Truyền Thái Y – Pass "Value" to a Function or Method

Pass by value Truyền tham trị	Pass by reference Truyền tham chiếu	Pass a delegate (A special kind of pass by reference – refer to a method)
• Give me a "single value" • Cho em xin 1 con số, một giá trị đơn giản nào đó	• Give me an object {data | methods} • Cho em xin một đối tượng mà bên trong chứa: { nhiều data và nhiều hàm }	• Give me a method – Give me a source code – Give me a "your working process" – Give me a "your condition to process" • Cho em xin đoạn 1 code – Gửi cho em, hoặc chỉ cho em cách bác muốn công việc được xử lí như thế nào – Gửi cho em 1 hàm duy nhất!

• Pass a **"single"** value
5 10 15 20
	• Pass an **object**: with many data & methods: x = new Student(…){};
	{ SE1, An, 2003, F1(), Show() }
		• Pass a **method only**
		(method is passed to a method)
		void DoThing()
		{
		code here ...
		code outside;
		}

void DoSomething(int n)
{
work on n
int result = n */+-…;
...
}
	void DoSomething(Student s)
	{
	work on s
	call many s' methods
	call many s' source code
	s.Show();
	code inside...
	}
		void DoSomething(Action f, Func f)
		{
		f points to a "source code outside"
		{... code outside}
		run f() means run code {... outside}
		and back to run code inside
		code inside ...
		}

Call/Use DoSomething()
DoSomeThing(5); //n = 5
DoSomeThing(10); //n = 10
	Call/Use DoSomething()
	Student x = new Student(…){…};
	DoSomething(x); //s = x
	DoSomething(new Student(…));
	1. Create object
	2. Pass object to the method and run
		Call/Use DoSomething()
		void DoThing()
		{
		code here ...
		code outside;
		}
		DoSomething(DoThing)
		1. Create method outside
		2. Pass outside method to the DoSomething() and run
		3. DoSomething() run, DoThing() run

Truyền Thái Y – Pass "Delegate" to a Function or Method

Pass a void delegate Action, Action< . . . params>	Pass a function delegate Func< . . . params >	Pass a bool function delegate Predicate< only one param >
• Give me a void method, a method just does something • Cho em xin 1 cái hàm mà nó làm gì đó đi, hàm không trả về gì cả!	• Give me method that returns a value • Cho em xin một hàm mà hàm này sẽ trả về giá trị/con số nào đó á	• Give me a method that takes an input, and will return true/false based on some criteria or conditions. • Cho em xin 1 cái hàm mà biết trả ra đúng sai khi hàm này nhận một đầu vào

• Pass a "void" method
`void DoThing(???)`
`{`
`  //code here...`
`  code outside; ☺`
`}`
	• Pass an object: with many data & methods
	`int DoThing(???)`
	`{`
	`  //code here...`
	`  code outside;`
	`  return ?;`
	`}`
		• Pass a method **only** (method is passed to a method)
		`bool DoThing(?)`
		`{`
		`  //code here...`
		`  code outside;`
		`  return true/false;`
		`}`

`void DoSomething(Action<> f)`
`void DoSomething(Action f)`
`{`
`  f points to a "source code outside"`
`  {... code outside}`
`  run f(???) means`
`  run code {... outside}`
`  and back to run code inside`
`  code inside ...`
`}`
	`void DoSomething(Func<?> f)`
	`{`
	`  f points to a "source code outside"`
	`  {... code outside}`
	`  run f(???) means`
	`  run code {... outside}`
	`  and back to run code inside`
	`  code inside ...`
	`}`
		`void DoSomething(Predicate<?> f)`
		`{`
		`  f points to a "source code outside"`
		`  {... code outside}`
		`  run if(f(?)== true) means`
		`  run code {... outside}`
		`  and back to run code inside`
		`  code inside ...`
		`}`

Delegte in Use – Sử dụng Delegate thế nào? (v.24.0305)

```
public class Cabinet_Container
{
  List<int> _arr = [1] [3] [5] [9] [2] [4] [6] [8] [7] [...]

  void PrintEvenNumbers()
  {
    foreach (var x in _arr)
    {
                    //x = [1] [3] [5] [9] [2] [4] [6] [8] [7] [...]
      if (x % 2 == 0)
      {
        Console.Write(x + " ");
      }
    }
  }

  void PrintoOddNumbers()
  {
    foreach (var x in _arr)
    {
      if (x % 2 != 0)
      {
        Console.Write(x + " ");
      }
    }
  }

  void PrintDivisibleBy5()
  {
    foreach (var x in _arr)
    {
      if (x % 5 == 0)
      {
        Console.Write(x + " ");
      }
    }
  }
}
```

```
public class Cabinet_Container
{
  List<int> _arr = [1] [3] [5] [9] [2] [4] [6] [8] [7] [...]

  void PrintNumsOnDemand(Action<int> f)
  {
    foreach (var x in _arr)
    {
                    //x = [1] [3] [5] [9] [2] [4] [6] [8] [7] [...]
      f(x); //call f() outside
    }
  }

  void PrintNumsOnDemand(Action<int> f)
  {
    foreach (var x in _arr)
    {
      f(x); //call f() outside
    }
  }

  void PrintNumsOnDemand(Action<int> f)
  {
    foreach (var x in _arr)
    {
      f(x); //call f() outside
    }
  }
}
```

```
void PrintEven(int X)
{
  if (X % 2 == 0)
  {
    Console.Write(X + " ");
  }
}
```

```
void PrintOdd(int X)
{
  if (X % 2 != 0)
  {
    Console.Write(X + " ");
  }
}
```

```
void PrintDivisableBy5(int X)
{
  if (X % 5 == 0)
  {
    Console.Write(X + " ");
  }
}
```

Delegte in Use – Sử dụng Delegate thế nào? (v.24.0305)

```
public class Cabinet_Container
{
  List<int> _arr = 1 3 5 9 2 4 6 8 7 …

  void PrintEvenNumbers()
  {
    foreach (var x in _arr)
    {
                        //x = 1 3 5 9 2 4 6 8 7 …
      if (x % 2 == 0)
      {
        Console.Write(x + " ");
      }
    }
  }

  void PrintoOddNumbers()
  {
    foreach (var x in _arr)
    {
      if (x % 2 != 0)
      {
        Console.Write(x + " ");
      }
    }
  }

  void PrintDivisibleBy5()
  {
    foreach (var x in _arr)
    {
      if (x % 5 == 0)
      {
        Console.Write(x + " ");
      }
    }
  }
}
```

```
public class Cabinet_Container
{
  List<int> _arr = 1 3 5 9 2 4 6 8 7 …

  void PrintOnDemand(Predicate<int> f)
  {
    foreach (var x in _arr)
    {
                        //x = 1 3 5 9 2 4 6 8 7 …
      if (f(x) == true)
      {
        Console.Write(x + " ");
      }
    }
  }

  void  PrintOnDemand(Predicate<int> f)
  {
    foreach (var x in _arr)
    {
      if (f(x) == true)
      {
        Console.Write(x + " ");
      }
    }
  }

  void PrintOnDemand(Predicate<int> f)
  {
    foreach (var x in _arr)
    {
      if (f(x) == true)
      {
        Console.Write(x + " “);
      }
    }
  }
}
```

```
bool IsEven(int X)
{
  if (X % 2 == 0)
     return true;
  return false;
} //try with return X % 2 == 0;
```

```
bool IsOdd(int X)
{
  if (X % 2 != 0)
     return true;
  retun false;
} //try with return X % 2 != 0;
```

```
bool IsDivisableBy5(int X)
{
  if (X % 5 == 0)
     return true;
  return false;
} //try with return X % 5 == 0;
```