Topic 7A

Methods with parameters and return value

Topics

Objectives:

- Review of Methods
- Know how to pass data between methods using parameter passing
- Know how to return values from a method
- Know how to make use of return values from methods

Why Methods?

- To break down a complex problem to smaller simpler sub problems of manageable size.
- We then develop algorithms/solution for each of these sub problems. Each algorithm is implemented as a module. In C#, modules are known as **methods**.

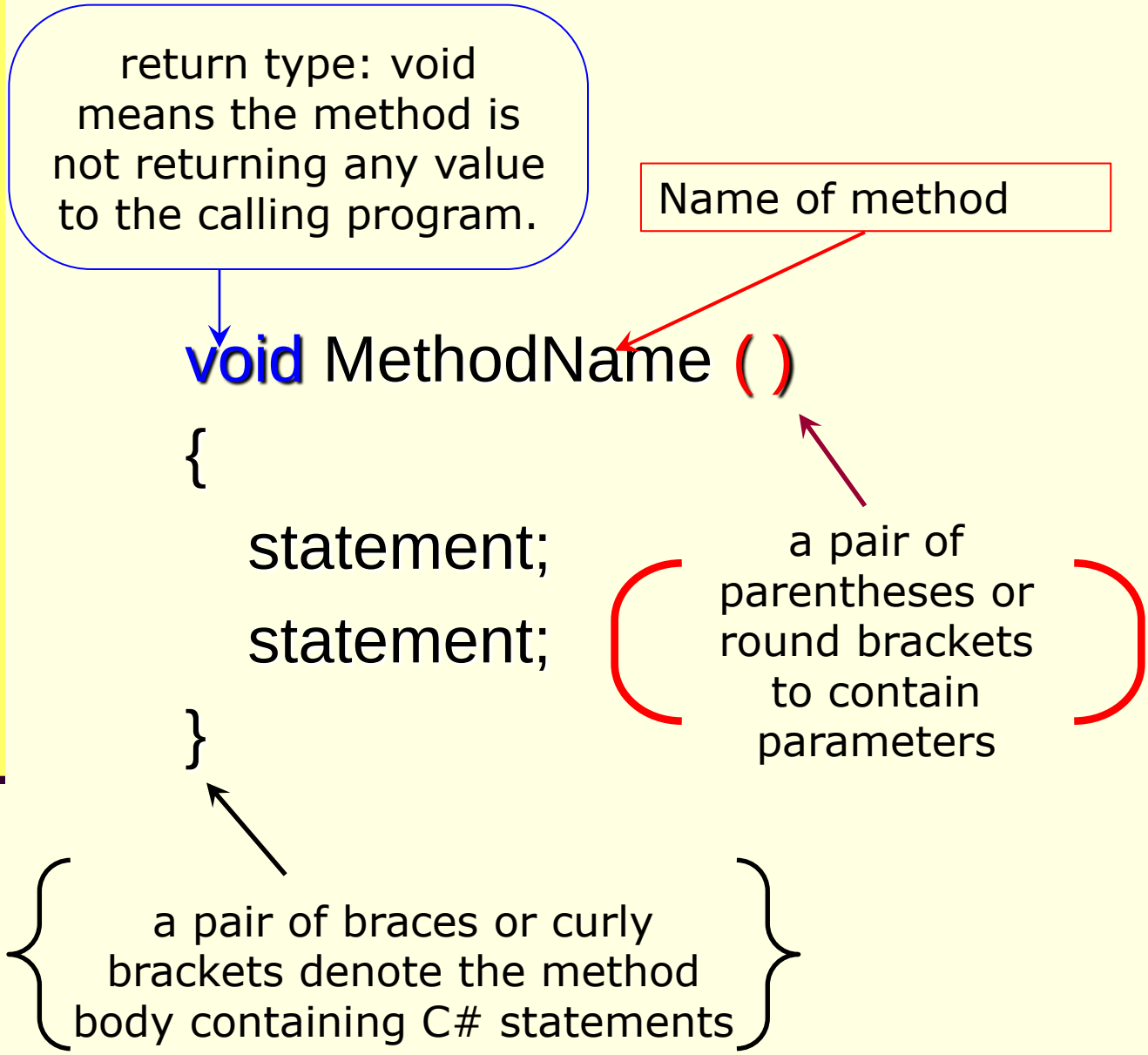
Review : Passing data between methods

- Methods can pass data to each other as follows:
  1) Through Class Variables
  2) By using Parameters
  3) By returning a value from a method.
- We have covered (1) (reminder: class variables are defined outside of any methods e.g
  - int count;
- (2) and (3) will be introduced in this topic
- Details of (2) and (3) will be covered in the next semester

Review : Passing data between methods

- Use of class variables for passing data between methods has **disadvantages**:
  - ✓ It is not clear what are the inputs and outputs of the method.
  - ✓ Variables are not protected against changes by other methods
  - ✓ A method using class variables loses its reusability.

Structure of a **Method with NO Parameter**



Structure of a **Method with Parameter**

Methods can pass data to each other as follows:

1)Through Class Variables

2)By using Parameters

3)By returning a value from a method.

Structure of a **Method with Parameter**

return type: void means the method is not returning any value to the calling program.

Name of method

```
void MethodName ( parameter )
{
    statement;
    statement;
}
```

a pair of parentheses or round brackets to contain parameters

a pair of braces or curly brackets denote the method body containing C# statements

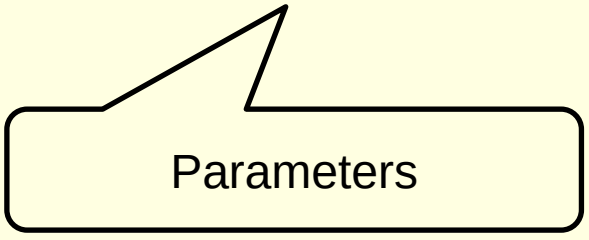
Method Definition with parameters

- Syntax :

```
private void MethodName(parameters)
  {
     //statements;
     …
   }
```

- Parameters are places that hold variable values from the calling program.
- Parameters must be declared (just like variables) eg.

```
private void GetPay( double hoursWorked,
double hourlyRate )
{
   // statements
}
```



Method Execution with arguments

- When the method is executed or called, the values passed into the method are known as arguments:

```
private void MethodA()
  {…
    MethodBName(arguments);
    …
  }
```

- *These arguments must be of the same data type as the parameters defined in the method.*

GetPay(**hourClocked, 10.5f**)

Arguments

- When passing argument, only values are **copied** into the method.

Arguments & Parameters

```
Private void MethodA( )
{
    int age = 17;   char race = 'O';
    float salary = 500f;
    string name = "Joe Sim";


Display( age, race, salary, name );

    }

Private void Display
        ( int X, char Y, float Z, string W )
{
    Console.WriteLine("Name is " +  W      );
    Console.WriteLine("Age is " +    X     );
    Console.WriteLine("Pay is $" +   Z     );
    Console.WriteLine("Race is " +   Y     );
}
```

Argument age, race … must be of the same type as the parameter

age must be **int** for the first parameter is **int**

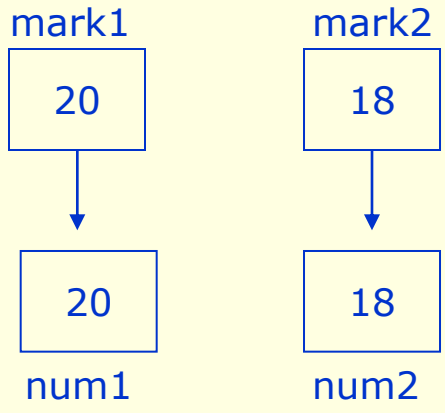
Content in
age is copied to X,
race is copied to Y,
salary is copied to Z and
name is copied to Y

Example #1 Method with parameter - FindMax

```
Private void MethodA()
{
    int mark1, mark2;

  mark1=int.Parse(
1   txtmark1.Text));
  mark2=int.Parse(
        txtmark2.Text);

2 FindMax(mark1,mark2);
}

                    3
Private void FindMax (
      int num1, int num2)
{
  int max = num1;
  if ( num2 > max)          4
       max = num2;
  MessageBox.Show
   (“maximum=” + max);
}
```

(1) User enters 2 marks. Data is stored into mark1, mark2

(2) MethodA calls FindMax() and passes in mark1, mark2



(3) A copy of values of mark1 and mark2 is passed to num1 and num2

(4) FindMax() receives values; calculates & displays the maximum

Class variables vs Parameters

```
static int mark1, mark2;

private  void MethodA()
{
  mark1=int.Parse(txtmark1.Text);
  mark2=int.Parse(txtmark2.Text);

   FindMax();
}

private void FindMax ()
{
   int max = mark1;
   if ( mark2 > max)
        max = mark2;
  MessageBox.Show("maximum=" + max);
}
```

```
private void MethodA()
{
   int mark1, mark2;
   mark1=int.Parse(txtmark1.Text);
   mark2=int.Parse(txtmark2.Tex);

   FindMax(mark1,mark2);
}

private void FindMax (int num1, int num2)
{
   int max = num1;
   if ( num2 > max)
        max = num2;
  MessageBox.Show("maximum=" + max);
}
```

How many differences can you find?

Structure of a **Method with Parameter AND return value**

Methods can pass data to each other as follows:

1)Through Class Variables

2)By using Parameters

3)By returning a value from a method.

Structure of a **Method with Parameter AND return value**

return type: int means the method is return a int value to the calling program. The return type can be any data type e.g float

Name of method

```
int MethodName ( parameter )
{
    statement;
    statement;
    return xxx;
}
```

a pair of parentheses or round brackets to contain parameters

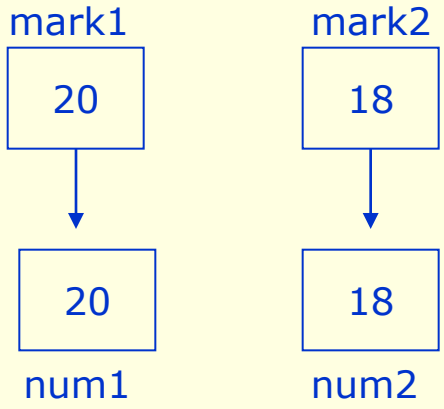
a pair of braces or curly brackets denote the method body containing C# statements

Example of a **Method with Parameter AND return value**

```
private void MethodA()
{
    int mark1, mark2;

1   mark1=int.Parse(txtmark1
.Text);
  mark2=int.Parse(
Txtmark2.Text));

                          2
5  int maxInt = FindMax(
            mark1,mark2);

MessageBox.Show
  ("maximum=" + maxInt);
}

                          3
private int FindMax (
      int num1, int num2)
{
  int max = num1;
  if ( num2 > max)
      max = num2;
  return max;  4
}
```

(1) User enters 2 marks. Data is stored into mark1, mark2

(2) Main calls FindMax() and passes in mark1, mark2



(3) A copy of values of mark1 and mark2 is passed to num1 and num2

(4) FindMax() receives values; calculates & **returns** the maximum number to Main()

(5) System **assigns** return value into maxInt

No Return vs Return

```
private  void MethodA()
{
  int mark1, mark2;
  mark1=int.Parse(txtmark1.Text);
  mark2=int.Parse(txtmark2.Text);

  FindMax(mark1,mark2);
}


private void FindMax (int num1, int num2)
{
   int max = num1;
   if ( num2 > max)
        max = num2;
  MessageBox.Show("maximum=" + max);

}
```

```
private void MethodA()
{
   int mark1, mark2, max;
   mark1=int.Parse(txtmark1.Text);
   mark2=int.Parse(txtmark2.Tex);

   max = FindMax(mark1,mark2);
MessageBox.Show("maximum=" + max);
}

private int FindMax (int num1, int num2)
{
     int temp = num1;
      if ( num2 > temp)
           temp = num2;
      return temp;
}
```

How many differences can you find?

Summary

- Introduce methods with parameter and method with return value
- Knowing that we could share data among methods without using class variable

How to apply it in the project?

Question 1

A method is defined as:

```
private int CalcArea( )
{
    int area;
    // codes to calculate area……
    return area;
}
```

How should you call the above method?

Question 2

Spot the error below:

```
private void CalcGST( )
{
    float gst = 0f;
   // codes to calculate gst…… ;

    return gst;
}
```

Question 3

A method is defined as:

```
private void CalcArea( int X, int Y )
{
    int area;
    area = X * Y;
    lblArea.Text = area.ToString();
}
```

How should you call the above method where X = 3 and Y = 10 ?

Question 4

The code below calls a method PrintBill:

PrintBill("Adam" , 300.00, 7.0);

Write the **Header** for the method PrintBill.

Question 5

Spot the error below:

```
private void  Display( string M, int X, float Y )
{
    lblAnswer.Text = M + X.ToString() +
        Y.ToString();
}
```

//Call the method above

```
Display( 55, "John", 2.2f );

Display( "Albert", 5f);

Display("Pam", 33, 67.8f, 800);
```

End of Topic 7A

Methods with parameters and return value