Array and Method(2)

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)	Index key (myKeys)
mary	90
jack	60
tom	50
david	70
grace	80

```
Before sort:
Subscript Relative value
                            Index key
                90
       mary:
                60
       jack :
       tom :
k=3
       david:
       grace: 80
k=4
After sort:
Subscript Relative value
                            Index key
k=0
                 50
       tom :
k=1
       jack :
                 60
                 70
k=2
       david:
       grace :
k=4
       mary:
```

```
FileName: arySort.sln
05
      static void Main(string[] args)
06
07
       int[] myKeys = new int[] { 90, 60, 50, 70, 80 };
       string[] myValues= new string[] { "mary", "jack", "tom", "david", "grace" };
80
10
       Console.WriteLine(" Before sort: ");
11
       Console.WriteLine(" Subscript Relative value Index key ");
12
       for (int k = 0; k \le myKeys.Length-1; k++)
13
           Console.Write("k=\{0\} \setminus \{1\} : \{2\} \setminus [n], k, myValues[k], myKeys[k]);
14
       Console.WriteLine();
16
       Array.Sort(myKeys, myValues);
17
       Console.WriteLine(" After sort: ");
18
       Console.WriteLine("Subscript Relative value Index key");
19
       for (int k = 0; k \le myKeys.Length - 1; k++)
         Console.Write(" k=\{0\} \setminus \{1\} : \{2\} \setminus [n], k, myValues[k], myKeys[k]);
20
21
       Console.Read();
22
```

```
Before sort:
                             Index key
Subscript Relative value
k=0
                 90
k=1
       jack :
                 60
k=2
                 50
       tom :
k=3
                70
       david:
k=4
       grace :
After sort:
Subscript Relative value
                             Index key
k=0
       jack :
                  60
k=1
k=2
                  70
       david:
k=3
                  80
       grace:
k=4
       mary:
```

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.

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

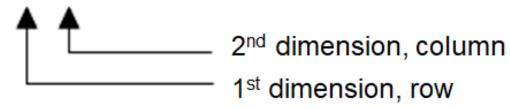
- 2 subscripts 2-D array
- 3 subscripts 3-D array
- Application of 2-D array can be seen in matrix, score table, sales achievement, stock quotation table, etc.

Declaration of 2-D Array

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

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

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:

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

$$0^{th}$$
 row 1^{st} row 2^{nd} row

Example (readAry2):

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	Student Name	Chinese	English	Math
0	1	AA	80	75	60
1	2	BB	65	67	62
2	3	CC	100	93	91
3	4	DD	98	25	50
4	5	EE	83	82	87

Result:

```
Input seat number (1-5): 3

1. Student name: CC

2. Chinese: 100

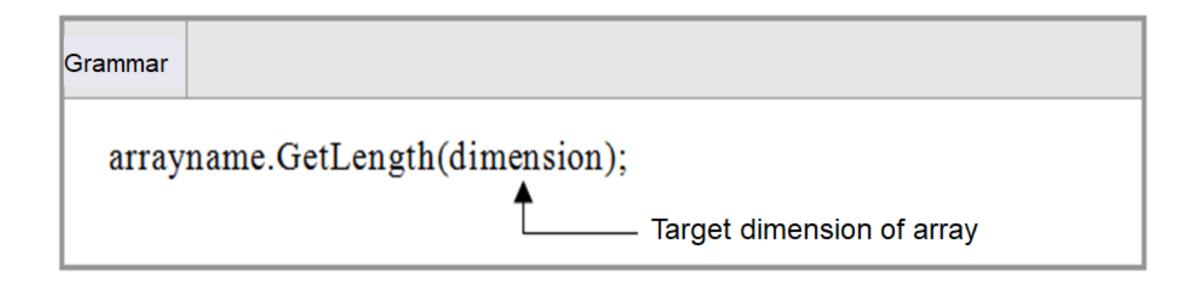
3. English: 93

4. Mathematics: 91
```

```
FileName : readAry2.sln
05
      static void Main(string[] args)
06
         string[] name = {"AA","BB","CC","DD","EE"};
07
         int[,] score = {\{80,75,60\},\{65,67,62\},\{100,93,91\},\{98,25,50\},\{83,82,87\}\}};
08
09
         int no;
10
         Console.Write(" Input seat number (1-5): ");
11
         no = int.Parse(Console.ReadLine());
         if (no >= 1 &  no <= 5)
12
13
14
            Console.WriteLine("1. Student name : {0} ", name[no-1]);
            Console.WriteLine("2. Chinese: {0} ", score[no-1,0]);
15
16
            Console.WriteLine("3. English: {0} ", score[no-1,1]);
                                                                                                               2
17
            Console.WriteLine("4. Mathematics: {0} ", score[no-1,2
                                                                            S/N
                                                                                 Student Name
                                                                                            Chinese
                                                                                                    English
                                                                                                             Math
18
                                                                                             80
                                                                                                      75
                                                                                                              60
                                                                            1
                                                                                    AA
19
         else
            Console.WriteLine(" ... This is an empty number ... ");
                                                                            2
                                                                                    BB
                                                                                             65
                                                                                                      67
                                                                                                              62
20
21
         Console.Read();
                                                                            3
                                                                                             100
                                                                                                      93
                                                                                                              91
                                                                                    CC
22
                                                                                    DD
                                                                                                      25
                                                                            4
                                                                                             98
                                                                                                              50
                                                                                                              87
                                                                            5
                                                                                             83
                                                                                                      82
                                                                                    EE
```

GetLength() Method

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



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

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

```
int[,,] score = new int[3, 15, 20]; // Used in 3-D array
Ex1:
       int index;
       index = score.GetUpperBound(0);
                                             Return the 1st upper bound: 2
       index = score.GetUpperBound(1);
                                             Return the 2<sup>nd</sup> upper bound: 14
                                             Return the 3<sup>rd</sup> upper bound: 19
       index = score.GetUpperBound(2);
       int[] score = new int[6];
                                             Used in 1-D array
Ex2:
       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
- AdvantageFlexible
- Disadvantage
 Out of memory possibility during running
 ⇒ Cause crash if no special handler and the program cannot continue

Example (dynamicAry):

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

Result:

輸入陣列大小:5 111 222 333 444 555 輸入陣列大小: 12 中 " 陣列大小範圍限 1-10!..."

FileName: dynamicAry.sln

```
輸入陣列大小:5 🖃
05
      static void Main(string[] args)
                                              111
06
                                              222
07
         int[] data;
                                              333
80
         string str1 = "";
                                              444
09
         int size;
                                              555
         Console.Write(" 輸入陣列大小:");
10
11
         size = int.Parse(Console.ReadLine());
         if (size >= 1 && size <= 9)
12
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)
                str1 += num.ToString() + "\n";
18
19
             Console.WriteLine("{0}", str1);
20
21
         else
22
           Console.WriteLine(" 陣列大小範圍限1-10!...");
23
         Console.Read();
24
```

輸入陣列大小:12 <u>-</u> " 陣列大小範圍限 1-10!..."

4-6 Method

- Use method to organize the source code advantage:
- 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	Ex1: num = rnd.Next(); Result: num = 556146380
Grammar: rnd.Next()	
Function: return an unsigned integer	Ex2: $n = rnd.Next(49)$;
random between 0 and 2147483647	Result: n = 43 // generate 0 ~ 48 random
Grammar: rnd.Next(n)	
Function: return an unsigned integer	Ex3: $n = rnd.Next(-5, 6)$
random smaller than n	Result: n = -3 // generate -5 ~ 5 random
Grammar: rnd.Next(n1, n2)	
Function: return an integer random n1≤n <n2< td=""><td></td></n2<>	

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):

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

Result:

第1個亂數: 40

第 2 個亂數: 46

第 3 個亂數: 10

第 4 個亂數: 19

第 5 個亂數: 10

```
// FileName : ranObj.sIn
05 static void Main(string[] args)
06 {
       Random ranObj = new Random();
07
80
      int ranNo;
      for (int i=1; i <= 5; i++)
09
10
11
          ranNo = ranObj.Next(10, 51);
          Console.WriteLine("第{0}個亂數:{1}",i,ranNo);
12
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;]
}
```

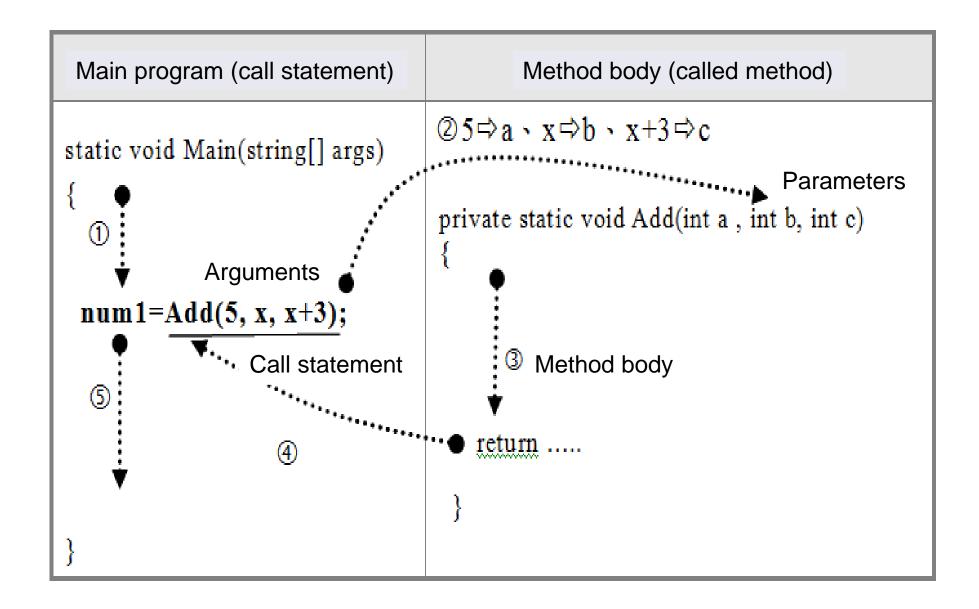
Example

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.

How to Call Method in the Same Class

Grammar

- [objectName.]methodName([arguments]);
- variable = [objectName.]methodName([arguments]);



Example (privateStatic):

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:

- 請輸入一個正整數: 25
 相加結果: 45

```
// FileName : privateStatic.sln
   private static int Add(int n1)
06
                       // local variable
         int sum;
07
                                           1. 請輸入一個正整數:25 🗀
80
         sum = n1 + 20;
                                           2. 相加結果:45
                         // return value
09
        return sum;
10
11
   static void Main(string[] args)
13
        Console.Write("1. 請輸入一個正整數:");
14
15
        int num1 = int.Parse(Console.ReadLine());
        int num2 = Add( num1);
16
        Console.WriteLine("2. 相加結果: {0} ", num2);
17
        Console.Read();
18
19
```

How to Call Method in Different Class

Grammar

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

Example (classAdd):

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

Result:

- 請輸入一個正整數: 25
 相加結果: 45

```
// FileName :classAdd.sln
01 namespace classAdd
02 {
                                                           1. 請輸入一個正整數:25 🗀
                      // Program類別
03
     class Program
04
                                                           2. 相加結果: 45
05
        static void Main(string[] args)
06
          Console.Write("1. 請輸入一個正整數:");
07
          int num1 = int.Parse(Console.ReadLine());
08
          int num2 = Class1.Add( num1); //有建立 static 所以可以直接哪個 className.methodName()
09
10
          Console.WriteLine("2. 相加結果: {0}", num2);
          Console.Read();
11
12
13
14
     public class Class1
                         //建立Class1類別
15
        public static int Add(int n1) // Class1類別內的Add方法
16
17
18
           int sum;
19
           sum = n1 + 20;
                         // 傳回值
20
           return sum;
21
22
23
```

Example (objAddMethod):

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

Result:

請輸入一個正整數:25
 相加結果:45

```
// FileName :objAddMethod.sln
01 namespace objAddMethod
                                          1. 請輸入一個正整數:25 🗀
02 {
   class Program
03
                                           2. 相加結果:45
04
05
      static void Main(string[] args)
06
07
       Class1 obj = new Class1(); // 建立Class1類別的物件
80
       Console.Write("1. 請輸入一個正整數:");
       int num1 = int.Parse(Console.ReadLine());
09
       int num2 = obj.Add( num1); // 呼叫obj物件的Add方法
10
       Console.WriteLine("2. 相加結果: {0} ", num2);
11
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
                      #傳回值
         return sum;
23
24
25 }
```

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

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

Example (passValue):

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.sIn
                                                   ===== 傳值呼叫 ======
   private static void passValue(int x, int y)
                                                  1. 呼叫前:a = 1 b = 10
06
                                                 2. 呼叫中: x = 6 y = 20
07
        x = x + 5; // x = 1 + 6 = 6
                                                  3. 呼叫後:a=1 b=10
        y = y + 10; // y = 10+10 = 20
80
        Console.WriteLine (" 2. 呼叫中: x = {0} y = {1}", x, y );
09
10 }
12 static void Main(string[] args)
13 {
14
         int a, b;
15
         a = 1:
16
         b = 10;
         Console.WriteLine(" ==== 傳值呼叫 === ");
17
         Console.WriteLine(" 1. 呼叫前: a = {0} b = {1} ", a, b);
18
         passValue(a,b);
19
         Console.WriteLine(" 3. 呼叫後: a = {0} b = {1} ", a, b);
20
         Console.Read();
21
22 }
```

Call by Reference

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

Argument	Memory addr	Memory value	Parameter
a	100000		X
b	100004		у
	100008		
	10000c		

```
Define method main body:
                         ref = * in C
    private void Add (ref int x, ref int y)
                                         Called method
                 Refer
                         Refer
Main program Main() method
          Add (ref a, ref b)
                              ref = & in C
```

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);
```

Example (sum):

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:

請輸入被加數: 4
 請輸入加數: 5
 利加額: 4 + 5 = 9

FileName: sum.sln

```
05
       private static void passRef(float a, float b, ref float c)
06
         c = a + b; /// c = 4 + 5 = 9
07
                                                         1. 請輸入被加數: 4 🗀
80
                                                         2. 請輸入 加 數:5 🗀
09
10
       static void Main(string[] args)
                                                         3. 相加結果: 4+5=9
11
12
          float sum=0;
13
          Console.Write("1. 請輸入被加數:");
          float num1 = float.Parse(Console.ReadLine());
14
          Console.Write("2. 請輸入加數:");
16
          float num2 = float.Parse(Console.ReadLine());
17
          passRef(num1, num2, ref sum);
18
          Console.WriteLine("3. 相加結果: {0} + {1} = {2} ", num1,
19
                             num2 ,sum );
          Console.Read();
20
21
```

4-9 Passing Array as Parameter

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

```
Main()
int[] Ary = new int[] \{10,20,30,40\}; \Leftrightarrow Create array Ary
 passAry (ref Ary);
                                Call passAry() method
                  Refer
Method:
```

Example (maxArg):

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:

請輸入第 1 數: 10 請輸入第 2 數: 20 請輸入第 3 數: 30 請輸入第 4 數: 40 請輸入第 5 數: 50 最大數為: 50 平均值為: 30

FileName: maxAvg.sln

```
05
      private static float FMax(ref float[] tAry)
06
        float tMax = tAry[0]; // 先假設陣列元素0 為最大值
07
        for (int i = 0; i < tAry.Length; i++)
80
           if (tAry[i] > tMax)
09
                                                   請輸入第1數:10 🖃
             tMax = tAry[i];
10
                                                    請輸入第2數:20 🗀
11
        return (tMax);
                                                    請輸入第3數:30 🔄
12
13
                                                    請輸入第4數:40 🖃
14
      private static float FAvg(ref float[] tAry)
                                                    請輸入第5數:50 🗀
15
                                                    最大數為:50
         float tAvg = new float();
16
                                                    平均值為:30
17
         foreach (float n in tAry)
18
            tAvg += n;
         return tAvg/tAry.Length;
19
20
21
```

```
請輸入第2數:20 🗀
      static void Main(string[] args)
22
                                                     請輸入第3數:30 🖃
23
                                                     請輸入第4數:40 🖃
        float[] numAry = new float[5];
24
                                                     請輸入第5數:50 🖃
25
        for (int i = 0; i < 5; i++)
                                                     最大數為:50
26
                                                     平均值為:30
27
          Console.Write("請輸入第 {0} 數:", i + 1);
28
          numAry[i] = float.Parse(Console.ReadLine());
29
30
31
        Console.WriteLine("最大數為 : {0} ", FMax(ref numAry));
        Console.WriteLine("平均值為 : {0} ", FAvg(ref numAry));
32
33
        Console.Read();
34
35
36
37 }
```

請輸入第1數:10 🗀

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)

Example (overload1):

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
- 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
                                                         1. 請輸入半徑:5 🗀
01 namespace overload1
                                                         1. 圓形的面積為: 78.5
02 {
                                                         2. 請輸入矩形的長度: 10 🗀
03 class Program
                                                         2. 請輸入矩形的寬度: 3
04
                                                         2. 矩形的面積為: 30
05
     private static double Area(double r)
06
                                                         3. 請輸入立方體的長度: 10 🗀
        return r*r*3.14;
07
                                                         3. 請輸入立方體的寬度: 3 🖳
08
                                                         3. 請輸入立方體的高度: 2 🔄
09
                                                         3. 立方體的表面積為: 112
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
       return 2 * (x * y + y * z + z * x);
17
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));
      Console.WriteLine(" -----");
25
26
      // -----
27
      Console.Write("2. 請輸入矩形的長度:");
28
      double length1 = double.Parse(Console.ReadLine());
      Console.Write("2. 請輸入矩形的寬度:");
29
30
      double width1 = double.Parse(Console.ReadLine());
      Console.WriteLine ("3. 矩形的面積為: {0}", Area(length1,
31
       width1));
      Console.WriteLine(" -----");
32
```

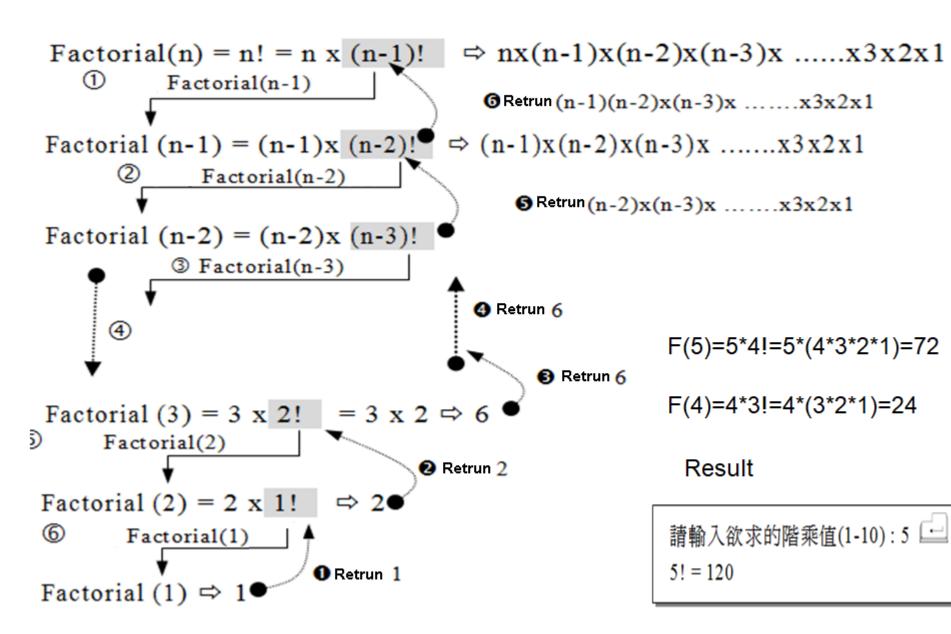
```
34
       Console.Write("3. 請輸入立方體的長度:");
35
       double length2 = double.Parse(Console.ReadLine());
36
       Console.Write("3. 請輸入立方體的寬度:");
37
       double width2 = double.Parse(Console.ReadLine());
       Console.Write("3. 請輸入立方體的高度:");
38
       double height2 = double.Parse(Console.ReadLine());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

Example (factoral):

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



```
FileName: factorial.sln
     private static int Factorial(int tno)
05
06
       if (tno == 1)
07
         return 1;
80
09
       else
         return tno * Factorial(tno-1);
10
11
12
     static void Main(string[] args)
13
14
15
        int no;
        Console.Write("請輸入欲求的階乘值(1-10):");
16
17
        no = int.Parse(Console.ReadLine());
18
19
         Console.WriteLine ("\{0\}! = \{1\}", no, Factorial(no));
         Console.Read();
20
21
22
23 }
```

4-12 Life time of Variable

Block variables: variables declared in for/switch block

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

```
C# Block variables
                                                                     🕶 🐾 Block_va
            using System.Collections.Generic;
           using System.Linq;
           using System.Text;
           using System. Threading. Tasks;
          ⊟namespace Block_variables
               0 個參考
               class Program
                   0 個參考
    11
                   static void Main(string[] args)
    12
    13
    14
                                for (int n = 0; n < 6; n++)
    15
    16
                                        Console. WriteLine(" n = \{0\} ", n);
    17
    18
                                Console. WriteLine(" n = \{0\} ", n);
    19
                                Console.Read();
    20
   21
    22
    23
    24
    25
錯誤清單
                   ▼ | ② 1 錯誤 | ▲ 0 警告 | ● 0 訊息
整個方案
                                                           組建 + IntelliSense
       程式碼 說明

☆ CS0103 名稱 'n' 不存在於目前的內容中
```

Local Variable

Variables declared in methods or functions

```
FileName: localVar.sln
      static void Main(string[] args)
05
06
        int n=20;
07
        for (int n = 0; n < 6; n++)
80
09
10
           Console.WriteLine(" n = \{0\} ", n);
11
        Console.WriteLine(" n1 = {0} ", n);
12
        Console.Read();
13
14
15 }
```

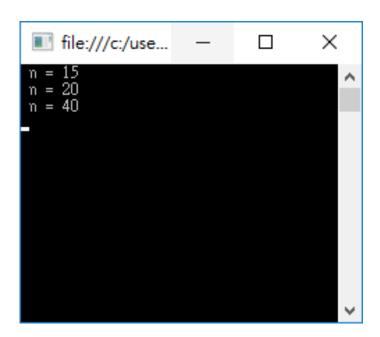
```
file:///D:/ncku_schoo...
                                          X
n = 1
```

Member Variable

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

FileName: memeberVar.sln

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



How to: Parse Strings Using String.Split

string phrase = "The quick brown fox jumps over the lazy dog."; string[] words = phrase.Split(' '); // string.split foreach (var word in words)

{ System.Console.WriteLine(\$"<{word}>"); }

```
string phrase = "The quick brown fox jumps over the lazy dog.";
string[] words = phrase.Split(' ');
foreach (var word in words)
{
    System.Console.WriteLine($"<{word}>");
}
```

result

```
<The>
<quick>
<brown>
<fox>
<jumps>
<over>
<the>
<lazy>
<dog.>
```

result

```
<The>
<quick>
<br/>brown>
\langle \rangle
\langle \rangle
\langle \rangle
<fox>
\langle \rangle
\langle \rangle
\langle \rangle
<>
<jumps>
<the>
<lazv>
<dog.>
```

How to: Parse Strings Using String.Split

```
char[] delimiterChars = { ' ', ', ', '.', ':', '\t' };
string text = "one\ttwo three:four,five six seven";
System.Console.WriteLine($"Original text: '{text}'");
string[] words = text.Split(delimiterChars);
System.Console.WriteLine($"{words.Length} words in text:");
foreach (var word in words)
{
    System.Console.WriteLine($"<{word}>");
}
```

result

```
Original text: 'one two three:four,five six seven'
7 words in text:
<one>
<two>
<three>
<four>
<five>
<six>
<seven>
```

The End

Take a Break.....