|  |  |
| --- | --- |
| **Ex. No. 07** | **APPLICATION DEVELOPMENT USING EXCEPTION HANDLING** |
| **07.10.2023** |

**Aim**

To develop C# console application using Exception Handling statements.

**Description**

**Exception Handling:**

Catching and recording the errors or bugs such that those can be fixed later mostly they can be logged in databases.

Blocks/Statements of Exception:

* try: Statements that causes exceptions will be included here
* catch: Statements that has to be performed when an exception is raised
* finally: Statements will be executed whether any exception is raised or not.
* throw: To manually throw an exception

Syntax:

class UserDefinedException:Exception{

public UserDefinedException(string msg):base(msg){}

}

class program{

try{ //try block of statements}

catch (Exception ex){/catch block of statements}

finally{//finally block of statements}

}

**Source Code**

**A 1.**

using System;

using System.Threading.Tasks;

namespace Ex7{

internal class SetA1{

static void Main(string[] args){

try{

Console.Write("Enter Numerator: ");

int num=Convert.ToInt32(Console.ReadLine());

Console.Write("Enter Denominator: ");

int den = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Result: " + num / den);}

catch (FormatException){

Console.WriteLine("System.ArgumentException");}

catch (DivideByZeroException){

Console.WriteLine("System.DivideByZeroException: / by zero");}

Console.ReadKey();}}}

**A 2.**

using System;

using System.Threading.Tasks;

namespace Ex7{

class MyCalculator{

public long power(int n, int p){

if (n < 0 || p < 0) throw new Exception("System.Exception: n or p should not be negative");

if (n == 0 && p==0) throw new Exception("System.Exception: n and p should not be zero");

return (long) Math.Pow(n, p);}}

internal class SetA2{

static void Main(string[] args){

Console.Write("Enter n value: ");

int n=Convert.ToInt32(Console.ReadLine());

Console.Write("Enter p value: ");

int p = Convert.ToInt32(Console.ReadLine());

MyCalculator mycal=new MyCalculator();

try{

long result=mycal.power(n, p);

Console.WriteLine("Result: " + result);}

catch (Exception ex){

Console.WriteLine(ex.Message);}

Console.ReadKey();}}}

**B.**

using System;

using System.Threading.Tasks;

namespace Ex7{

class InvalidEmpidException : Exception{

public InvalidEmpidException(string msg) : base(msg) { }}

class InvalidNameException : Exception{

public InvalidNameException(string msg) : base(msg) { }}

class InvalidAgeException:Exception{

public InvalidAgeException(string msg) : base(msg) { }}

class Employee{

string empid,name;

int age;

public Employee(string empid, string name, int age){

if (empid.Length < 4) throw new InvalidEmpidException("Length of the Empid should be greater than 4");

if (int.TryParse(name, out int result)) throw new InvalidNameException("Name Should not be a number");

if (age > 50) throw new InvalidAgeException("Age should not be less than or equal to 50");

this.empid = empid;

this.name = name;

this.age = age;}}

internal class SetB{

static void Main(string[] args){

Console.Write("Enter Employee Id: ");

string eid=Console.ReadLine();

Console.Write("Enter Employee Name: ");

string ename = Console.ReadLine();

Console.Write("Enter Employee Age: ");

int age = Convert.ToInt32(Console.ReadLine());

try{

Employee emp1 = new Employee(eid, ename, age);

Console.WriteLine("Employee Object Created Successfully");}

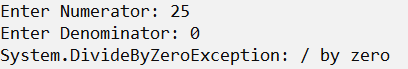
catch (Exception ex){

Console.WriteLine(ex.Message);}

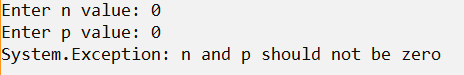
Console.ReadKey();}}}

**Output**

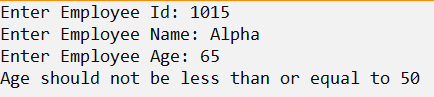
**A 1.**

****

**A 2.**

****

**B.**

****

**Result**

The C# console application using Exception Handling statements has been executed successfully and the desired output is displayed on the screen.