

Day 12 Assignment
By
Triveni Anumolu
08-02-2022

1. What is exception handling and why we need exception handling.

Exception handling is a process of handling runtime errors.

Exception handling is done to handle the errors gracefully so that the application will not crash and will not display any errors to the end user.

2. Write a simple division program and handle three exceptions discussed in the class., also add super exception at the last.

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day12Project1
{
    /**
    Author: Triveni Anumolu
    Purpose: To handle exceptions using try and catch block
    */
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
                int a, b, c;
                Console.WriteLine("Enter a value");
                a = Convert.ToInt32(Console.ReadLine());
                Console.WriteLine("Enter b value");
                b = Convert.ToInt32(Console.ReadLine());
                c = a / b;
                Console.WriteLine(c);
            }
        }
    }
}
```

```

        catch(DivideByZeroException ex)
        {
            Console.WriteLine("Denomintor shouldn't be zero. Enter a valid input");
        }
        catch(FormatException ex)
        {
            Console.WriteLine("Format is not correct.Enter a numerical value");
        }
        catch(OverflowException ex)
        {
            Console.WriteLine("Given input is out of range. Enter a value between 1 to
1000000");
        }
        catch(Exception ex)
        {
            Console.WriteLine("Some error occured. Contact admin@abc.com");
        }
        Console.ReadLine();
    }
}

```

Result:

Enter a value

1

Enter b value

2

0

Enter a value

1243

Enter b value

0

Denomintor shouldn't be zero. Enter a valid input

Enter a value

52@\$

Format is not correct.Enter a numerical value

Enter a value

24

Enter b value

1342354653747487

Given input is out of range. Enter a value between 1 to 1000000

4.What is the use of "finally" block illustrate with an example.

Statements inside finally block gets executed irrespective of occurrence of exception.

Example:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day12Project2
{
    class Program
    {
        /*****
        Author:Triveni Anumolu
        Purpose: handling exceptions and using finally block along with try and catch blocks
        *****/
        static void Main(string[] args)
        {
            try
            {
                int a, b, c;
                Console.WriteLine("Enter a value");
                a = Convert.ToInt32(Console.ReadLine());
                Console.WriteLine("Enter b value");
                b = Convert.ToInt32(Console.ReadLine());
                c = a / b;
                Console.WriteLine(c);
            }
            catch (DivideByZeroException ex)
            {
                Console.WriteLine("Denomintor shouldn't be zero. Enter a valid input");
            }
            catch (FormatException ex)
            {
                Console.WriteLine("Format is not correct.Enter a numerical value");
            }
            catch (OverflowException ex)
            {
                Console.WriteLine("Given input is out of range. Enter a value between 1 to 1000000");
            }
            catch (Exception ex)
            {
                Console.WriteLine("Some error occured. Contact@abc.gmail.com");
            }
        }
    }
}
```

```

    }

    finally
    {
        Console.WriteLine("\n\n\n\nDesigned by abc");
    }
    Console.ReadLine();

}
}
}

```

Result:



D:\DotnetProjects\Day12Assign

Enter a value

23

Enter b value

12

1

Designed by abc

5. Write the 5 points explained about exception handling.

- i. Exception handling is done to handle the errors gracefully so that the application will not crash and to not display any errors to the end user.
- ii. A single try block can have multiple catch blocks.
- iii. Always remember to write general exceptions at the end.
- iv. Statements which are in finally block get executed irrespective of occurrence of exception.
- v. The general syntax for writing exception is try block, catch block and finally block.

6. What is compilation and Runtime error. Write at least 3 differences between them.

Compilation Error:

Compilation error occurs when the syntax of the programming language is not followed.

Runtime Error:

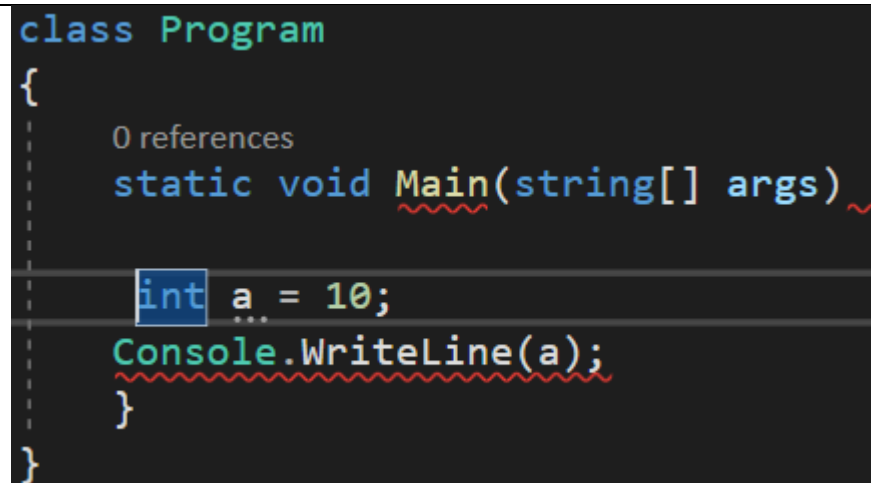
Runtime errors refer to the error encountered during the execution of code at runtime.

Compilation Error	Runtime Error
<ul style="list-style-type: none"> These errors are detected by compiler. 	<ul style="list-style-type: none"> These errors are not detected by the compiler.
<ul style="list-style-type: none"> They prevent the code from running as it detects some syntax errors. 	<ul style="list-style-type: none"> They prevent the code from complete execution.
<ul style="list-style-type: none"> Fixing an error at compilation stage is easy. 	<ul style="list-style-type: none"> Fixing an error requires going back to code.

7. Write any 6 compilation errors with small code snippet.
Add compilation error screen shots.

1.Missing Parenthesis:

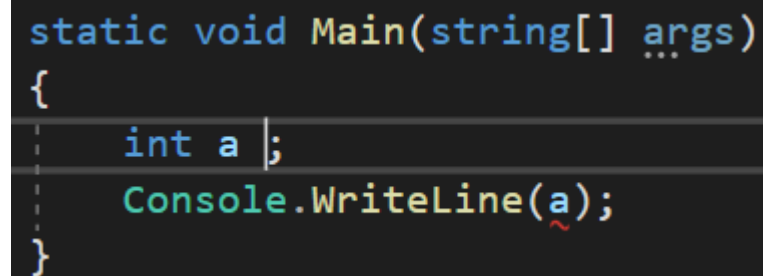
```
int a=10;
Console.WriteLine(a);
}
```



```
class Program
{
    0 references
    static void Main(string[] args) ~
    {
        int a = 10;
        Console.WriteLine(a);
    }
}
```

2.Not initializing variable:

```
{
    int a;
    Console.WriteLine();
}
```



```
static void Main(string[] args)
{
    int a ;
    Console.WriteLine(a);
}
```

3.When we give spelling mistakes or if there is a case difference:

```
{
    int a=10;
    console.WriteLine();
}
```

```
static void Main(string[] args)
{
    int a ;
    console.WriteLine(a);
}
```

4. When namespace is not imported:

```
{
    int a=10;
    Console.WriteLine(a);
}
```

```
//using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp3
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            int a ;
            console.WriteLine(a);
        }
    }
}
```

5. when semicolon is removed:

```
{
    int a=10;
    Console.WriteLine(a)
```

```

}

static void Main(string[] args)
{
    int a=10 ;
    Console.WriteLine(a)
}

```

6. When the variable is not declared:

```

{
    Console.WriteLine("Enter a value");
    b=Console.ReadLine();
}

```

```

static void Main(string[] args)
{
    Console.WriteLine("Enter a value");
    b = Console.ReadLine();
}

```

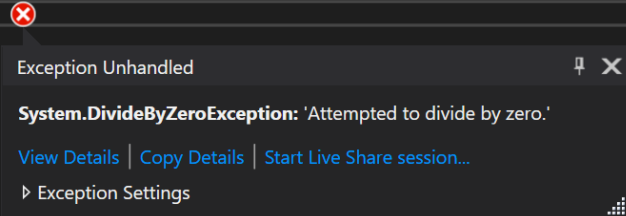
8. Write any 6 runtime errors with small code snippets and add run time error screen shots.

1. {
 int a=10, b=0;
 Console.WriteLine(a/b);
 }

```

static void Main(string[] args)
{
    int a = 20, b=0;
    Console.WriteLine(a/b);
}

```



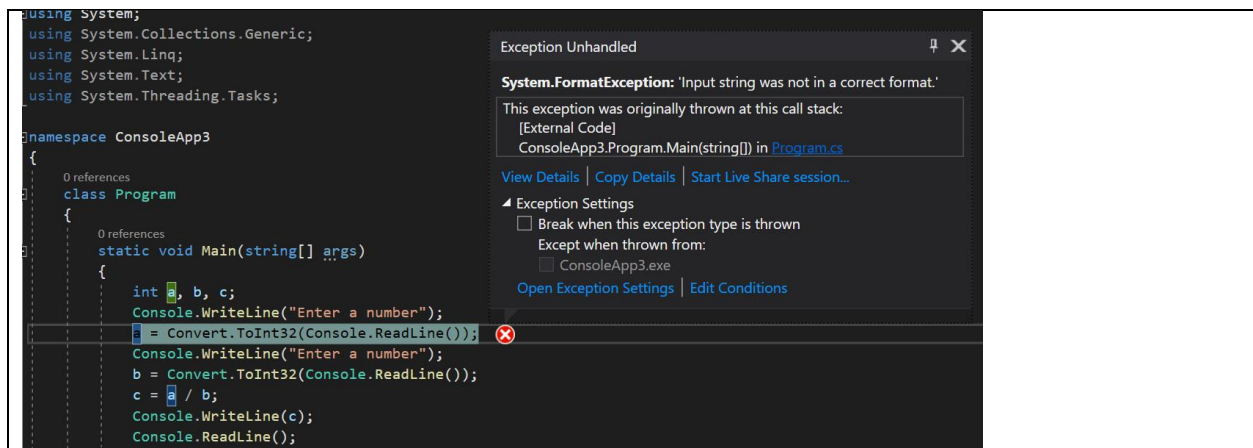
Exception Unhandled

System.DivideByZeroException: 'Attempted to divide by zero.'

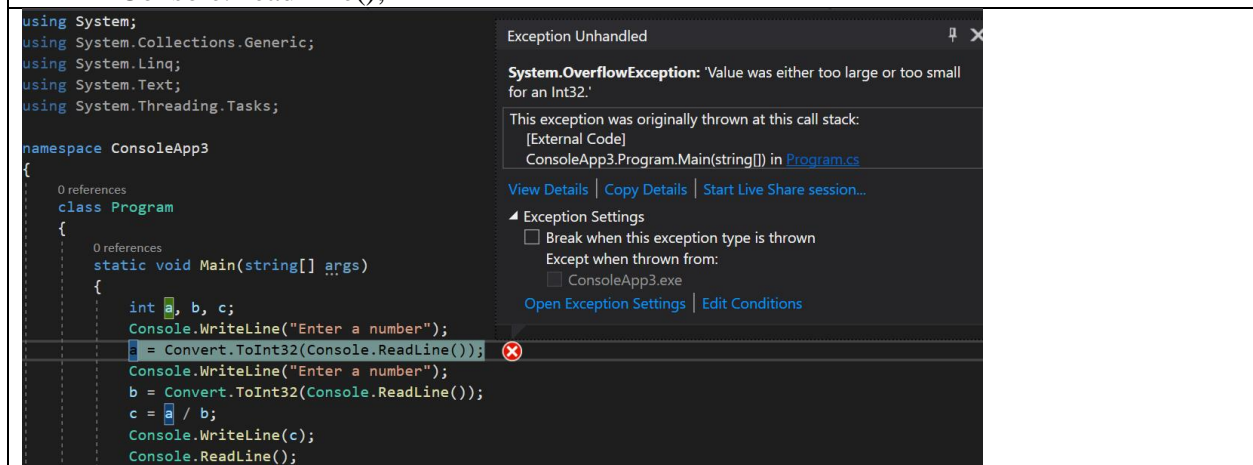
[View Details](#) | [Copy Details](#) | [Start Live Share session...](#)

▸ Exception Settings

2. int a, b, c;
 Console.WriteLine("Enter a number");
 a = Convert.ToInt32(Console.ReadLine());
 Console.WriteLine("Enter a number");
 b = Convert.ToInt32(Console.ReadLine());
 c = a / b;
 Console.WriteLine(c);
 Console.ReadLine();



3. `int a, b, c;`
`Console.WriteLine("Enter a number");`
`a = Convert.ToInt32(Console.ReadLine());`
`Console.WriteLine("Enter a number");`
`b = Convert.ToInt32(Console.ReadLine());`
`c = a / b;`
`Console.WriteLine(c);`
`Console.ReadLine();`



4. `int[] data = new int[5];`
`data[6]=11;`

