

Lab 06

Exception Handling

Events and Delegate

Mục tiêu

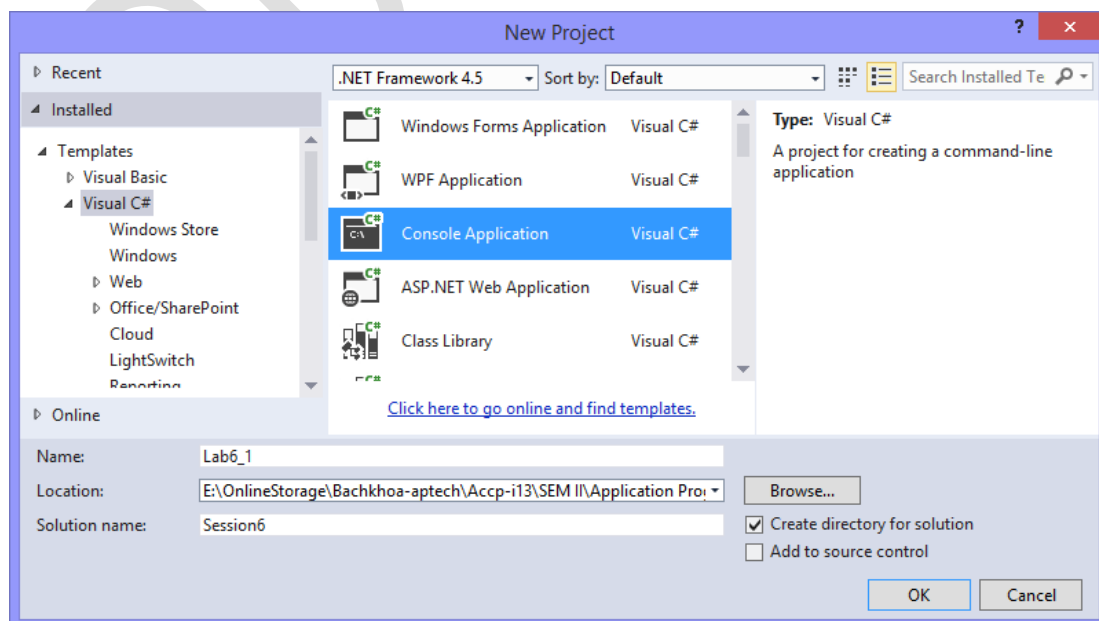
- Bắt và điều khiển ngoại lệ
- Tạo custom ngoại lệ
- Tạo và sử dụng Delegate
- Tạo và sử dụng Sự kiện

Phần I Bài tập step by step

Bài 6.1

Viết chương trình C# minh họa một số các ngoại lệ như: `FormatException`, `OverflowException`, `IndexOutOfRangeException`.

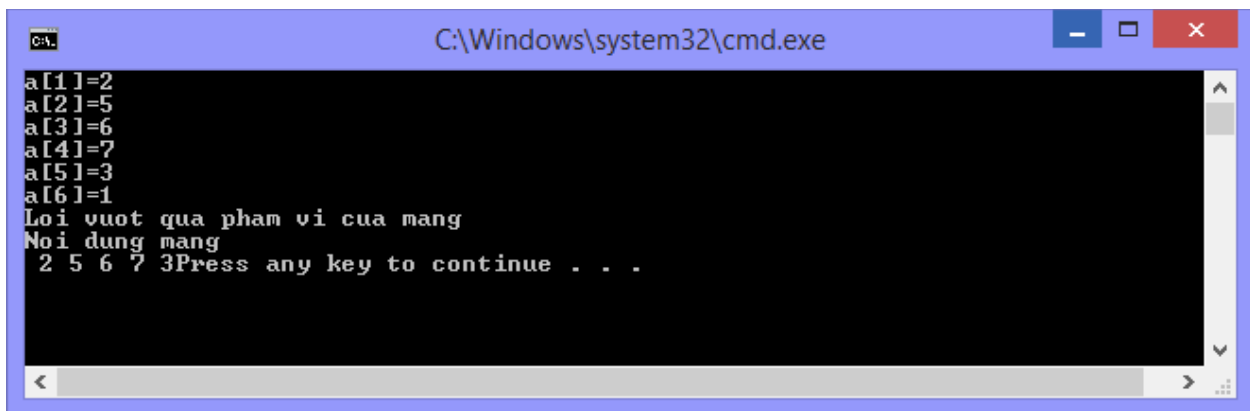
Bước 1: Mở Visual Studio 2013, vào menu File -> New -> Project -> chọn loại project “Console Application”, nhập tên project, tên solution -> OK.



Bước 2: Mở tệp Program.cs và code cho hàm Main theo gợi ý sau:

```
static void Main(string[] args)
{
    //khai báo mảng
    byte[] a = new byte[5];
    //nhập mảng
    try
    {
        for (int i = 0; i <= 5; i++)
        {
            Console.Write("a[{0}]=", i + 1);
            a[i] = Convert.ToByte(Console.ReadLine());
        }
    }
    catch (FormatException ex)
    {
        //Console.WriteLine(ex.Message);
        Console.WriteLine("Khong duoc nhap ki tu cho mang so");
    }
    catch (OverflowException ex)
    {
        //Console.WriteLine(ex.Message);
        Console.WriteLine("Khong duoc nhap gia tri nam ngoai mien 0-255");
    }
    catch (IndexOutOfRangeException ex)
    {
        //Console.WriteLine(ex.Message);
        Console.WriteLine("Loi vuot qua pham vi cua mang");
    }
    //in mảng
    Console.WriteLine("Noi dung mang");
    for (int i = 0; i < 5; i++)
        Console.Write(" {0}", a[i]);
}
```

Bước 3: Nhấn Ctrl+F5 để chạy và xem kết quả



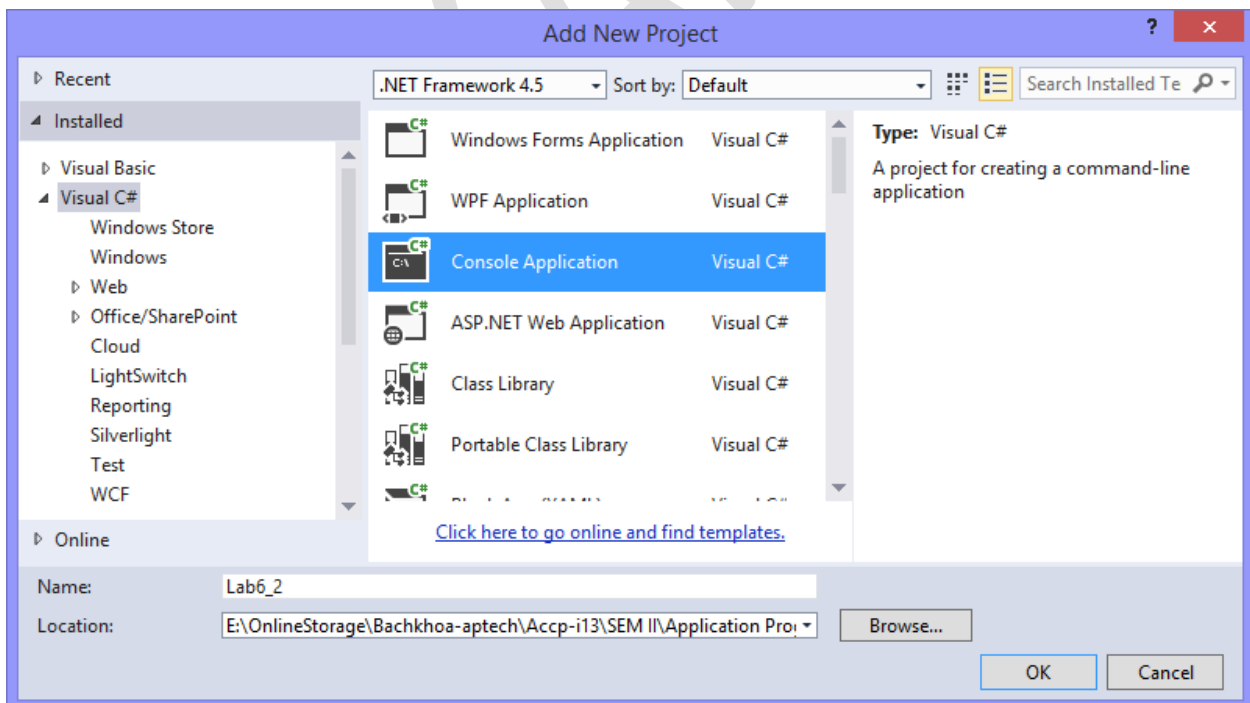
```
C:\Windows\system32\cmd.exe
a[1]=2
a[2]=5
a[3]=6
a[4]=7
a[5]=3
a[6]=1
Lỗi vượt qua phạm vi của mảng
Nội dung mảng
2 5 6 7 3
Press any key to continue . . .
```

Lưu ý: hãy thử nhập sai dữ liệu và quan sát thông báo về lỗi.

Bài 6.2

Viết chương trình C# minh họa ngoại lệ do người dùng định nghĩa sau đó tung và bắt ngoại lệ có sử dụng khối finally.

Bước 1: Kích chuột phải vào Solution “Session6” chọn Add -> New Project -> nhập tên.



Bước 2: Mở tệp Program.cs và code theo gợi ý sau:

```
//định nghĩa lớp custom ngoại lệ chỉ cho nhập số nguyên dương
```

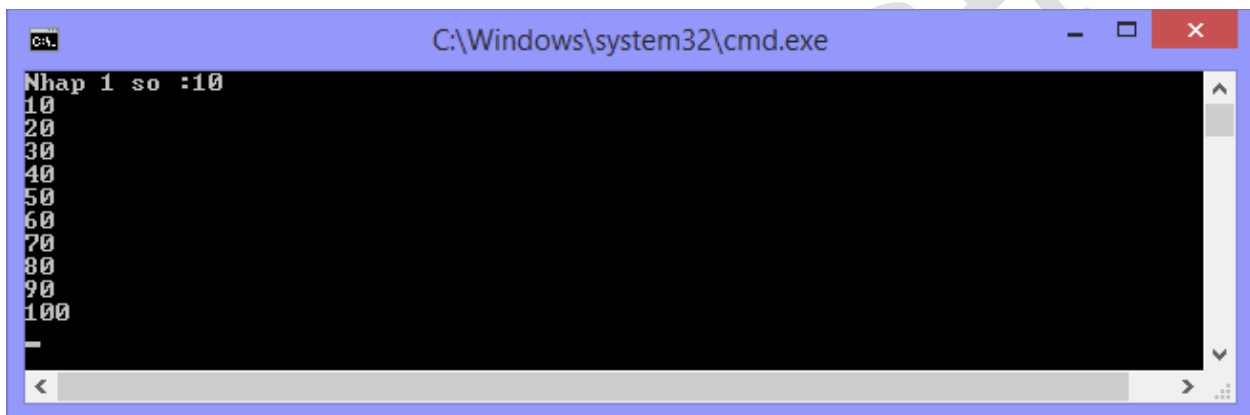
```
public class InvalidInputNumber : Exception
{
    public InvalidInputNumber()
        : base("Hay nhap 1 so >0") { }
}

class Program
{
    static void Main(string[] args)
    {
        int intCnt;
        int intNum = 0;
        Console.Write("Nhap 1 so :");
        //nhập và tung ngoại lệ
        try
        {
            intNum = Convert.ToInt32(Console.ReadLine());
            if (intNum <= 0)
            {
                throw new InvalidInputNumber();
            }
        }
        catch (InvalidInputNumber objInvalidInput)
        {
            Console.WriteLine(objInvalidInput.Message);
        }
        catch (System.FormatException objFormatException)
        {
            Console.WriteLine(objFormatException.Message);
        }
        finally
        {
            if (intNum > 0)
            {
                //in ra dãy số *100
                for (intCnt = 1; intCnt <= 10; intCnt++)
            }
        }
    }
}
```

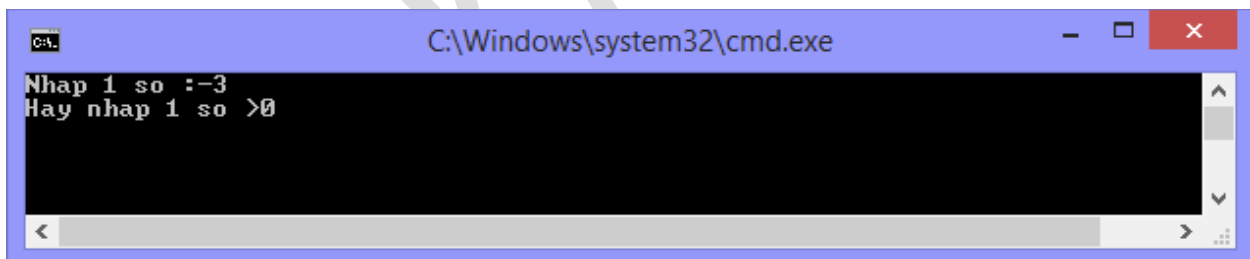
```
        Console.WriteLine(intCnt * intNum);  
    }  
}  
Console.ReadLine();  
}  
}
```

Bước 3: Nhấn Ctrl+F5 để chạy và xem kết quả

- Trường hợp không lỗi



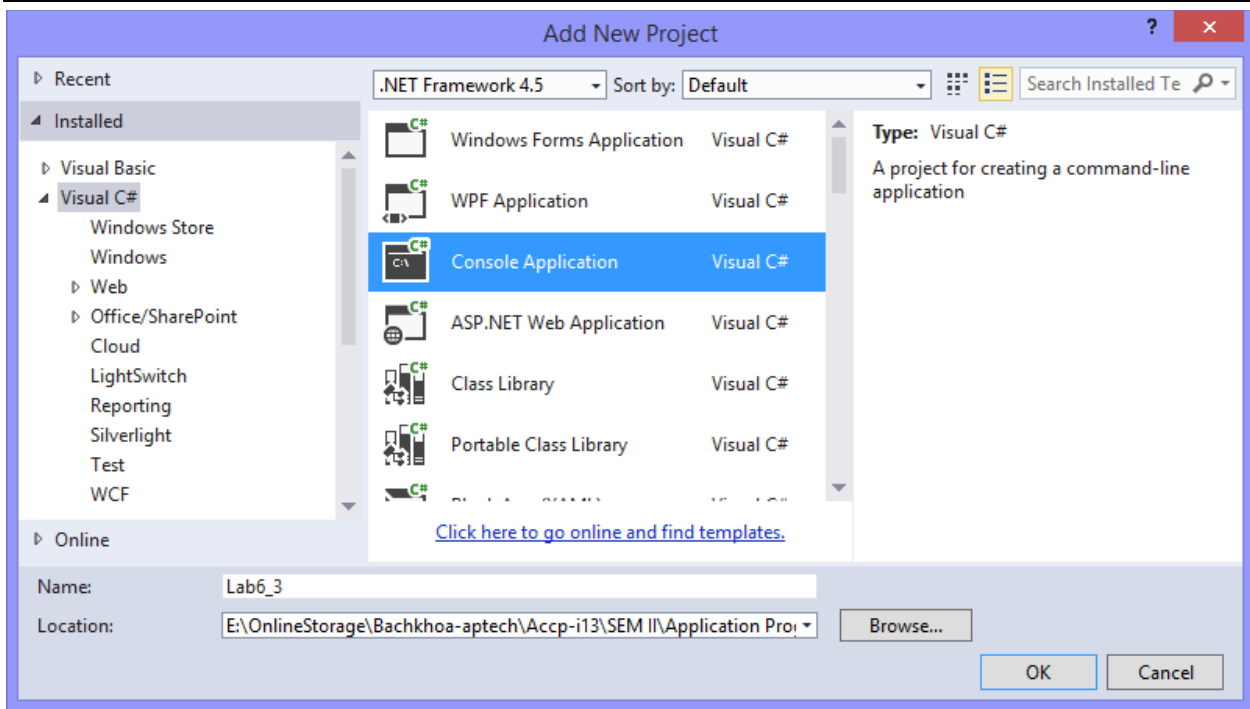
- Trường hợp có lỗi



Bài 6.3

Viết chương trình C# minh họa việc sử dụng Delegate và Multicast Delegate.

Bước 1: Kích chuột phải vào Solution “Session6” chọn Add -> New Project -> nhập tên.

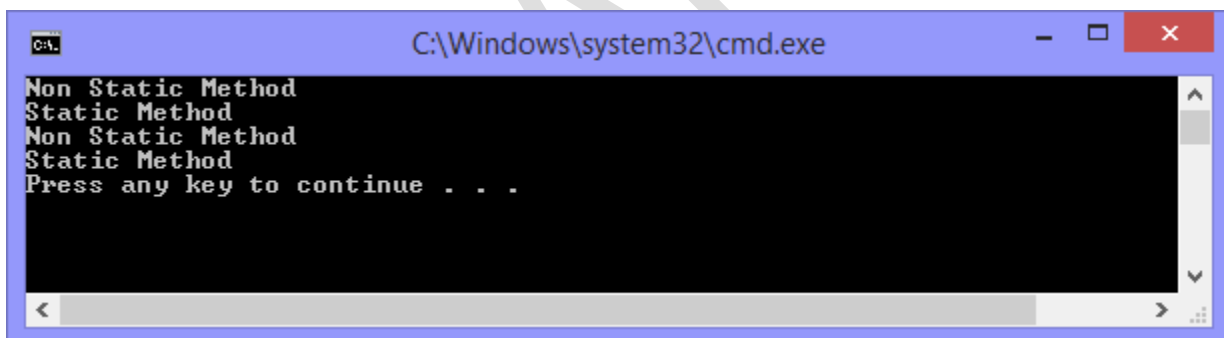


Bước 2: Mở tệp Program.cs và code theo gợi ý sau:

```
//định nghĩa delegate
delegate void FunctionToCall();
//định nghĩa lớp
class MyClass
{
    //định nghĩa phương thức không tĩnh
    public void nonStaticMethod()
    {
        Console.WriteLine("Non Static Method");
    }
    //định nghĩa phương thức tĩnh
    public static void staticMethod()
    {
        Console.WriteLine("Static Method");
    }
}
class Program
{
    static void Main()
```

```
{  
    //khởi tạo đối tượng MyClass  
    MyClass t = new MyClass();  
    //tạo delegate tham chiếu tới phương thức không tĩnh  
    FunctionToCall functionDelegate = t.nonStaticMethod;  
    //cộng tiếp với phương thức tĩnh  
    functionDelegate += MyClass.staticMethod;  
    //cộng tiếp với phương thức không tĩnh  
    functionDelegate += t.nonStaticMethod;  
    //cộng tiếp với phương thức tĩnh  
    functionDelegate += MyClass.staticMethod;  
    //thực thi delegate -> tất cả các phương thức tham chiếu tới đều  
    //được gọi  
    functionDelegate();  
}  
}
```

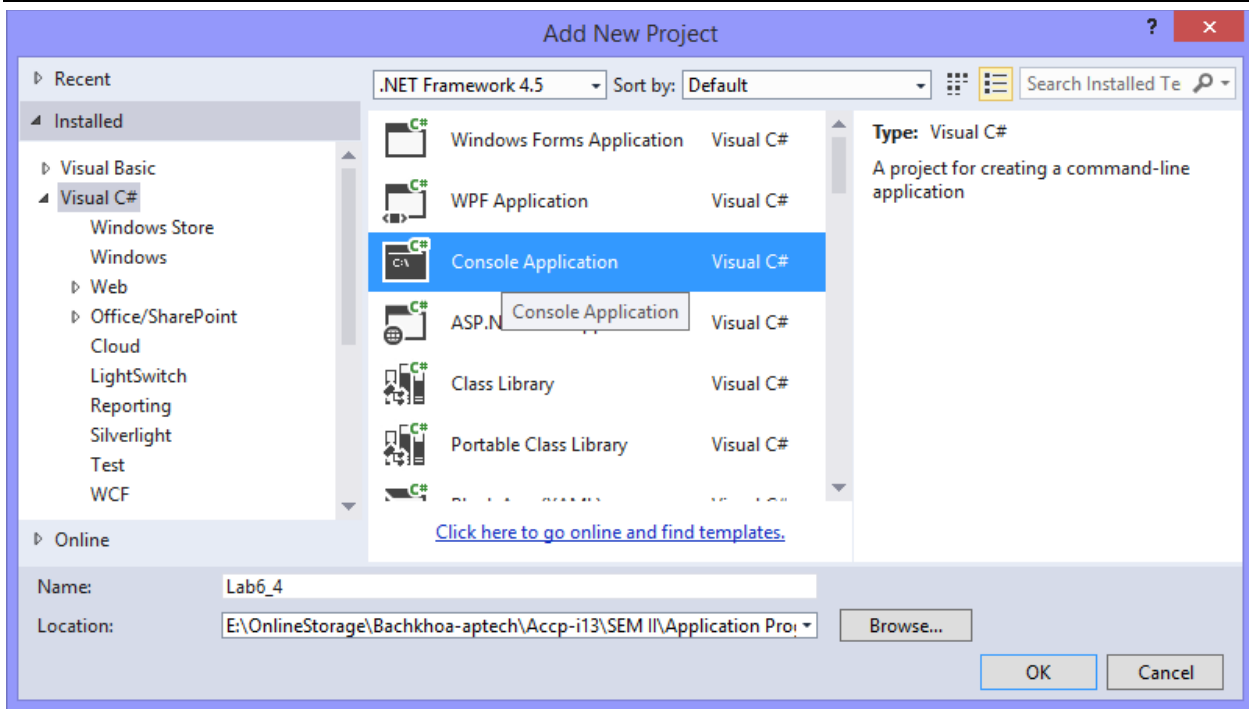
Bước 3: Nhấn Ctrl+F5 để chạy và xem kết quả



Bài 6.4

Viết chương trình C# minh họa việc sử dụng Delegate và Multicast Delegate trong đó có sử dụng tham số với từ khóa ref.

Bước 1: Kích chuột phải vào Solution "Session6" chọn Add -> New Project -> nhập tên.



Bước 2: Mở tệp Program.cs và code theo gợi ý sau:

```
//khai báo delegate
delegate void FunctionToCall(ref int X);

class Program
{
    //định nghĩa phương thức Add2 với tham số sử dụng từ khóa ref
    public static void Add2(ref int x)
    {
        x += 2;
    }

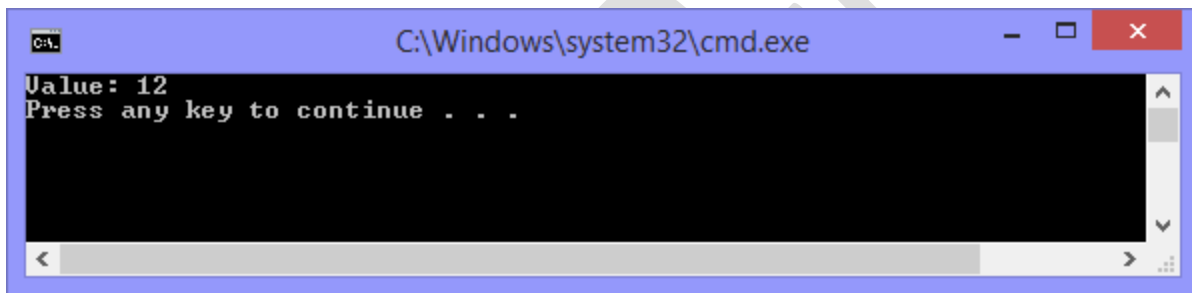
    //định nghĩa phương thức Add3 với tham số sử dụng từ khóa ref
    public static void Add3(ref int x)
    {
        x += 3;
    }

    static void Main(string[] args)
    {
        //gán delegate
    }
}
```



```
FunctionToCall functionDelegate = Add2;
//cộng tiếp delegate
functionDelegate += Add3;
//cộng tiếp delegate
functionDelegate += Add2;
//khai báo biến x
int x = 5;
//gọi delegate và truyền tham số sử dụng ref
functionDelegate(ref x);
//in kết quả
Console.WriteLine("Value: {0}", x);
}
}
```

Bước 3: Nhấn Ctrl+F5 để chạy và xem kết quả

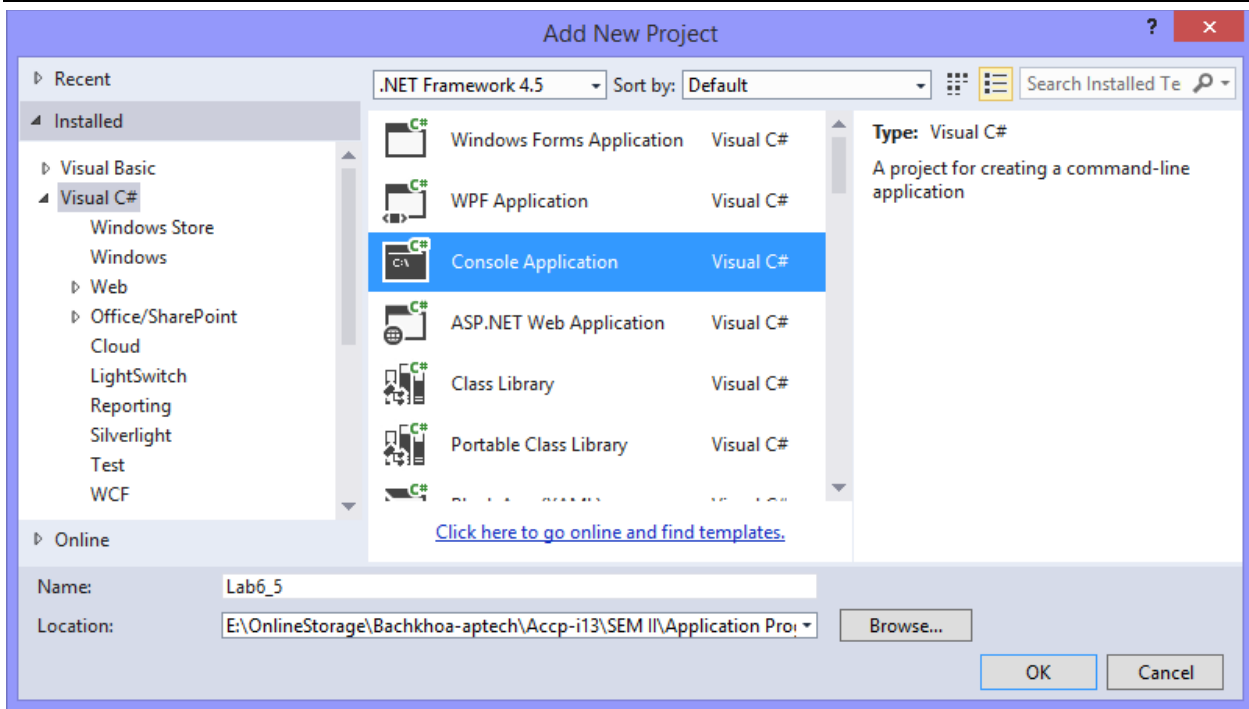


Bài 6.5

Viết chương trình C# với yêu cầu sau:

- Duyệt dãy số từ 1-50
- Hãy định nghĩa delegate, đăng ký sự kiện để mỗi khi gặp một số lẻ thì phát ra sự kiện thấy số lẻ và in lên màn hình.

Bước 1: Kích chuột phải vào Solution "Session6" chọn Add -> New Project -> nhập tên.

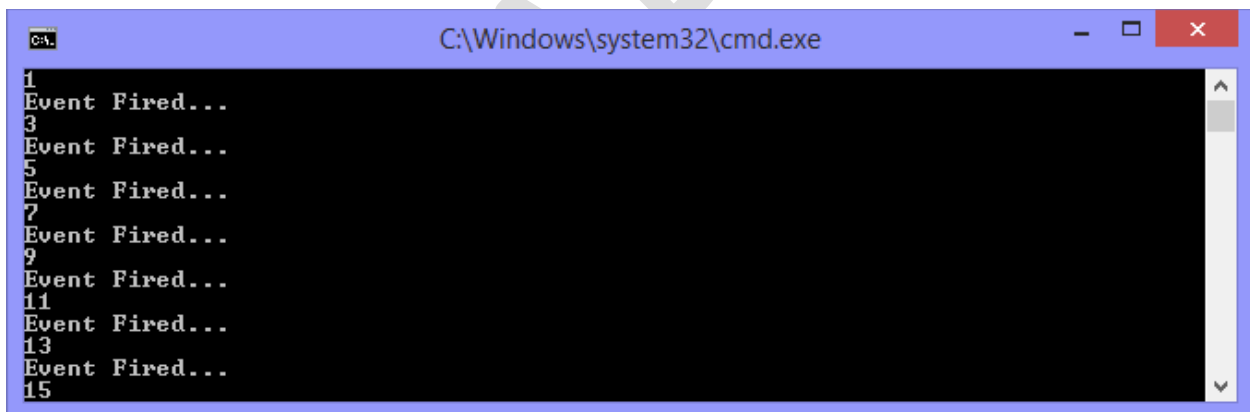


Bước 2: Mở tệp Program.cs và code theo gợi ý sau:

```
//khai báo delegate
public delegate void OddNumberFinder(int num);
//định nghĩa lớp kiểm tra số lẻ
class TestOdd
{
    //khai báo sự kiện cho việc tìm thấy số lẻ
    public event OddNumberFinder OnOddNumber;
    //phương thức đếm số lẻ
    public void CountOdd()
    {
        int odd;
        for (odd = 1; odd < 50; odd += 2)
            //gọi sự kiện kiểm tra số lẻ
            OnOddNumber(odd);
    }
    //định nghĩa phương thức hiển thị số lẻ
    public void ShowOdd(int odd)
    {
        Console.WriteLine(odd);
    }
}
```

```
        Console.WriteLine("Event Fired...");
    }
}
class Program
{
    public static void Main()
    {
        //Khởi tạo đối tượng lớp TestOdd
        TestOdd objOdd = new TestOdd();
        //Đăng ký sự kiện
        objOdd.OnOddNumber += new OddNumberFinder(objOdd.ShowOdd);
        //gọi phương thức đếm số lẻ
        objOdd.CountOdd();
    }
}
```

Bước 3: Nhấn Ctrl+F5 để chạy và xem kết quả



Phần II Bài tập tự làm

Bài 6.1: Viết chương trình C# minh họa việc sử dụng các ngoại lệ

- InvalidCastException.
- IndexOutOfRangeException.
- ArrayTypeMismatchException.

Bài 6.2: Viết chương trình C# với các yêu cầu sau:

- Tạo lớp ngoại lệ tự định nghĩa InvalidMarkException.
- Tạo lớp Student với các thông tin (id, name , theorymark, labmark).
- Tạo thuộc tính cho các thông tin trên.
- Nếu theorymark(điểm lý thuyết) hoặc labmark(điểm thực hành) nhập vào nằm ngoài đoạn 0-10 thì tung ngoại lệ InvalidMarkException.
- Test các ngoại lệ trên.

Bài 6.3: Viết chương trình C# với các yêu cầu sau:

- Tạo lớp Lecture với các thông tin Name, Salary, Bonus
- Viết các phương thức thuộc tính cơ bản cho lớp Lecture.
- Tạo lớp custom exception **AmountException** để điều khiển nghiệp vụ
 - o Khi giảng viên lương (Salary) thấp hơn 60,000\$
 - o Hoặc thưởng (Bonus) nhiều hơn 10,000\$
- Test chương trình với ngoại lệ trên.

Bài 6.4: Viết chương trình C# với yêu cầu sau:

- Tạo lớp TestDelegate.cs, khai báo một delegate kiểu IntAction có kiểu trả về void và nhận tham số kiểu int.
- Khai báo một phương thức tĩnh PrintInt có kiểu trả về void và một tham số kiểu int để in tham số đó lên console.
- Khai báo biến act kiểu IntAction và gán cho phương thức PrintInt (như là một delegate).
- Gọi phương thức act(42).
- Khai báo một phương thức static void Perform(IntAction act, int[] arr) { ... } mà áp dụng delegate act cho mọi phần tử của mảng.
- Sử dụng lệnh foreach để thực thi phương thức Perform. Tạo mảng số nguyên arr và gọi phương thức Perform(PrintInt, arr).

Bài 6.5: Viết chương trình C# minh họa việc lưu trữ danh sách nhân viên trong công ty với các yêu cầu sau:

-
- Tạo lớp Employee (EmpNo, Name, Email, Phone) lưu thông tin nhân viên
 - Tạo lớp Company lưu danh sách nhân viên
 - Tạo delegate và event để điều khiển sự kiện khi có một nhân viên mới được thêm vào công ty, nếu tên tồn tại thì phát ra sự kiện ErrorInput, nếu tên chưa tồn tại thì phát ra sự kiện Success và thông báo.
-

HẾT

@BKAP.PC#