Methods

- Method calling
  - Namespace
  - Type
  - Method name
  - *using* declarative
    - Adding references

```
namespace myspace
{
    class myclass // data Type
    {
        public static  void mymethod() // Method declaring
        {

         }
    }
}
```

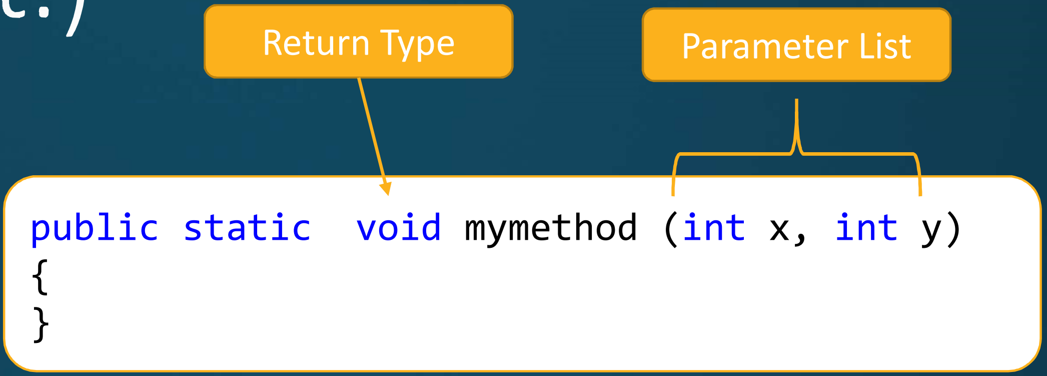
```
myspace.myclass.mymethod();
```

```
using myspace;
{

}…….
myspace.myclass.mymethod();
```

Methods (cont.)

- Declaring Method
  - Scope
  - Parameters
    - Parameter list
  - Return value
  - Call by value
    - Value type
    - Reference type
      - ex: array ,string
- Call using ***ref*** keyword
- Call using ***out*** keyword



```
public static  void mymethod (int x, int y)
{
}
```

```
mymethod (x, y); // call by value
```

```
public static  void mymethod2 (ref int x, ref int y)
{
}
```

```
mymethod2 (ref x, ref y); // call by reference
```

Methods (cont.)

- Passing variable by value
  - Value type

```
static public void swap( ref int k, ref int l)
{
    int temp;
    temp = k;
    k = l;
    l = temp;
}
```

- Passing variable by reference
  - Value type

Methods (cont.)

- Passing variable by value
  - Reference type

```
static void method1( int []a)
{
    a[1] = 100;
}
```

- Passing variable by reference
  - Reference type

```
static void method1(ref int []a)
{
    a = new int[] { 1, 1, 1 };
}
```

Methods (cont.)

- Param Arrays (method parameter)
  - Declaration

```
public static  void mymethod (params int[] x)
{
    x[0]=10;
}
```

  - Calling

```
Int []arr=new int[3];
mymethod(10);
mymethod(10,20);
Mymethod(arr);
```

- Methods in memory (Stack)

Slide 50

w6 int.parse example

wael, 12/15/2016

Methods (cont.)

- Named Argument (parameter)

```
public static void mymethod4(string name,string address)
{
}
```

  - Calling

```
mymethod4(address: "haram street",name:"ahmed");
```

- Optional Arguments

```
public static void mymethod4(string name,string address=“hhhhh”)
{
}
```

  - calling

```
mymethod4(“www”);
mymethod4(“www”, “gggggg”);
```

Methods (cont.)

- Main method parameter and return

```
static int Main(string[] args)
```

```
xyz.exe hello  hhhh
```

Assignments

- Menu Program
  - Convert menu program to methods
    - Method for drawing menu
    - Method for every button
      - Method for display employees
      - Method for add new employee
      - Method for sort employee
  - Adding handling exception for employee data input

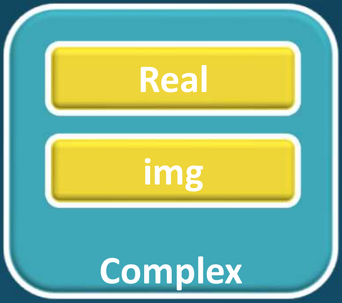
Complex data types

- Structure (Value type)
- Class (reference type)

Structure

- Declare structure data type

```
struct complex
       {
           public float real;
           public float img;
       }
```

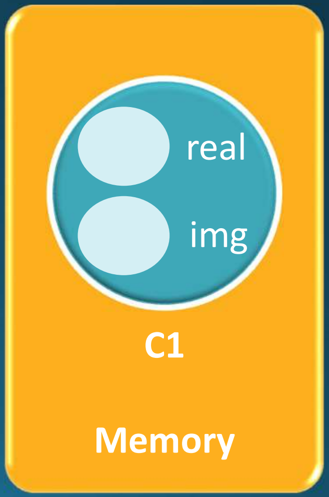


- Declare structure variable
  - Without using new keyword

```
complex c1;
```

  - (using new keyword)

```
complex c2 = new complex(10.5f);
```



w3

Slide 55

w3	acces modifier wael, 11/24/2016

Structure (cont.)

- Access structure members

```
c1.real = 10.0f;
c1.img = 10.5f;
```

- Structures is value type *(passing to method)*

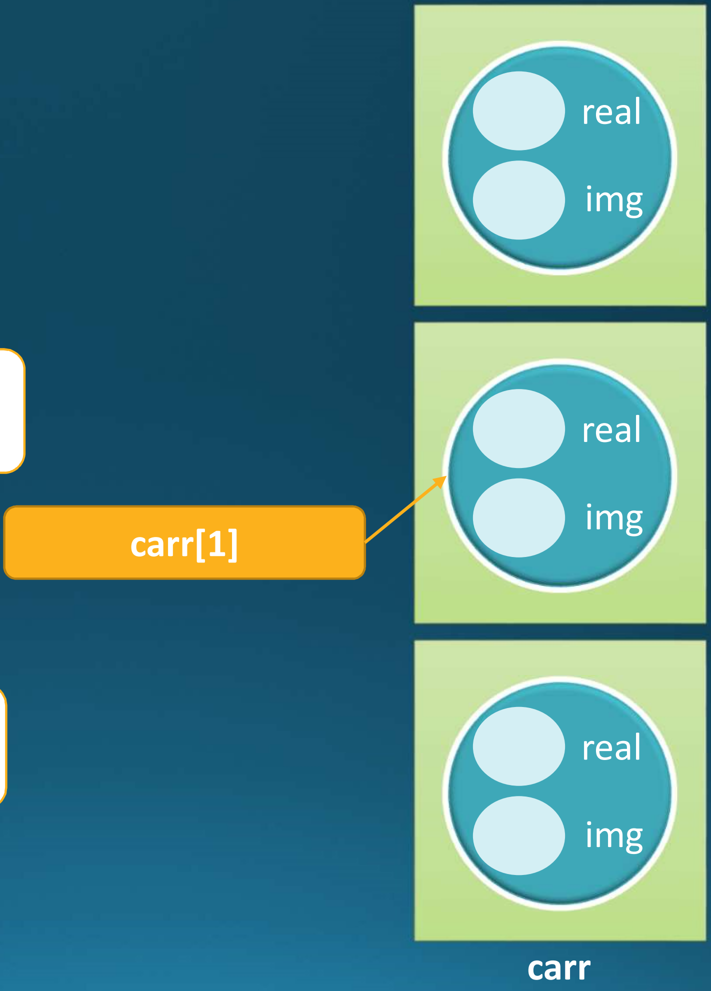
```
static complex AddComplex(complex c1,complex c2)
        {
            complex total;
            total.real = c1.real + c2.real;
            total.img = c1.img + c2.img;
            return total;
        }
```

Structure (cont.)

- Array of structures
  - Declare an array of structuers

  -
    ```
    complex[] carr = new complex[3];
    ```
  - Accessing structure elements

```
carr[0].real=15.7;
```



Assignments

- Write a program to add, subtract two complex

struct
Complex

float real;

float img;