Topic 4B

Methods with class variable

Topics

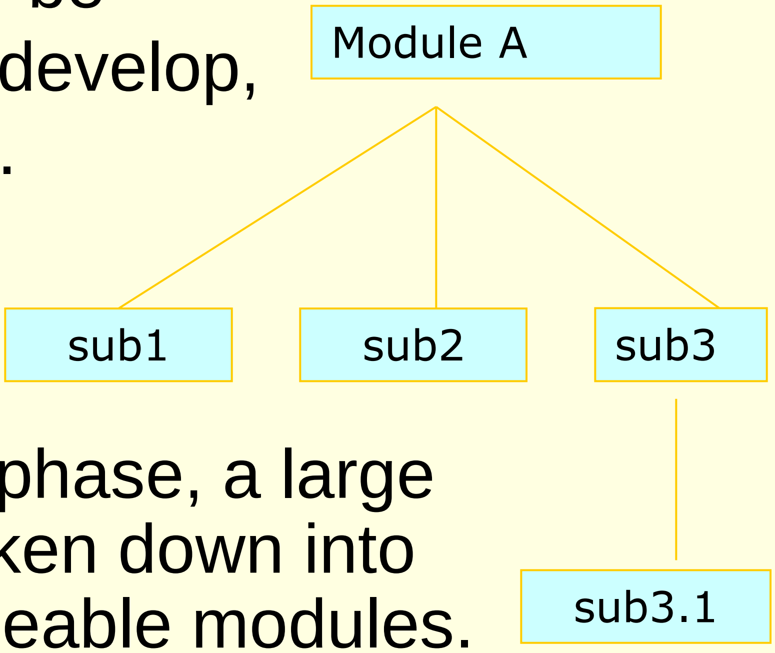
- Methods
- Local & Class Variables
- Methods using Class variables

Objectives:

- Know how to implement methods
- Understand the differences between local and class variables
- Ablle to write methods using class variables

Review on Modular Program Design

- C# codes written in one large single program would virtually be impossible to develop, debug, or test.



- During design phase, a large problem is broken down into smaller manageable modules.
- These modules are implemented as **methods** in C# code.

In C#, all methods belong to a class.

Why use Methods?

- Good for program organization
  - Partitions large programs into smaller, manageable units
  - Simplifies programming & debugging
- Avoiding repetition of code
  - Avoids writing the same code over and over again.
- Independence
  - Methods developed in one program may be incorporated into others
  - eg. CalculateAverage()

Avoid Repeating Codes

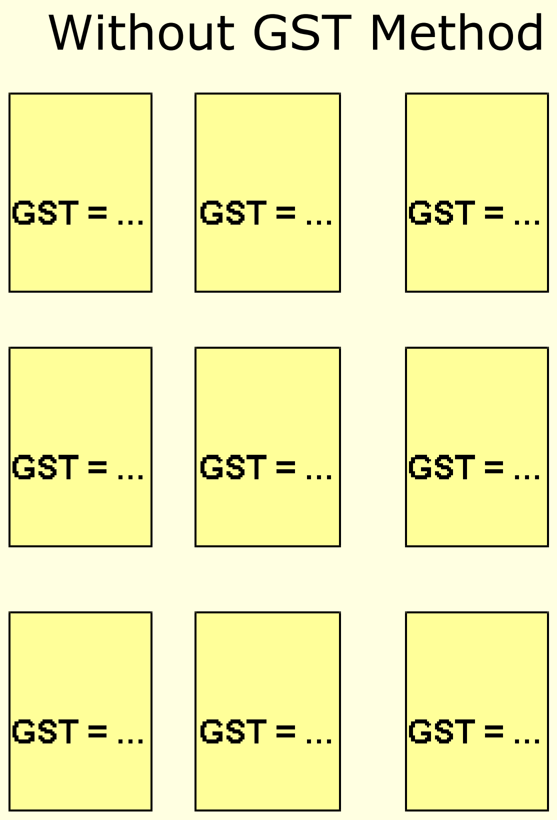


Fig A

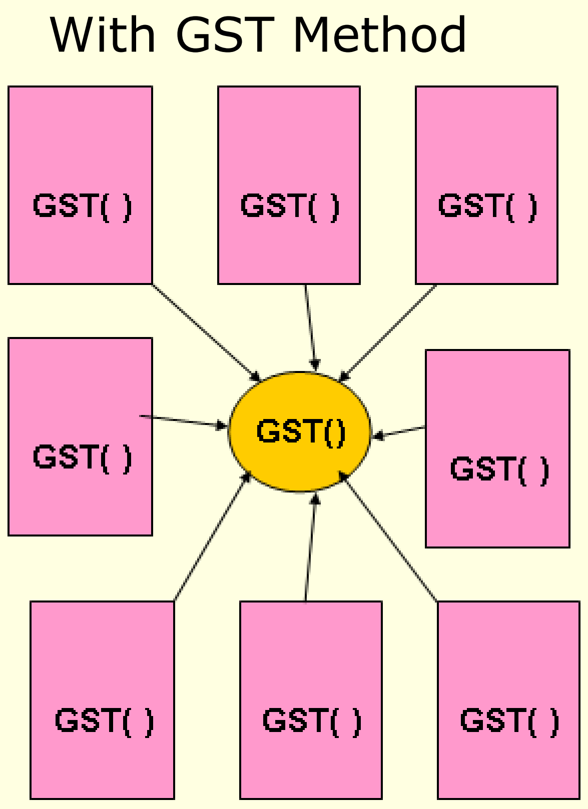


Fig B

In Figure A, the GST is calculated in each of the methods (yellow boxes). If the GST amount is changed (eg. 5% to 7%), the codes in all the yellow boxes need to change.

In Figure B, the GST is calculated by a single Method and all the other methods (purple boxes) call this single Method. If the GST changes, you only need to modify one time (ie. orange oval).

Passing data between methods

- There are 4 ways through which methods can pass data to each other:
  1) Through Class Variables
  2) Through Instance Variables
  3) By using Parameters
  4) By returning a value after a method is executed.

- We will cover method (1) in this session
- (2), (3) and (4) covered in Term 2.
- Instance variables will be covered next semester when we talk about objects.

What are Local variables?

- Recall that variables must be declared before we can use them. Example:

CalculateGrade is a method

```
private void CalculateGrade ()
 {
    int age; // local variable
    …
 }
```

- Variables declared **inside** a method are known as **LOCAL** variables.
  - Eg. age
  - It can only be accessible in the method CalculateGrade()

Recap about Local Variables

- **Local variables**
  - Are variables declared in a method body { }
  - They are used or seen only in the method they are defined (*local*)
  - They are created when the method is executed and destroyed when the method ends.

```
    private void CalculateGrade ()
     {
            int age; // local variable
            age = 17;
     }
```

1 Local variable age is created here.

2 Local variable age can only be “seen” in this method.

3 The Local variable is destroyed when the method ends.

Recap about Local Variables

Example :

```
private void CalculateGrade()
{
    // local to Calculate()
    int age=10;

         // call method
         Print100();
         TxtAge.Text = age.ToString()
  }
```

1

Can we print number here?
Txtdisplay.text =
number.ToString()

```
private void Print100()
{
// local to Print100()
   int number = 100;        2

   Txtdisplay.Text = number.ToString()

}    3
```

1 **CalculateGrade() calls Print100()**

2 **Variable *number* is created & used inPrint100()**

3 ***number* is destroyed when Print100() ends**

Class Variables

- **Class variables**
  - Are defined **outside** the method in a class.
  - Exists during the entire existence of the program.
  - Can be accessed by all methods in the same class.

```
namespace topic4B_solution
{
    public partial class q1 : Form
    {
        int hours;
        float pay;
        public q1()
        {
            Initialize Component();
        }

        //method
        private void CalculatePay()
        {

         }
      }
}
```

Recap about Class Variables

Example :

```
int sTotal; // class variable
private void CalculateGrade()
{
    // local to Calculate()
    int age=10;

          // call method
          UpdateTotal();                        1
    4     TxtAge.Text = age.ToString()
          Txtdisplay.Text = sTotal.ToString()
  }
```

Can we print sTotal here?
Txtdisplay.text =
sTotal.ToString()

```
private void UpdateTotal()
{

         sTotal=111;     2

}   3
```

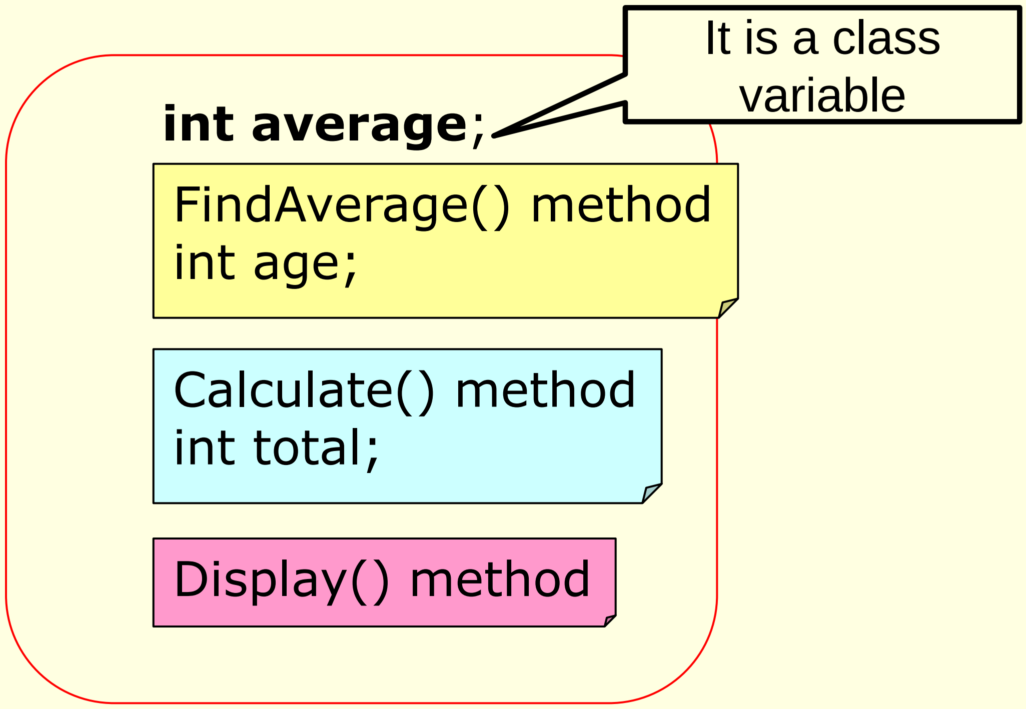
1 **CalculateGrade() calls UpdateTotal()**

2 **Data input is stored in class variable sTotal**

3 **UpdateTotal() ends, but sTotal is NOT destroyed**

4 **stotal is printed out in CalculateGrade()**

Local vs Class Variables



- ✓ average is a **CLASS variable** as it is declared OUTSIDE all the methods. All the methods can Read from and Write to the class variable average.
- ✓ age is a LOCAL variable in **FindAverage** method as it is declared INSIDE **FindAverage**. It is only visible to **FindAverage** method. The other 2 methods (Calculate and Display) DOES NOT know about age.
- ✓ a Local variable is destroyed once the method that contains it completes execution. For example, the Local variable total is destroyed once the Calculate method finish running.

Class Vs Local Variables

- **Guidelines for using local & class variables**
  - Minimize use of class variables
  - Use local variables as far as possible to maintain the reusability of functions

- **Use class variable when**:
  - Data needs to be stored for future computation
    - e.g grandTotal
  - Data needs to be referenced by other methods
    - Sharing data among methods

Class and Local Variables

Example : What is wrong with the following program to read and display the maximum of 2 marks read?

These are local variables!

```
Private void FindMax()
{
int mark1,mark2,max;
   ReadMarks();
   CalculateMax();
}
```

```
private void ReadMarks()
{
   mark1=int.Parse(
          txtMark1.Text);
   mark2=int.Parse(
          txtMark2.Text);
}
private void CalculateMax()
{
   max = mark1;
   if ( mark2 > max)
          max = mark2;
   txtMaxMark.Text =
max.ToString();
}
```

Answer: Local variables mark1 and mark2 are not available in ReadMarks() and CalculateMax() methods

Class and Local Variables

Program should be re-written, as follows.

Will the program work if int max is declared as class variable?

Class variables

```
int mark1, mark2;
```

```
Private void FindMax()
{
    ReadMarks();
    CalculateMax();
}
```

```
private void ReadMarks()
{
   mark1=int.Parse(
           txtMark1.Text);
   mark2=int.Parse(
           txtMark2.Text);
}
private void CalculateMax()
{
   int max;
max = mark1;
   if ( mark2 > max)
           max = mark2;
   txtMaxMark.Text =
max.ToString();
}
```

Local variable

Summary

- Why use methods?
- How to write methods
- What are local and class variables
- How to pass data between methods using class variables

Practical 4B

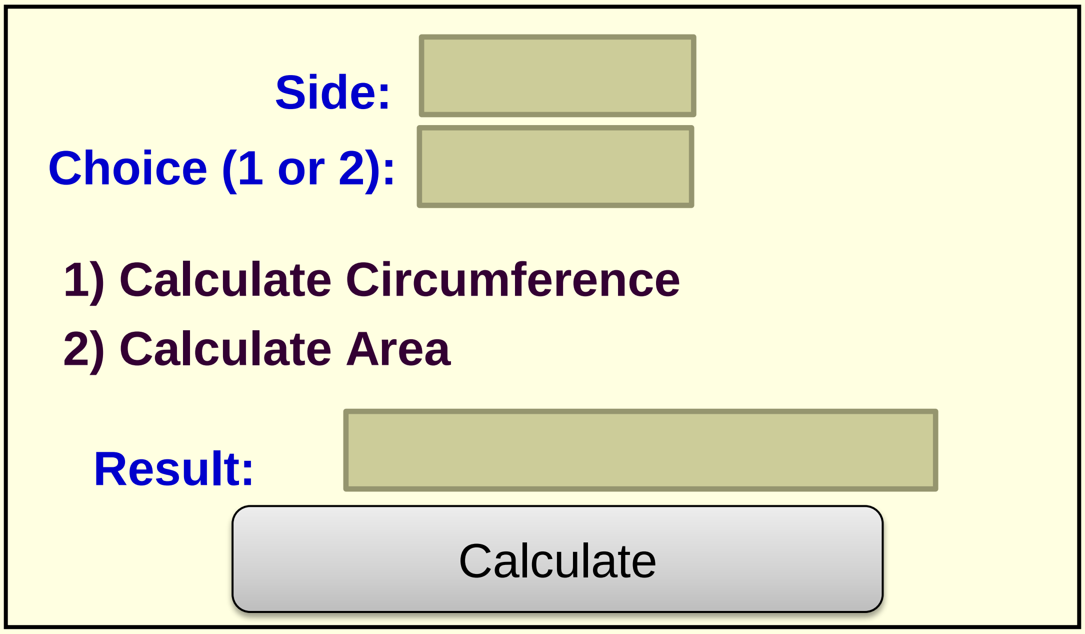
1. Study the below program and find the error/s.

```
//class variables
int hours, pay;

private void CalculatePay( )
{
   int pay;
   if (hours > 50)
       pay = 550 + 20 * hours;
   else if (hours >= 40)
           pay = 400 + 15 * hours;
        else
           pay = 10 * hours;
}
// Assume the control are Calculate button,
//txtPay and txtHour text boxes

private void btnCalculate_click(object sender, EventArgs e )
{

   hours=int.Parse(txthour.Text);
       CalculatePay(); //calling method
       txtPay.Text = pay.ToString();
}
```

Practical 4B

2. Analyze the following program. Identify all the CLASS and LOCAL variables.

When user clicks on Calculate button, program reads 2 values , *side* and *choice* . If *choice* is 1, it calculates and displays the *circumference*, where *circumference* = 4 * *side*. If *choice* is 2, it calculates and displays the *area*, where *area* = *side* * *side*. If *choice* is not 1 or 2, it displays "wrong choice". Both calculations of circumference and area are implemented as methods. Class variables are used to communicate between the methods.



Practical 4B

```
double side;

private void btnCalculate_click(object sender,
    EventArgs e )
{
        int choice;

        side=double.Parse(txtsize.Text);
        choice=int.Parse(txtchoice.Text);

        if (choice == 1)
        {
             CalculateCircumference();
        }
        else if (choice == 2)
        {
            CalculateArea();
        }
        else
        {
             MessageBox.Show("Wrong choice! ");
        }
}
```

Practical 4B

```
private void CalculateCircumference()
{
      double circumference;
      circumference = 4.0 * side;
      resultxt.Text =
      "Circumference = “ +
      circumference.ToString();
}

private void CalculateArea()
{
      double area;
      area = side * side;
      resultxt.Text =
      "Area = “ + area.ToString();
}
```

Practical 4B

3. Write a *window application* to allow user to enter a home loan amount (in dollars). When user clicks on Calculate button, it reads the loan amount and calls a method called *CalculateDeposit* . The method calculates and displays the required deposit based on the schedule below.

Loan ($)	**Deposit ($)**
Greater than $300,000	$10,000 + 15% of loan
$100,000 - $300,000	$5,000 + 10% of loan
Less than $100,000	5% of loan

Write the C# code for both the btnCalculate_click(object sender, EventArgs e) and the *CalculateDeposit* methods. Use a class variable to communicate between the two methods.

4. Write a *window form application* to allow user to enter *t*he weight of a parcel (in kg). When user clicks on Calculate button, it reads the weight and calls a method called *CalculateCharge.* It calculates and displays the postage charge based on the schedule below.

Weight (kg)	**Charge ($/kg)**
Greater than 5.0 kg	$2.30 / kg
2.5 - 5 kg	$2.50 / kg
Less than 2.5 kg	$3.00 / kg

Write the C# code for both the btnCalculate_click(object sender, EventArgs e) and the *CalculateCharge* methods. Use a class variable to communicate between the two methods.

Practical 4B

5. Write a *window application* to allow user to enter the radius of a sphere, displays the following menu with the choices selection using radio buttons.
   - **Calculate Area of Sphere**
   - **Calculate Volume of Sphere**

   When user clicks on Calculate button, the program reads in the **choice , radius** and call the respective method.

   Method *CalculateArea()* calculates and displays the area of the sphere using a message box.

   Area = 4 x PI x radius x radius

   Method *CalculateVolume()* calculates and displays the volume of the sphere using a message box.

   $$\textbf{Volume} = \frac{\textbf{4 x PI x radius x radius x radius}}{\textbf{3}}$$

   Write the C# code for the btnCalculate_click(object sender, EventArgs e) , the *CalculateArea ()* and the *CalculateVolume()* methods. Use a class variable to communicate among the methods.

Practical 4B

6. Write a *window application* to allow user to enter the height (in metres) and weight (in kg) of a person. When user clicks on Calculate button, the program reads in the **<u>height and weight</u>**. It then calls a method called *CalculateBMI()* that calculates the Body Mass Index (BMI) based on the following formula:

$$\textbf{BMI} = \frac{\textbf{weight (kg)}}{\textbf{height (m) x height (m)}}$$

In the *CalculateBMI()* method should then display the relevant message shown below based on the BMI. Display the message using message box.

BMI	**Message to display**
Less than 18.5	Under weight
18.5 to below 25	Normal weight
25 to below 30	Overweight
30 and above	Obese

Write the C# code for the btnCalculate_click(object sender, EventArgs e) and *CalculateBMI ()* methods. Use 2 class variables to communicate between the methods.

End of Topic 4B

Methods with class variable