

# 예외 처리

HyoJoon Han  
동국대학교  
han6343@dongguk.edu



예외와 기본 예외 처리

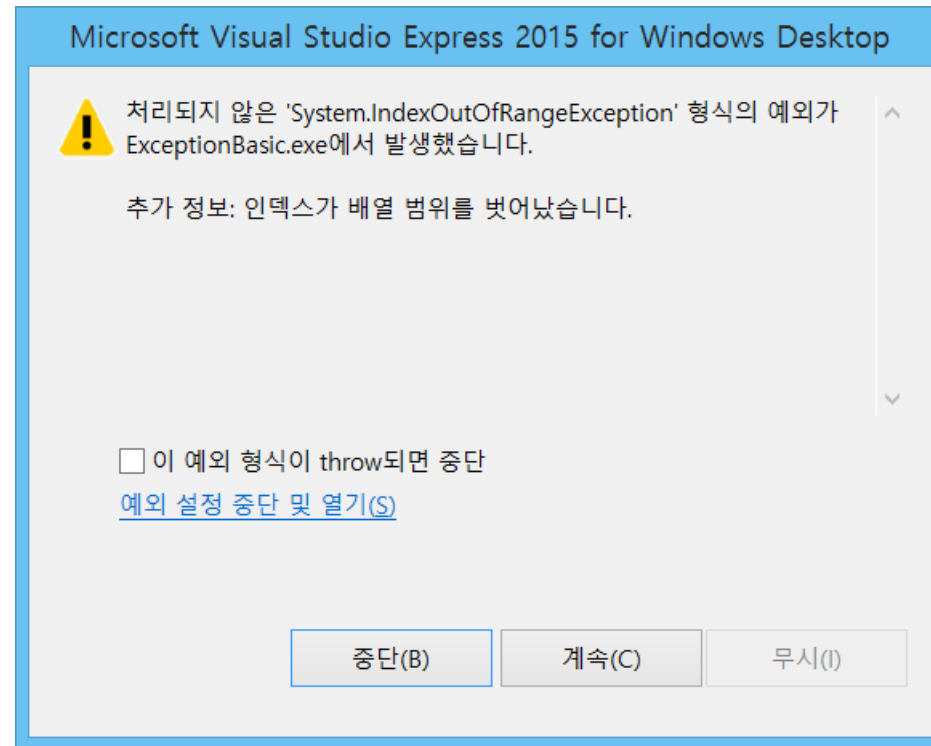
고급 예외 처리

예외 객체

예외 객체를 사용한 예외 구분

예외 강제 발생

- 예외 (Exception) : 프로그램 실행 중 프로그램이 중단되는 오류
- 예외 처리 (Exception Handling) : 오류를 대처할 수 있게 하는 것
  - 기본 예외 처리, 고급 예외 처리

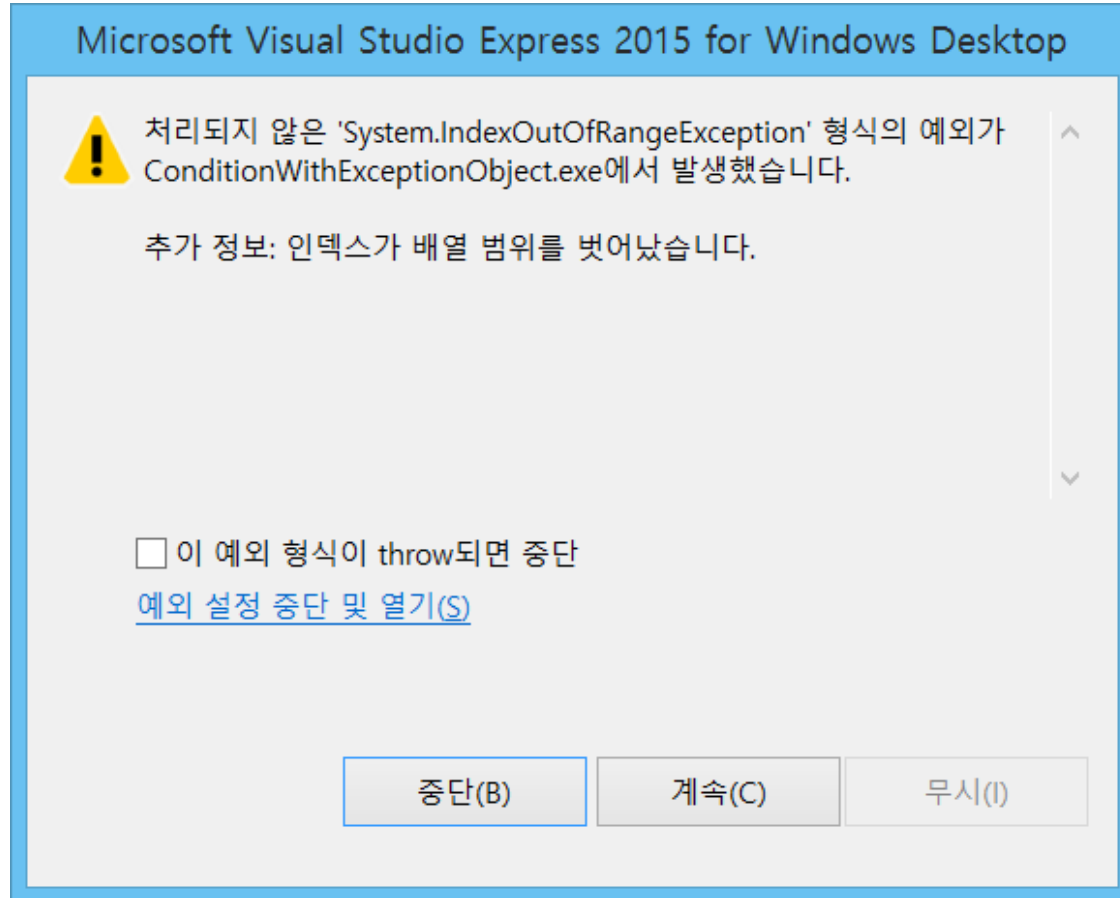


- 에러 (Error) : 프로그램이 컴파일조차 안 되게 하는 프로그래밍 언어상의 문법적 오류

- 예외 상황 확인하기

코드 10-1 예외 상황 확인

```
static void Main(string[] args)
{
    string[] array = { "가", "나" };
    Console.Write("숫자를 입력해주세요: ");
    int input = int.Parse(Console.ReadLine());
    Console.WriteLine("입력한 위치의 값은 '" + array[input] + "'입니다.");
}
```

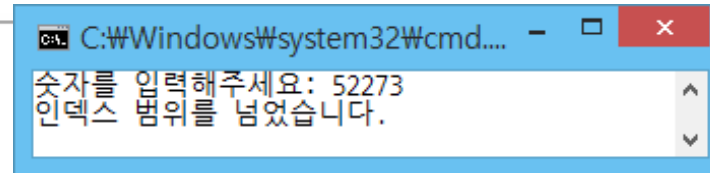


- 기본 예외 처리하기
  - 배열 길이 확인해서 입력 값이 배열 길이 넘으면, 잘못되었다고 알려 줌

코드 10-2 기본 예외 처리

```
static void Main(string[] args)
{
    string[] array = { "가", "나" };
    Console.Write("숫자를 입력해주세요: ");
    int input = int.Parse(Console.ReadLine());

    if (input < array.Length)
    {
        Console.WriteLine("입력한 위치의 값은 '" + array[input] + "'입니다.");
    }
    else
    {
        Console.WriteLine("인덱스 범위를 넘었습니다.");
    }
}
```



- try 키워드, catch 키워드, finally 키워드로 예외를 처리하는 방법
- 고급 예외 처리 형식 (try catch finally 구문)

```
try
{
    // 예외가 발생하면
}
catch (Exception exception)
{
    // 여기서 처리합니다.
}
finally
{
    // 여기는 무조건 실행합니다.
}
```

- catch 구문 또는 finally 구문이 필요 없을 때

```
try
{
    // 예외가 발생하면
}
catch (Exception exception)
{
    // 여기서 처리합니다.
}
```

```
try
{
    // 예외가 발생하면 그냥 넘어갑니다.
}
finally
{
    // 여기는 무조건 실행합니다.
}
```



- Parse() 메서드 예외 처리
  - 예외 상황 확인하기

코드 10-3 예외 상황 확인

```
static void Main(string[] args)
{
    Console.Write("입력: ");
    string input = Console.ReadLine();

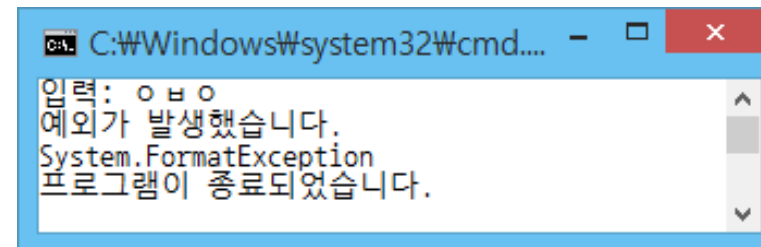
    int index = int.Parse(input);
    Console.WriteLine("입력 숫자: " + index);
}
```

- 고급 예외 처리하기

코드 10-4 고급 예외 처리

```
static void Main(string[] args)
{
    Console.Write("입력: ");
    string input = Console.ReadLine();

    try
    {
        int index = int.Parse(input);
        Console.WriteLine("입력 숫자: " + index);
    }
    catch (Exception exception)
    {
        Console.WriteLine("예외가 발생했습니다.");
        Console.WriteLine(exception.GetType());
    }
    finally
    {
        Console.WriteLine("프로그램이 종료되었습니다.");
    }
}
```



- finally 구문
  - "프로그램이 종료되었습니다." 라는 글자 출력 안됨

코드 10-5 finally 구문을 사용하지 않은 코드

/10장/Exceptions

```
static void Main(string[] args)
{
    Console.Write("입력: ");
    string input = Console.ReadLine();

    try
    {
        int index = int.Parse(input);
        Console.WriteLine("입력 숫자: " + index);
    }
    catch (Exception exception)
    {
        Console.WriteLine("예외가 발생했습니다.");
        Console.WriteLine(exception.GetType());
    }

    Console.WriteLine("프로그램이 종료되었습니다.");
}
```

- 결과가 달라지는 경우
  - catch 구문 내부에서 return 키워드 만날 때
  - catch 구문 내부에서 try catch 구문 사용했는데 break 또는 continue 키워드 만날 때

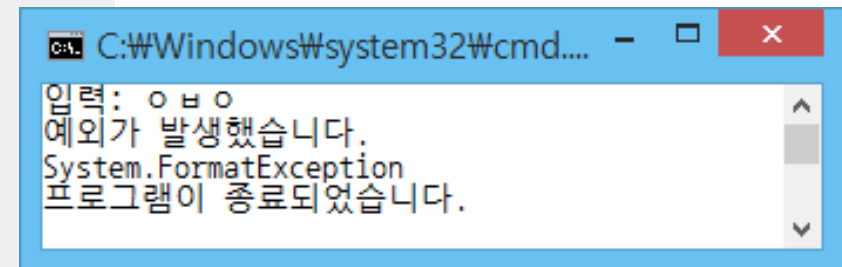
코드 10-6 finally 구문 사용

/10장/Exceptions

```
static void Main(string[] args)
{
    Console.Write("입력: ");
    string input = Console.ReadLine();

    try
    {
        int index = int.Parse(input);
        Console.WriteLine("입력 숫자: " + index);
    }
    catch (Exception exception)
    {
        Console.WriteLine("예외가 발생했습니다.");
        Console.WriteLine(exception.GetType());
        return;
    }
    finally
    {
        Console.WriteLine("프로그램이 종료되었습니다.");
    }
}
```

return 키워드를 사용해 여기서 코드를 종료합니다.



```
C:\Windows\system32\cmd.exe
입력: 0 0 0
예외가 발생했습니다.
System.FormatException
프로그램이 종료되었습니다.
```

- finally 구문과 return 키워드
  - finally 구문 내부는 무조건 실행하고 끝낸다는 규칙
  - 중간에 구문을 벗어나는 키워드들은 불가능

```
try
{

}
catch (Exception exception)
{

}
finally
{
    return;
}
```

제어가 finally 절의 본문을 벗어날 수 없습니다.

그림 10-3 finally 구문 내부에서의 return 키워드 오류

- 예외 발생 시 어떤 예외가 발생 했는지와 관련된 정보 함께 전달해주는 객체
- 예

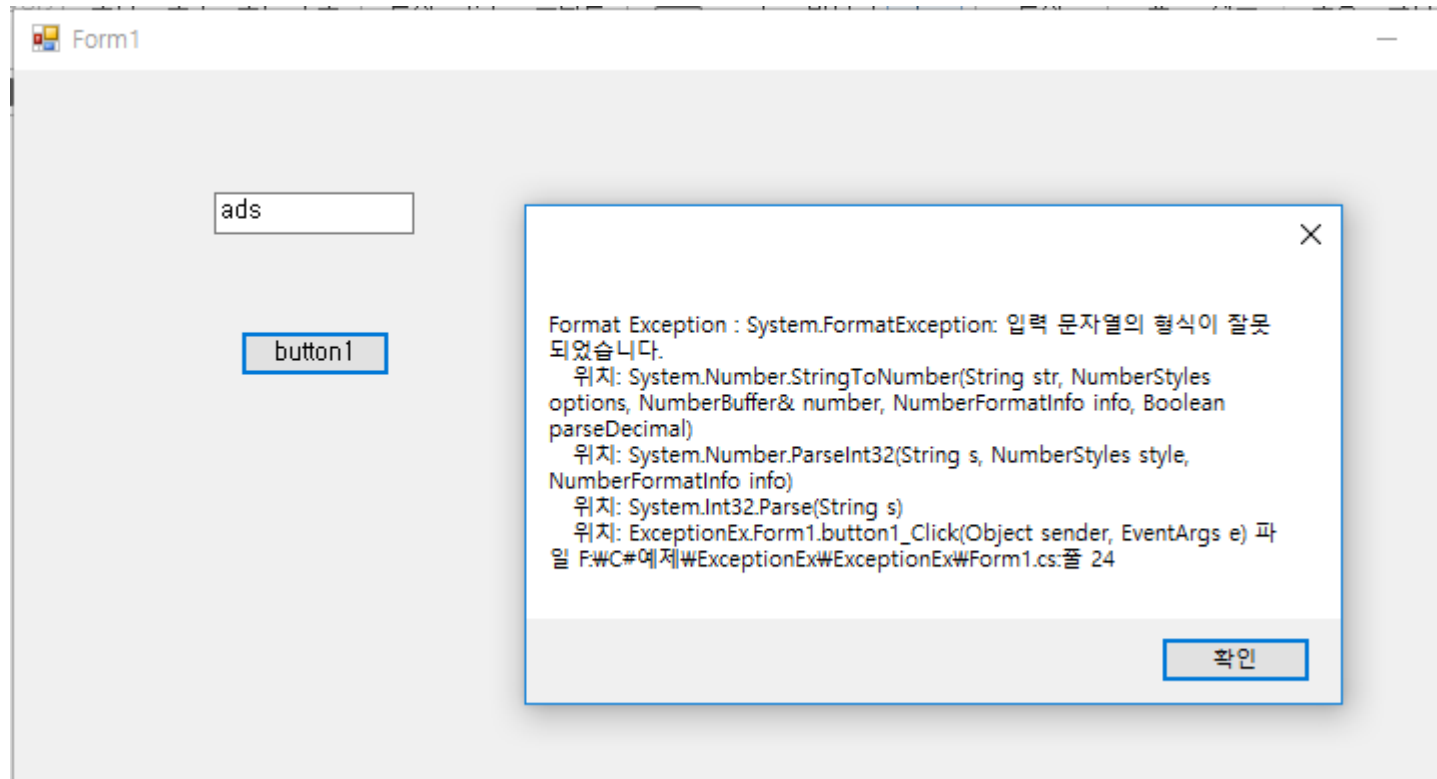
```
try
{
}
catch (Exception exception)
{
}

try
{
}
catch (Exception exception)
{
    exception.
```

Exception 클래스의 인스턴스로 예외 객체라고 부

- Data
- Equals
- GetBaseException
- GetHashCode
- GetObjectData
- GetType
- HelpLink
- HResult
- InnerException

- 예외 객체에서 정보 추출

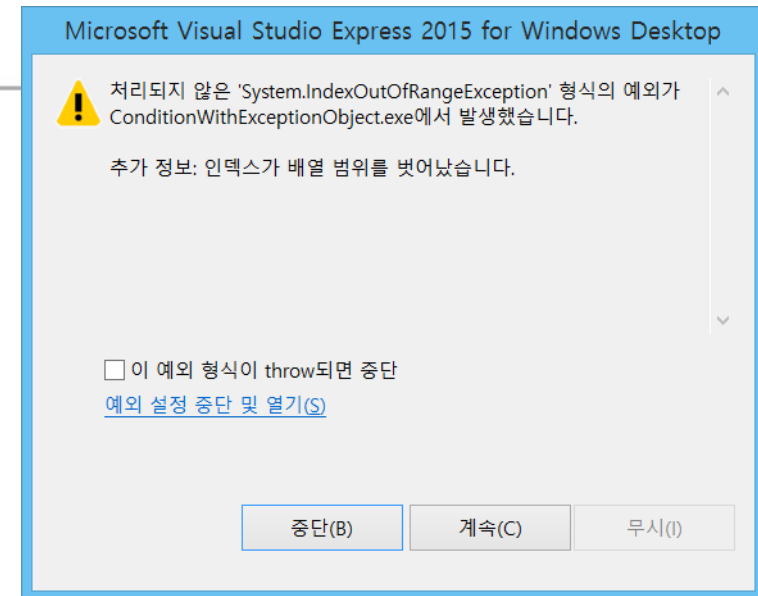


- 예외 객체를 사용한 예외 구분
  - 예외 상황 확인하기

코드 10-8 예외 상황 확인

```
static void Main(string[] args)
{
    Console.Write("입력: ");
    string input = Console.ReadLine();
    int[] array = { 52, 273, 32, 103 };

    int index = int.Parse(input);
    Console.WriteLine("입력 숫자: " + index);
    Console.WriteLine("배열 요소: " + array[index]);
}
```





- 고급 예외 처리하기

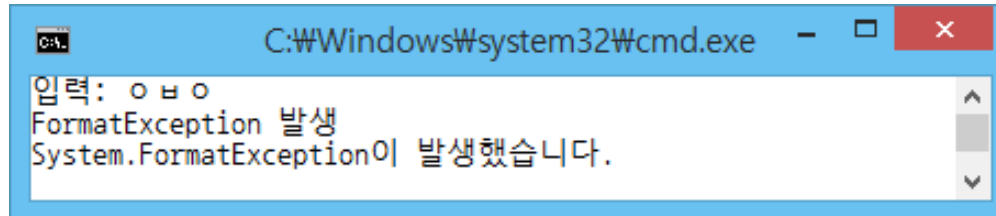
코드 10-9 고급 예외 처리

```
static void Main(string[] args)
{
    Console.Write("입력: ");
    try
    {
        string input = Console.ReadLine();
        int[] array = { 52, 273, 32, 103 };

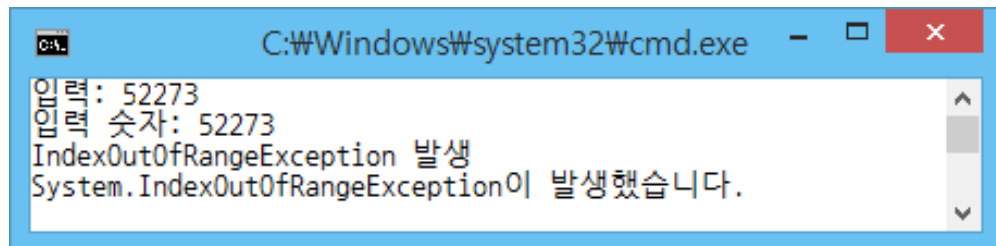
        int index = int.Parse(input);
        Console.WriteLine("입력 숫자: " + index);
        Console.WriteLine("배열 요소: " + array[index]);
    }
    catch (FormatException exception)
    {
```

# 예외 객체를 사용한 예외 구분

```
        Console.WriteLine("FormatException 발생");  
        Console.WriteLine(exception.GetType() + "이 발생했습니다.");  
    }  
    catch (IndexOutOfRangeException exception)  
    {  
        Console.WriteLine("IndexOutOfRangeException 발생");  
        Console.WriteLine(exception.GetType() + "이 발생했습니다.");  
    }  
}
```



```
C:\Windows\system32\cmd.exe  
입력: ㅇㅂㅇ  
FormatException 발생  
System.FormatException이 발생했습니다.
```



```
C:\Windows\system32\cmd.exe  
입력: 52273  
입력 숫자: 52273  
IndexOutOfRangeException 발생  
System.IndexOutOfRangeException이 발생했습니다.
```

- catch 구문과 var 키워드
  - catch 구문의 괄호 안에는 var 키워드 사용 불가(오류 발생)

```
try
{

}
catch (var exception)
{

}
```

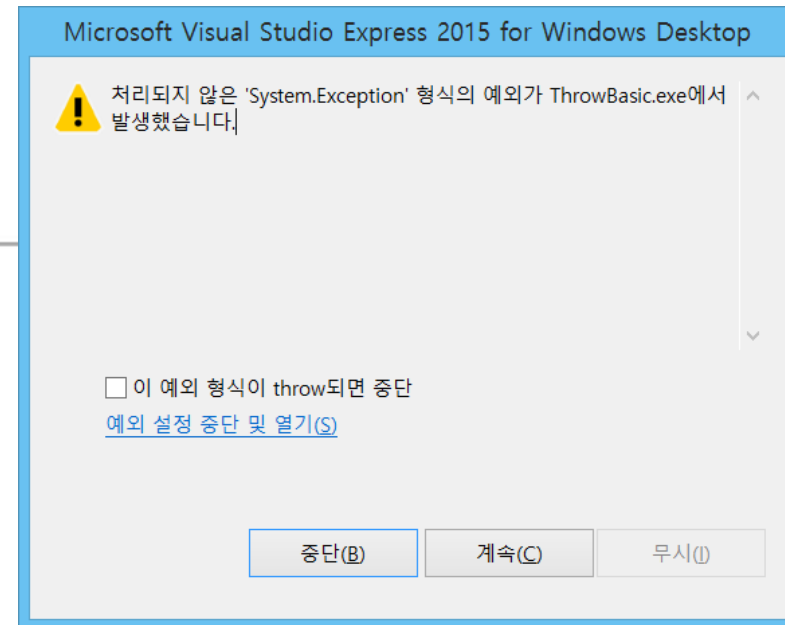
- 예외 강제 발생 방법

```
throw new Exception();
```

- 예외 던지기
  - 예외 던지기 확인하기

코드 10-10 예외 던지기 확인

```
class Program
{
    static void Main(string[] args)
    {
        throw new Exception();
    }
}
```



- 강제로 던진 예외의 예외 처리하기

코드 10-11 강제로 던진 예외의 예외 처리

```
class Program
{
    static void Main(string[] args)
    {
        try
        {
            throw new Exception();
        }
        catch (Exception exception)
        {
            Console.WriteLine("예외가 발생했습니다.");
        }
    }
}
```

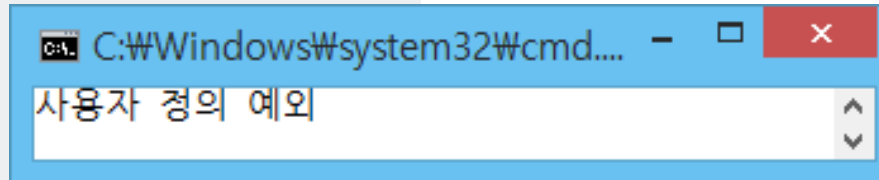
- 사용자 정의 예외

코드 10-13 사용자 정의 예외

/10장/Exceptions

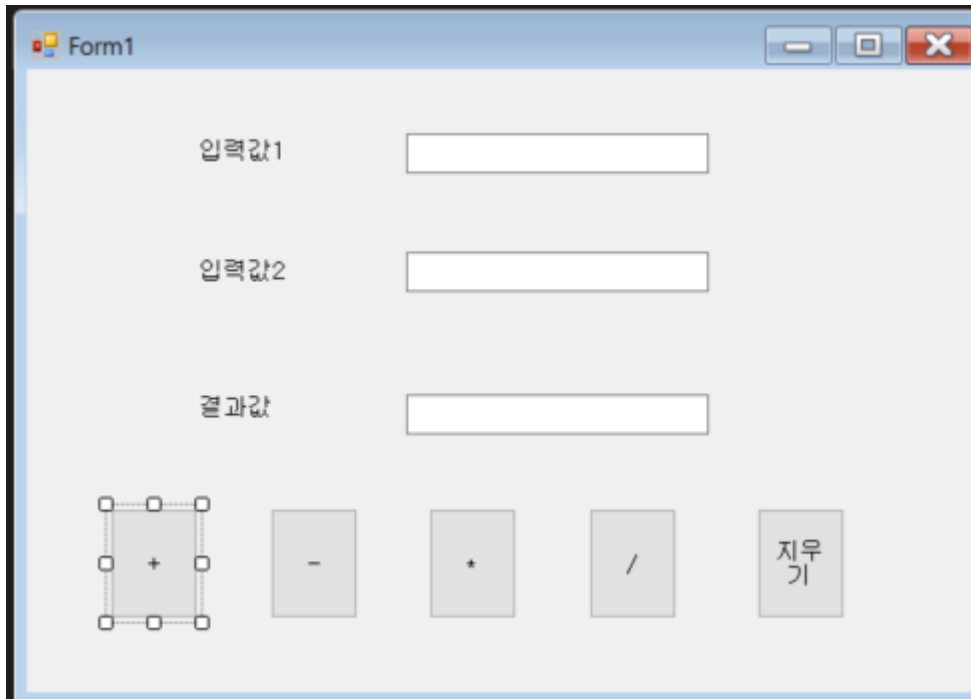
```
class CustomException : Exception
{
    public CustomException(string message) : base(message)
    {
    }
}

class Program
{
    static void Main(string[] args)
    {
        try
        {
            throw new CustomException("사용자 정의 예외");
        }
        catch (CustomException exception)
        {
            Console.WriteLine(exception.Message);
        }
    }
}
```



- 실습 – 계산기 예외 처리

## 1) 윈도우 폼 디자인



## 2) 코드 작성

```
public partial class Form1 : Form
{
    int op;
    public Form1()
    {
        InitializeComponent();
    }

    private void button1_Click_1(object sender, EventArgs e)
    {
        op = 1;
        calc();
    }

    private void button2_Click(object sender, EventArgs e)
    {
        op = 2;
        calc();
    }

    private void button3_Click(object sender, EventArgs e)
    {
        op = 3;
        calc();
    }
}
```

```
private void button4_Click(object sender, EventArgs e)
{
    op = 4;
    calc();
}

private void button5_Click(object sender, EventArgs e)
{
    textBox1.Text = "";
    textBox2.Text = "";
    textBox3.Text = "";
    textBox1.Focus();
}
```



## 2 ) 코드 작성

```
public void calc()
{
    try
    {
        int num1 = int.Parse(textBox1.Text);
        int num2 = int.Parse(textBox2.Text);
        int result = 0;
        switch (op)
        {
            case 1:
                result = num1 + num2;
                break;
            case 2:
                result = num1 - num2;
                break;
            case 3:
                result = num1 * num2;
                break;
            case 4:
                result = num1 / num2;
                break;
        }
        textBox3.Text = result.ToString();
    } catch (Exception e)
    {
        MessageBox.Show("에러가 발생했습니다.\n" + e.ToString());
    }
}
```

## 3 ) 결과 확인

