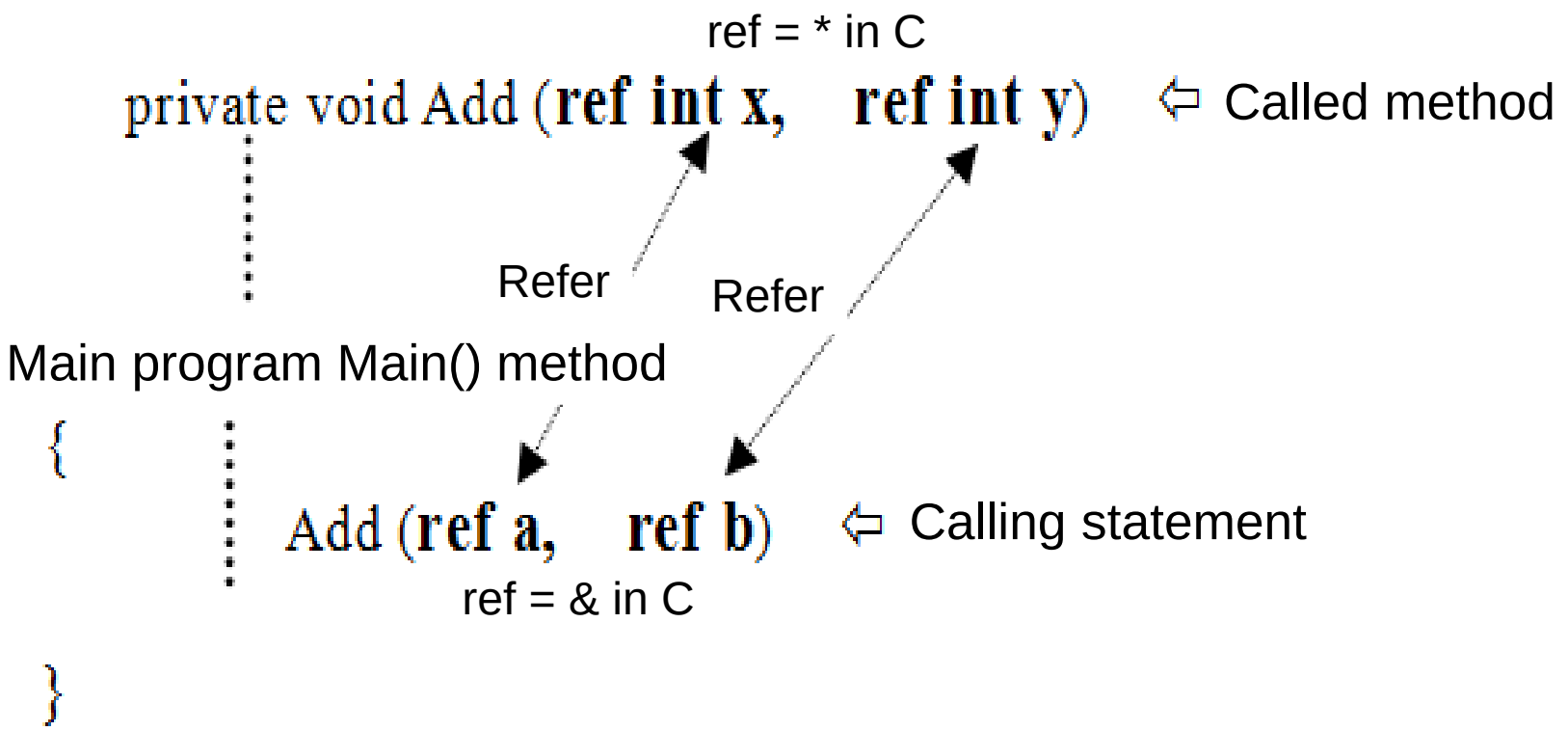
```
===== 傳值呼叫 ======
1. 呼叫前：a = 1   b = 10
2. 呼叫中：x = 6   y = 20
3. 呼叫後：a = 1   b = 10
```

Call by Reference

- **Function parameter gets memory address from the function argument**
- **The parameter's value changes if the argument's value changes**

Parameter	Memory addr	Memory value	Argument
a	100000		x
b	100004		y
	100008		
	10000c		

Define method main body:



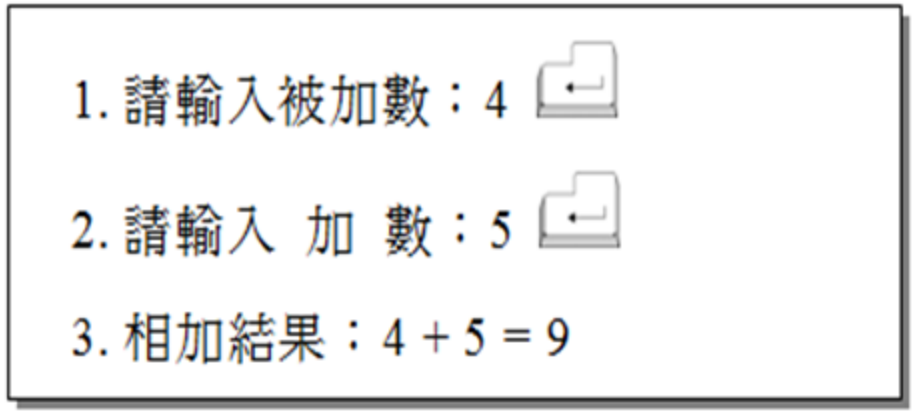
Call by Reference of c

```
void swap (int *c , int *d){
        int temp=*c;
        *c=*d;
        *d=temp;
}
int main(){
        int a=5,b=10;
        swap(&a,&b);
        printf(" %d %d ", a,b);

}

 void swap (int *c , int *d){
        int temp=*c;
        *c=*d;
        *d=temp;
}
int main(){
        int a=5,b=10;
        swap(&a,&b);
        printf(" %d %d ", a,b);
```

Practice:sum.sln

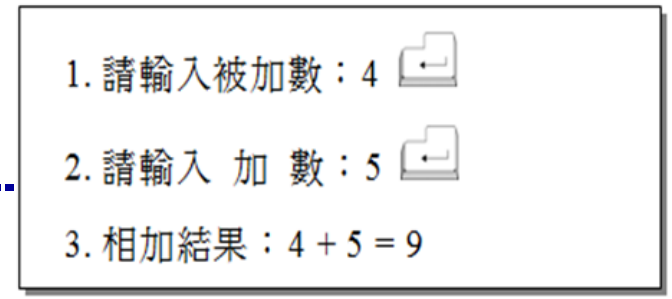
Try to write a program which calls function by reference. The program gets summand and addend from keyboard. Call Add() by value, plus 2 parameters and return the result

Result:



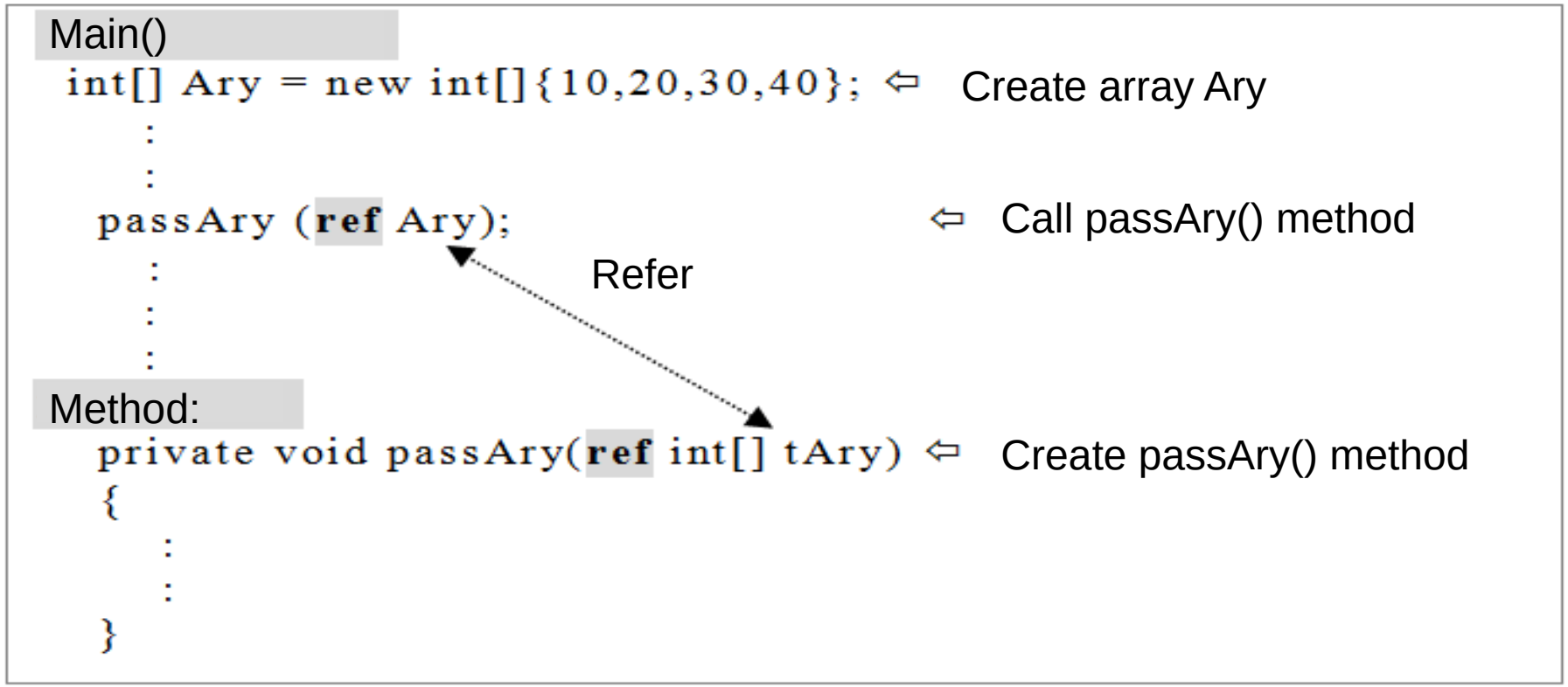
FileName : sum.sln

```
05        private static void passRef(float a, float b, ref float c)
06        {
07            c = a + b;  /// c = 4 +5 = 9
08        }
09        // ------------------------------------------------------------
10        static void Main(string[] args)
11        {
12             float sum=0;
13             Console.Write("1. 請輸入被加數 : ");
14             float num1 = float.Parse(Console.ReadLine());
16             Console.Write("2. 請輸入 加 數 : ");
17             float num2 = float.Parse(Console.ReadLine());
18             passRef(num1, num2, ref sum);
19            Console.WriteLine("3. 相加結果 : {0} + {1} = {2} " , num1,
                                          num2 ,sum );
20             Console.Read();
21          }
```



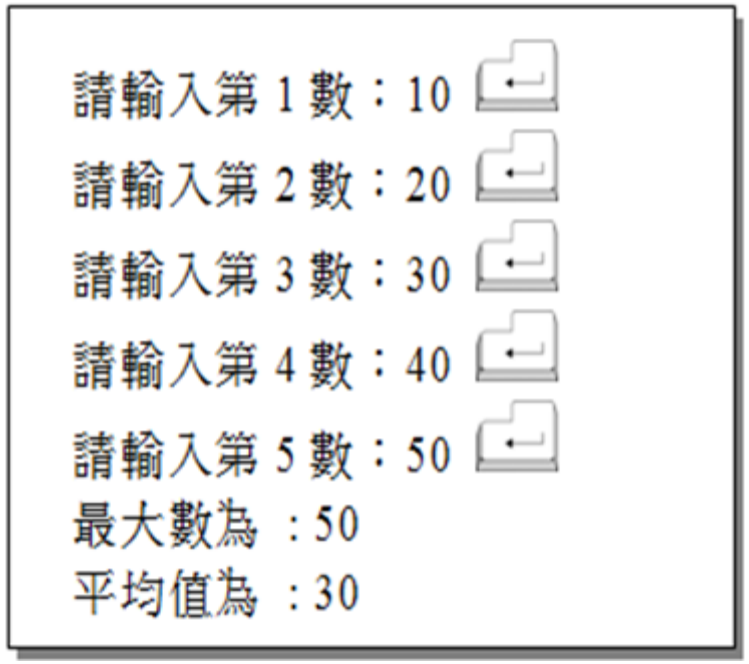
4-9 Passing Array as Parameter

- **Array elements are passed by value**
- **Array are passed by reference. ref keyword is required**



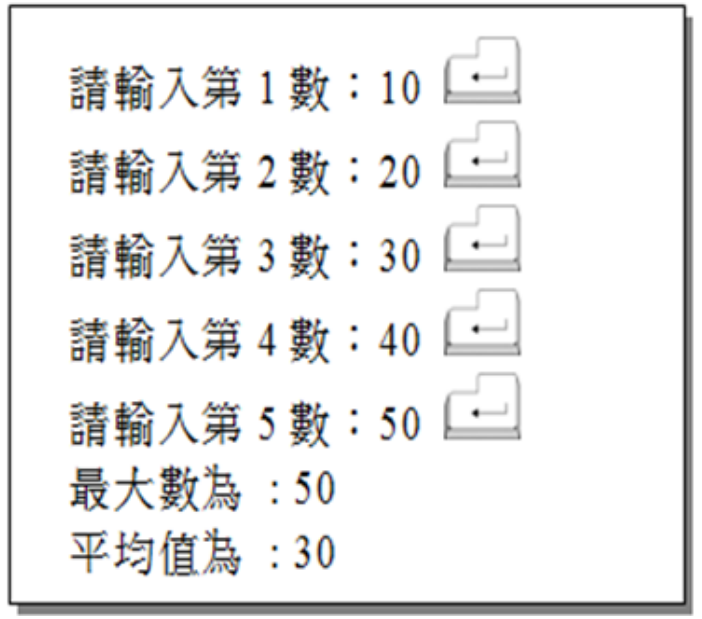
Practice:maxAvg.sln

Try to get 5 integer input from keyboard. Use FAvg() method to get the average of numbers in the array. Use FMax() method to get the maximum number. The array is passed by reference. The following figure shows the result:



FileName : maxAvg.sln

```
05      private static float FMax(ref float[] tAry)
06      {
07          float tMax = tAry[0]; // 先假設陣列元素0 為最大值
08          for (int i = 0; i < tAry.Length; i++)
09              if (tAry[i] > tMax)
10                  tMax = tAry[i];
11          return (tMax);
12      }
13      // ----------------------------------------------------------
14      private static float FAvg(ref float[] tAry)
15      {
16          float tAvg = new float() ;
17          foreach (float n in tAry)
18              tAvg += n;
19          return  tAvg/tAry.Length;
20      }
21      // ----------------------------------------------------------------------
```



```
22        static void Main(string[] args)
23        {
24            float[] numAry = new float[5];
25
26            for (int i = 0; i < 5; i++)
27            {
28                Console.Write("請輸入第 {0} 數：", i + 1);
29                numAry[i] = float.Parse(Console.ReadLine());
30            }
31            Console.WriteLine("最大數為 ：{0} ", FMax(ref numAry));
32            Console.WriteLine("平均值為 ：{0} ", FAvg(ref numAry));
33            Console.Read();
34
35        }
36    }
37 }
```

4-10 Overloading

- **Allow one method with many definitions**
- **Automatically find the method with matched number of arguments and data types when the method is called**
- **Define an overloading method called product, 3 ways to call the method:**

Calling expression	Defined methods (method main part)
`product();`	`product()`
`product(6);`	`product(x)`
`product(5, 3);`	`product(x, y)`

Practice:overload1.sln

Use overloading to use Area() to calculate the area of circle, rectangle and the surface area of cube

1. Area(a)
   only 1 argument, the radius, calculate the area of circle
2. Area(a, b)
   2 arguments, length and width, calculate the area of rectangle
3. Area(a, b, c)
   3 arguments, length and width and height, calculate the surface area of cube

1. 請輸入半徑：5
1. 圓形的面積為: 78.5

---

2. 請輸入矩形的長度: 10
2. 請輸入矩形的寬度: 3
2. 矩形的面積為: 30

---

3. 請輸入立方體的長度: 10
3. 請輸入立方體的寬度: 3
3. 請輸入立方體的高度: 2
3. 立方體的表面積為: 112

FileName : overload1.sln

```
01  namespace overload1
02 {
03   class Program
04   {
05      private static double Area(double r)
06      {
07         return r*r*3.14;
08      }
09      // ------------------------------------------------------------------------------
10      private static double Area(double x, double y)
11      {
12         return x * y;
13      }
14      // ------------------------------------------------------------------------------
15      private static double Area(double x, double y, double z)
16      {
17        return 2 * (x * y + y * z + z * x);
18      }
```

```
20      static void Main(string[] args)
21      {
22          Console.Write("1. 請輸入半徑：");
23          double a=double.Parse(Console.ReadLine());
24          Console.WriteLine ( "1. 圓形的面積為：{0} " , Area(a));
25          Console.WriteLine(" -------------------------");
26         // ------------------------------------------------------------
27          Console.Write("2. 請輸入矩形的長度：");
28          double length1 = double.Parse(Console.ReadLine());
29          Console.Write("2. 請輸入矩形的寬度：");
30          double width1 = double.Parse(Console.ReadLine());
31          Console.WriteLine ( "3. 矩形的面積為：{0} " , Area(length1,
             width1));
32          Console.WriteLine(" -------------------------");
```

```
34          Console.Write("3. 請輸入立方體的長度：");
35          double length2 = double.Parse(Console.ReadLine());
36          Console.Write("3. 請輸入立方體的寬度：");
37          double width2 = double.Parse(Console.ReadLine());
38          Console.Write("3. 請輸入立方體的高度：");
39          double height2 = double.Parse(Console.ReadLine());
40          Console.WriteLine("3. 立方體的表面積為：{0} ",
                        Area(length2, width2, height2));
41          Console.WriteLine(" -------------------------");
42          Console.Read();
43      }
44  }
45 }
```

4-11 Recursive

- The method calls itself in the method
- Advantage:
  ⇨ simplify program structure
- Disadvantage:
  if no ending statements, the recursion becomes infinite loop

Practice:factorial.sln

Try to write a program, the program get integers 1~10 from keyboard and calculate the factorial of inputted number

1 ! ⇨ 1

2! ⇨ 2x1! ⇨ 2x1 = 2

3! ⇨ 3x2! ⇨ 3x2x1! = 3 x2 x 1 = 6

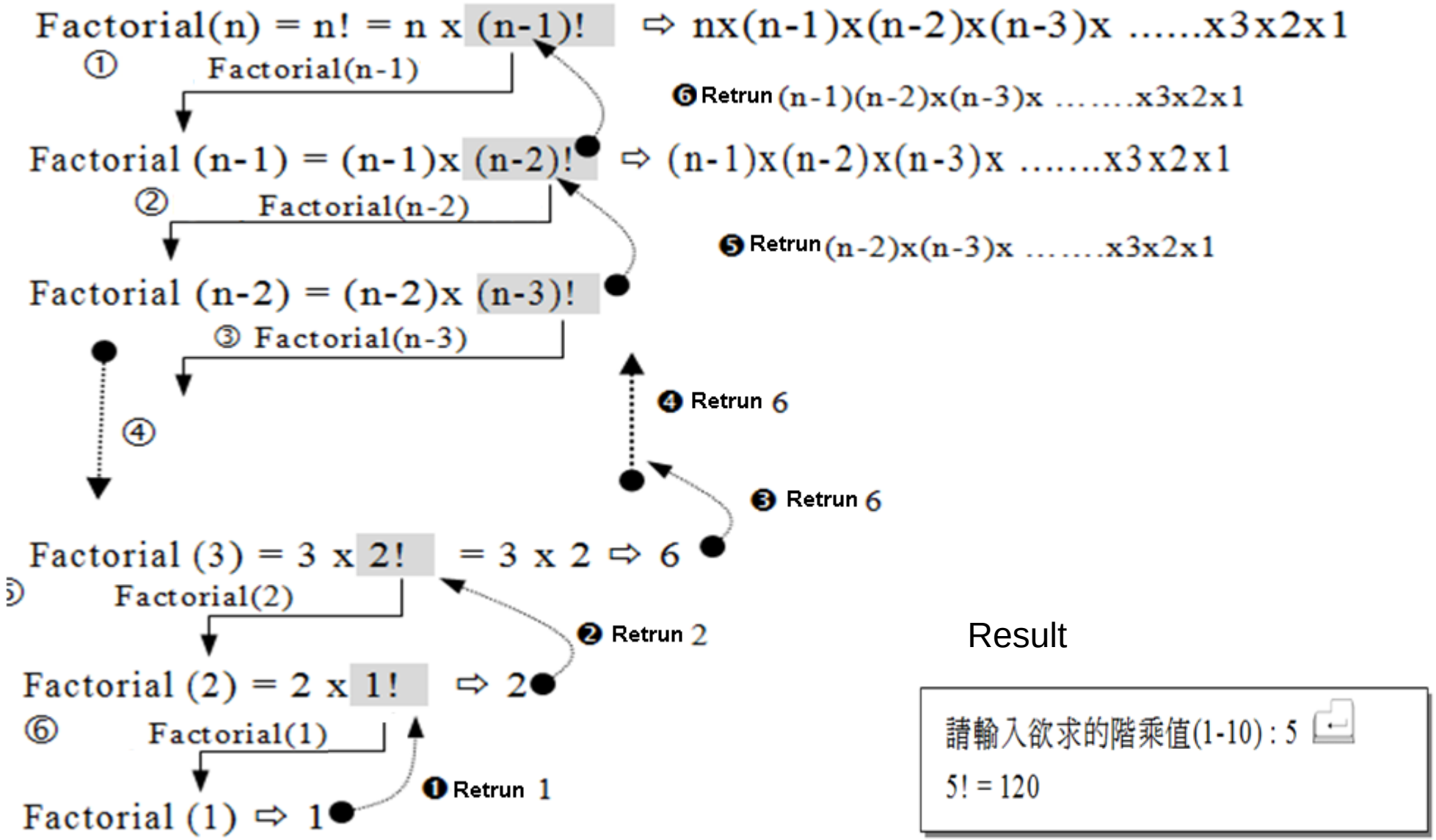
4! ⇨ 4x3! ⇨ 4x3x2! = 4x3x2x1! = 4x3x2x1= 24

⋮

(n-2)! ⇨ (n-2)x(n-3)! ⇨ (n-2)x(n-3)x(n-4)x …. x 3 x 2 x 1

(n-1)! ⇨ (n-1)x(n-2)! ⇨ (n-1)x(n-2)x(n-3)x …. x 3 x 2 x 1

n! ⇨ n x (n-1)! ⇨ nx(n-1)x(n-2)x(n-3)x …. x 3 x 2 x 1



Result

```
請輸入欲求的階乘值(1-10) : 5
5! = 120
```

FileName : factorial.sln

```
05      private static int Factorial(int tno)
06      {
07        if (tno == 1)
08          return 1;
09        else
10          return tno * Factorial(tno-1);
11      }
12      // -----------------------------------------------------
13      static void Main(string[] args)
14      {
15          int no ;
16          Console.Write("請輸入欲求的階乘值(1-10) : ");
17          no = int.Parse(Console.ReadLine());
18
19          Console.WriteLine ("{0}! = {1} " , no, Factorial(no));
20          Console.Read();
21       }
22   }
23 }
```

4-12 Life time of Variable

Block variables: variables declared in for/switch block

FileName : blockVar.sln

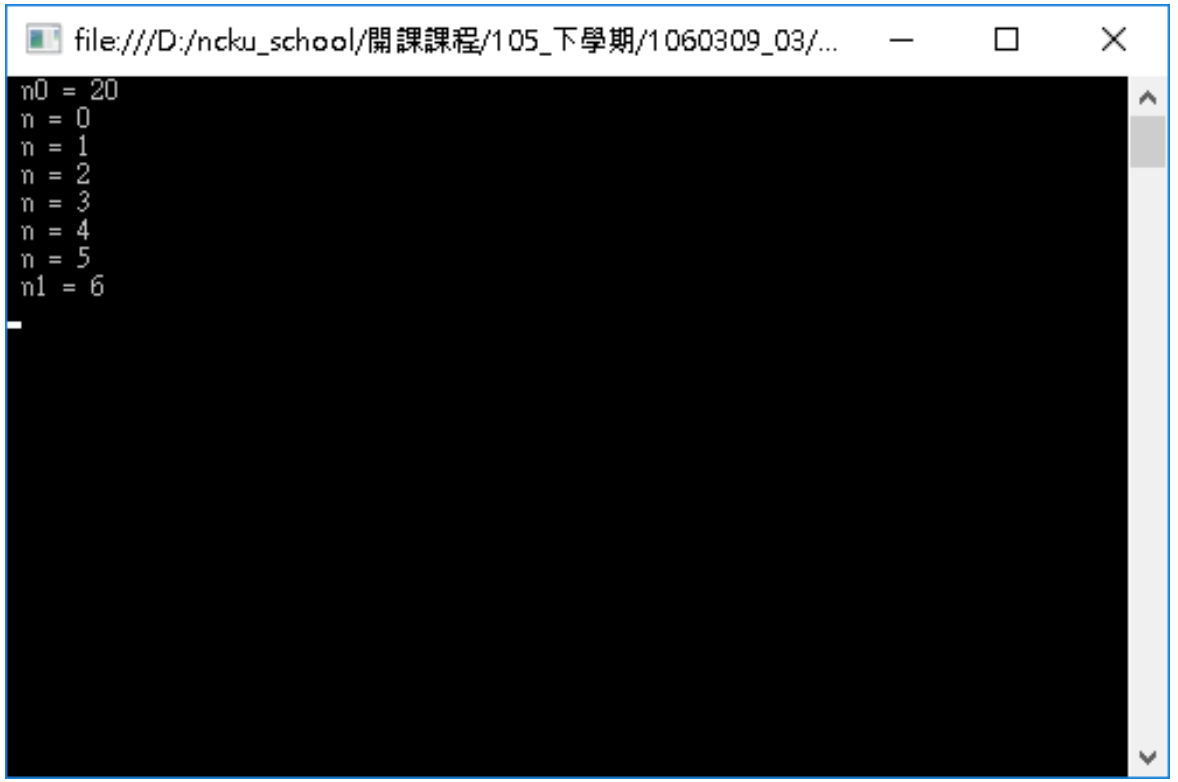
```
01 namespace ConsoleApplication1
02 {
03   class Program
04    {
05      static void Main(string[] args)
6    {
07         for (int n = 0 ; n < 6 ; n++)
08         {
09            Console.WriteLine (" n = {0} ", n);
10         }
11         Console.WriteLine(" n = {0} ", n);
12         Console.Read();
13      }
14   }
15 }
```

Local Variable

Variables declared in methods or functions

FileName : localVar.sln

```
05      static void Main(string[] args)
06      {
07          int n=20;
08          for (int n = 0; n < 6; n++)
09          {
10              Console.WriteLine(" n = {0} ", n);
11          }
12          Console.WriteLine(" n1 = {0} ", n);
13          Console.Read();
14      }
15  }
```



Member Variable

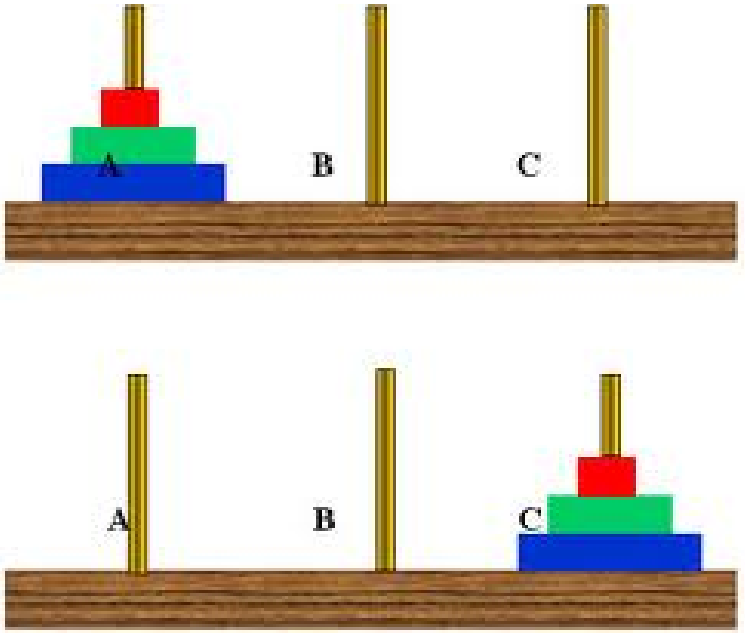
- Declared out of all methods or functions
- In general, these variables are in the top of class

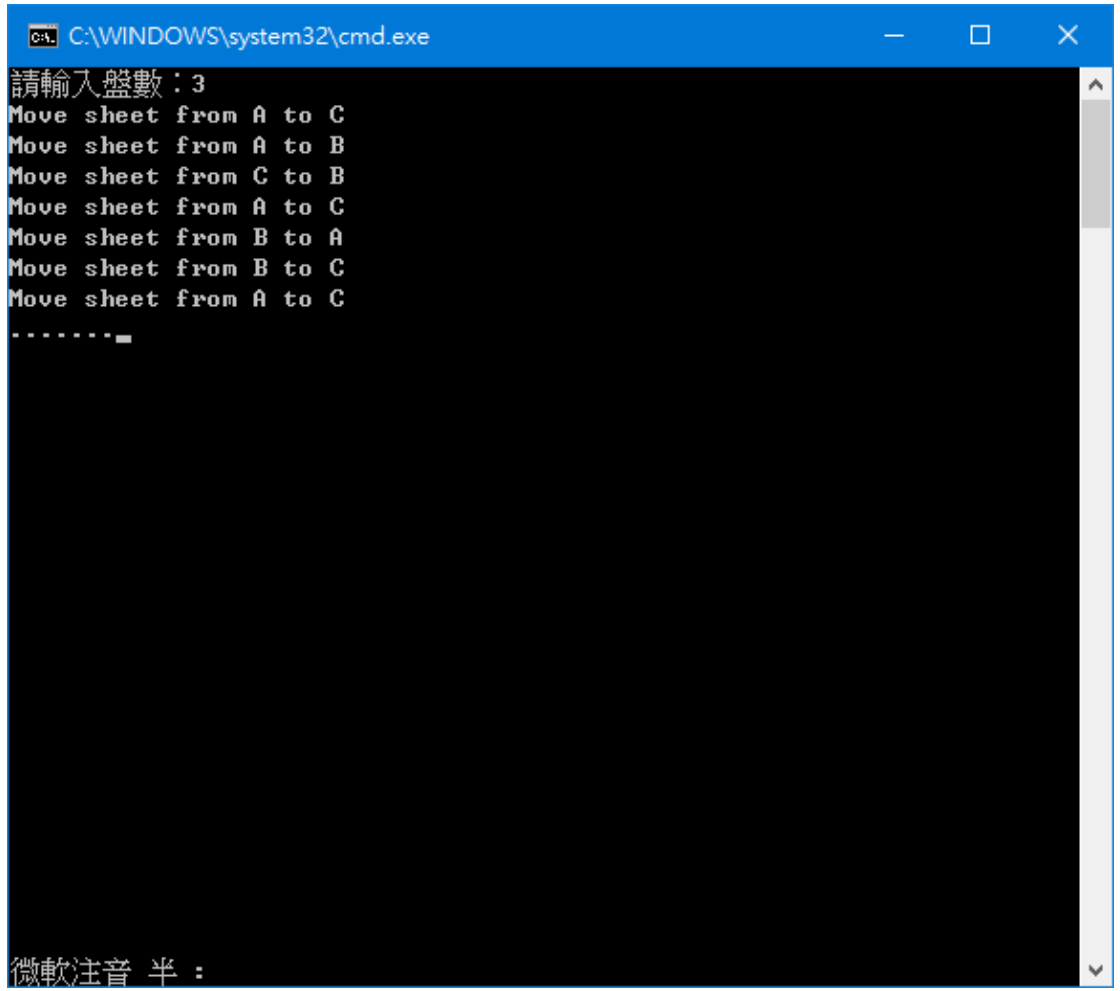
FileName : memeberVar.sln

```
05      private static int n=5;
06
07      private static void Add()
08      {
09        n += 5;    // n=15+5=20
10        Console.WriteLine(" n = {0} ", n);
11      }
13      static void Main(string[] args)
14      {
15          n+=10;
16          Console.WriteLine(" n = {0} ", n);
17          Add();
18          n += 20;
19          Console.WriteLine(" n = {0} ", n);
20          Console.Read();
21      }
```

Practice 2 – Tower of Hanoi by User-defined Method

- **Tips**
- Three sticks named as A, B, and C.
- Hollow circle plates can put placed on them.
- Start from putting N plates on the A stick.
- Plates will become smaller from the bottom to top.
- Only ONE plate can be moved each time.
- Big plates are not allowed to be on the small ones.
- Final goal: Remove all the plates to C stick.




C:\WINDOWS\system32\cmd.exe
請輸入盤數：3
Move sheet from A to C
Move sheet from A to B
Move sheet from C to B
Move sheet from A to C
Move sheet from B to A
Move sheet from B to C
Move sheet from A to C
....
微軟注音 半 ：

The End

Take a Break …