DAY12 ASSIGNMENT
BY
PAVAN KUMAR (08-02-2022)

Q1). What is Exception Handling and why we need exception handling

EXCEPTION HANDLING:

→ EXCEPTION HANDLING is done to ensure that our application will not crash.
→ Exception Handling will not display any technical details to make sure that we handle errors gracefully and display friendly messages.
→ C# exception handling is built upon four keywords: **try, catch and finally.**
→ There are few types of methods in Exception Handling. They are

a. **OVERFLOWEXCEPTION METHOD.**
b. **DIVIDEBYZERO EXCEPTION METHOD.**
c. **FORMAT EXCEPTION METHOD**
d. **GENERAL EXCEPTION METHOD**

```
try
{   // statements causing exception}
catch( ExceptionName e1)
 {
 catch( ExceptionName e2)
{   finally
{   // statements to be executed}
```

Q2). Write a simple division program and handle three exceptions discussed in the class., also add super exception at the last.

CODE:

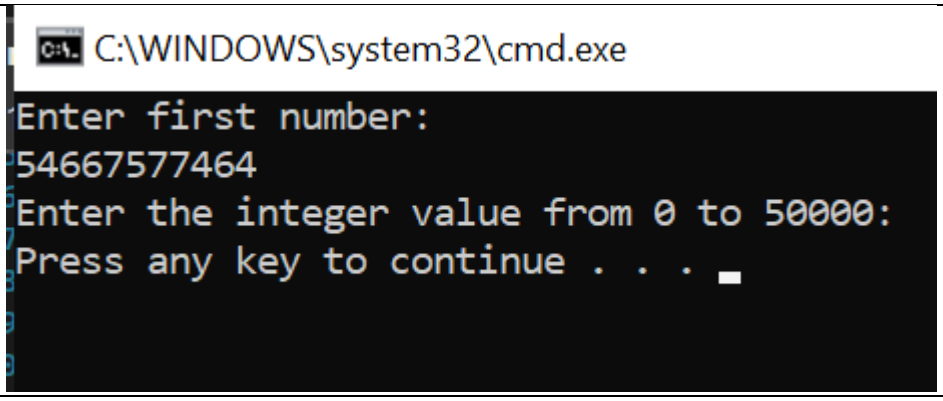
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Day12Project1
{
   /// <summary>
   /// DONE BY: PAVAN
   /// PURPOSE: A SIMPLE DIVISION PROGRAM USING EXCEPTION HANDLING:
   /// </summary>
   internal class Program
   {
      static void Main(string[] args)
      {
```

```
        try
        {
           int a, b, c;
           Console.WriteLine("Enter first number:");
           a = Convert.ToInt32(Console.ReadLine());
           Console.WriteLine("Enter second number:");
           b = Convert.ToInt32(Console.ReadLine());
           c = a / b;
           Console.WriteLine("{0}", c);
           Console.ReadLine();
        }
        //USING OVERFLOW EXCEPTION//
        catch (OverflowException)
        {
           Console.WriteLine("Enter the integer value from 0 to 50000: ");
        }
        //USING DIVIDE BY ZERO EXCEPTION//
        catch(DivideByZeroException)
        {
           Console.WriteLine("Enter the integer value greater than Zero: ");
        }
        //USING FORMAT EXCEPTION
        catch(FormatException)
        {
           Console.WriteLine(" Please Enter only integer value");
        }
        //USING GENERAL EXCEPTION//
        catch (Exception)
        {
           Console.WriteLine("Unable to display Output please contact admin@nbht.com");
           Console.ReadLine();
        }
     }

   }
}
```

OUTPUT:



Q3). Research and write at least 6 exceptions that occur in C# with sample code.

A) NULLREFERENCE EXCEPTION:

Handles errors generated from referencing a null object.

```
using System;
using System.Collections.Generic;
namespace RunTimeProjects
{
    class check
    {
      static void Main(string[] args)
       {
         string value = null;
         if(value.Length == 0)
         {
            Console.WriteLine(value);
         }
       }
    }
}
```

```
Unhandled Exception: System.NullReferenceException: Object reference not set to an instance of an object.
   at RunTimeProjects.check.Main(String[] args) in C:\HTML\ConsoleApp1\ConsoleApp1\Program.cs:line 13
```

InvalidCastException:

Handles the error generated by invalid typecasting.

System.IO.IOException:

Handles the Input Output errors.

System.FieldAccessException

Handles the error generated by invalid private or protected field access

Q4). What is the use of "finally" block illustrate with an example.

CODE:

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

namespace Day12Project2
{
   /// <summary>
   /// DONE BY: PAVAN//
```

```
    /// PURPOSE: Use of "finally" block illustrate with an example.
    /// </summary>
    internal class Program
    {
        /// <summary>
        /// USING DIVIDEBYZERO EXCEPTION METHOD//
        /// </summary>
        /// <param name="args"></param>
        static void Main(string[] args)
        {
            int a, b, c, d, e;
            Console.WriteLine("Enter first number:");
            a = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter second number:");
            b = Convert.ToInt32(Console.ReadLine());
            try
            {
                c = a + b;
                Console.WriteLine($"Addition of {a} and {b}, {c}");
                d = a / b;
                Console.WriteLine($"Div of {a} and {b}, {d}");
            }
            catch(DivideByZeroException)
            {
                Console.WriteLine("The second number should not be zero:");
            }
            finally
            {
                e = a * b;
                Console.WriteLine($"Mul of {a} and {b}, {e}");
                Console.ReadLine();
            }
        }
    }
}
```

OUTPUT:

```
Enter first number:
46
Enter second number:
6
Addition of 46 and 6, 52
Div of 46 and 6, 7
Mul of 46 and 6,276
```

Q5). Write the 5 points I explained about exception handling.

- Exception Handling is done to ensure that our application will not crash.
- In a single try block we will have multiple catch blocks.
- Statements in finally block are executed irrespective of exceptions.
- General Exceptions are always at the end of the code.
- All exceptions the derived from System.
- Exception handling is built upon four keywords: **try, catch and finally.**

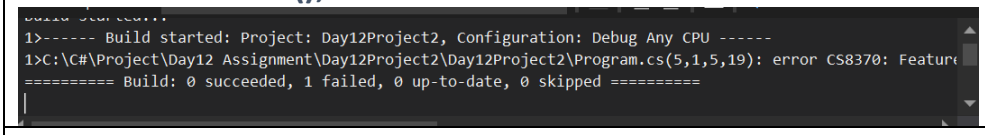
Q6). What is compilation and Runtime error? Write at least 3 differences between them

COMPILATION ERROR	RUN TIME ERROR
1) Errors that occur when you violate the rules of writing syntax are known as Compile-Time errors.	1) Errors which occur during program execution(run-time) after successful compilation are called run-time errors.
2) These errors are referenced to errors in syntax or semantics.	2) These errors are a reference to the execution of the code in a runtime environment.
3). These types of errors can be rectified easily and fixed out.	3) These errors cannot be rectified easily.

Q7). Write any 6 compilation errors with a small code snippet. Add compilation error screen shots.

1. **USING OF UN-ASSIGNED VALUES:**

```
using System;
using System.Collections.Generic;
 int a = 10, b = 5;
Console.WriteLine("enter any number");
a= Convert.ToInt32(Console.ReadLine());
b= Convert.ToInt32(Console.ReadLine());
sum = a + b + c;
Console.WriteLine(sum);
Console.ReadLine();
```



2. **SPELLING MISTAKES:**

```
using System;
using System.Collections.Generic;
int a = 10, b = 5;
```

```
Console.Writeline("enter any number");
a= Convert.ToIn32(Console.ReadLine());
b= Convert.ToInt32(Console.ReadLine());
sum = a + b + c;
Console.Writeline(sum);
Console.Readine();
```

CS1519	Invalid token '(' in class, record, struct, or interface member declaration	Day12Project2	Program.cs	9	Active
CS1031	Type expected	Day12Project2	Program.cs	9	Active
CS8124	Tuple must contain at least two elements.	Day12Project2	Program.cs	9	Active
CS1026	) expected	Day12Project2	Program.cs	9	Active

3. IMPORTING THE NAMESPACE:

Without using namespace like **System. Collections, System. Collections. Generic.**

```
int a = 10, b = 5;
Console.WriteLine("enter any number");
a= Convert.ToInt32(Console.ReadLine());
b= Convert.ToInt32(Console.ReadLine());
```

Preview Changes

System.Console

Changes

Program.cs

System.Console.WriteLine("Enter first number:");

Preview Code Changes:

```
9       {
10
11          int a, b, c, d, e;
12
13          System.Console.WriteLine("Enter first number:
14
15          a = Convert.ToInt32(Console.ReadLine());
16
17          Console.WriteLine("Enter second number:");
18
```

Ln: 19 Ch: 50

Apply Cancel

4. Mis-using Methods:

```
using System;
namespace Day12Project2
{

   internal class Program
```

```
{
    static void Main(string[] args)
    {
        int a = 16;
        Console.WriteLine("Enter any number:");
        a= Convert.ToInt32(Console.ReadLine()); e
        int squarerrot = Math.squareroot(a);
```

CS0117: 'Math' does not contain a definition for 'squareroot'

Q8). Write any 6 runtime errors with small code snippets and add run time error screen shots.

1. **ZERO RUNTIME ERROR:**

```
using System;

using System.Collections.Generic;
namespace RunTimeProject
{
   internal class Program

   {
      static void Main(string[] args)

      {
         int a, b, c;
         Console.WriteLine("enter any number:");
         a = Convert.ToInt32(Console.ReadLine());
         Console.WriteLine("enter another number");
         b = Convert.ToInt32(Console.ReadLine());
         c = a / b;
         Console.WriteLine(" Div of {a} / {b} is {c}");
      }
   }
}
```

```
enter any number:
234
enter another number
0

Unhandled Exception: System.DivideByZeroException: Attempted to divide by zero.
   at RunTimeProjects.Program.Main(String[] args) in C:\HTML\ConsoleApp1\ConsoleApp1\Program.cs:line 17
Press any key to continue . . .
```

2. NULL REFERENCE ERROR:

```
using System;
using System.Collections.Generic;
namespace RunTimeProjects
{
    class check
    {
      static void Main(string[] args)
       {
         string value = null;
         if(value.Length == 0)
         {
            Console.WriteLine(value);
         }

      }
   }
}
```

```
Unhandled Exception: System.NullReferenceException: Object reference not set to an instance of an object.
   at RunTimeProjects.check.Main(String[] args) in C:\HTML\ConsoleApp1\ConsoleApp1\Program.cs:line 13
```

3. ARRAY MIS-MATCH:

```
using System;

using System.Collections.Generic;
namespace RunTimeProjects
{
    class check

   {
      static void Main(string[] args)

      {
         string[] arr1 = {"HELLO"};
```

```
        object[] arr2 = arr1;
        arr2[0] = 2;



      }
   }
}
```

```
Unhandled Exception: System.ArrayTypeMismatchException: Attempted to access an element as a type incompatible with the a
rray.
   at RunTimeProjects.check.Main(String[] args) in C:\HTML\ConsoleApp1\ConsoleApp1\Program.cs:line 14
```

4. STACK OVERFLOW:

```
using System;

using System.Collections.Generic;

public class check

{
   static void Recurse(int val)

   {
      Console.WriteLine(val);
      Recurse(++val);
   }
   public static void Main()
   {
      Recurse(0);
   }
      }
```

```
14334
14334
14334
14335

Process is terminated due to StackOverflowException.
```