**Assignment\_5**

Q1:

internal class Q1

    {

        static void Main(string[] args)

        {

           Console.WriteLine("Enter num1:");

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

           Console.WriteLine("Enter num2:");

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

           try

           {

               int div = x / y;

               Console.WriteLine("Division: " + div);

           }

           catch (DivideByZeroException ex)

           {

               Console.WriteLine("Division by zero is not allowed.");

           }

           finally

           {

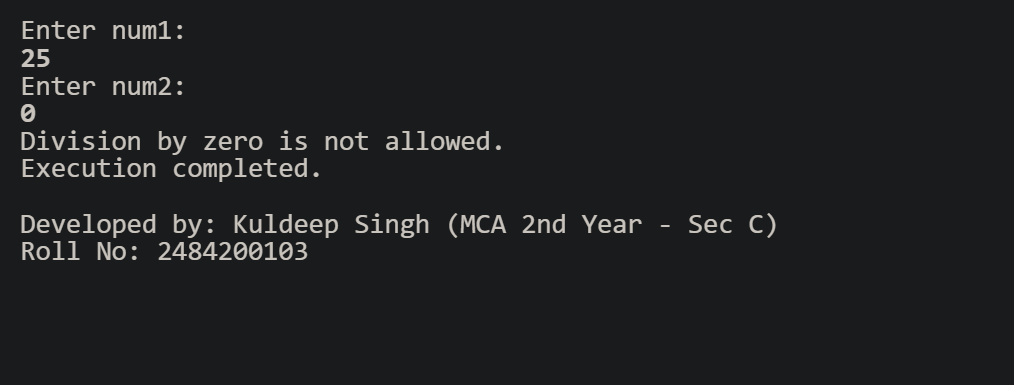
               Console.WriteLine("Execution completed.");

               Console.WriteLine("\nDeveloped by: Kuldeep Singh (MCA 2nd Year - Sec C)\nRoll No: 2484200103");

           }

        }

    }



Q2:

internal class Q2

    {

        static void Main(string[] args)

        {

           Console.WriteLine("Enter your input:");

           String x = Console.ReadLine();

           try

           {

               int num = Convert.ToInt32(x);

           }

           catch (FormatException ex)

           {

               Console.WriteLine("Input string is not in a correct format.");

           }

           catch (OverflowException ex)

           {

               Console.WriteLine("The number is too large or too small for an Int32.");

           }

           catch (Exception ex)

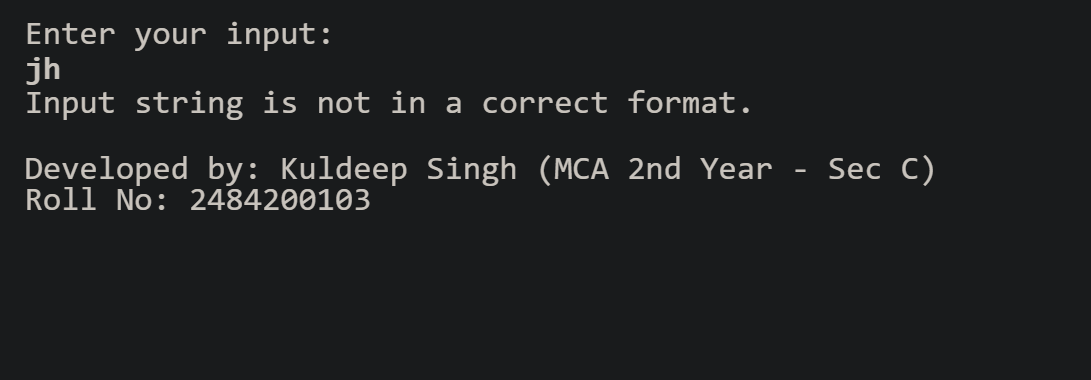
           {

               Console.WriteLine("An unexpected error occurred: " + ex.Message);

           }

           Console.WriteLine("\nDeveloped by: Kuldeep Singh (MCA 2nd Year - Sec C)\nRoll No: 2484200103");

        }

    }

Q3:

internal class Q3

    {

        class NegativeSalaryException : Exception

        {

           public NegativeSalaryException(string message) : base(message) { }

        }

        static void Main(string[] args)

        {

           try

           {

               Console.WriteLine("Enter your salary:");

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

               if (salary < 0)

               {

                   throw new NegativeSalaryException("Salary cannot be negative.");

               }

               else

               {

                   Console.WriteLine("Your salary is: " + salary);

               }

           }

           catch (NegativeSalaryException ex)

           {

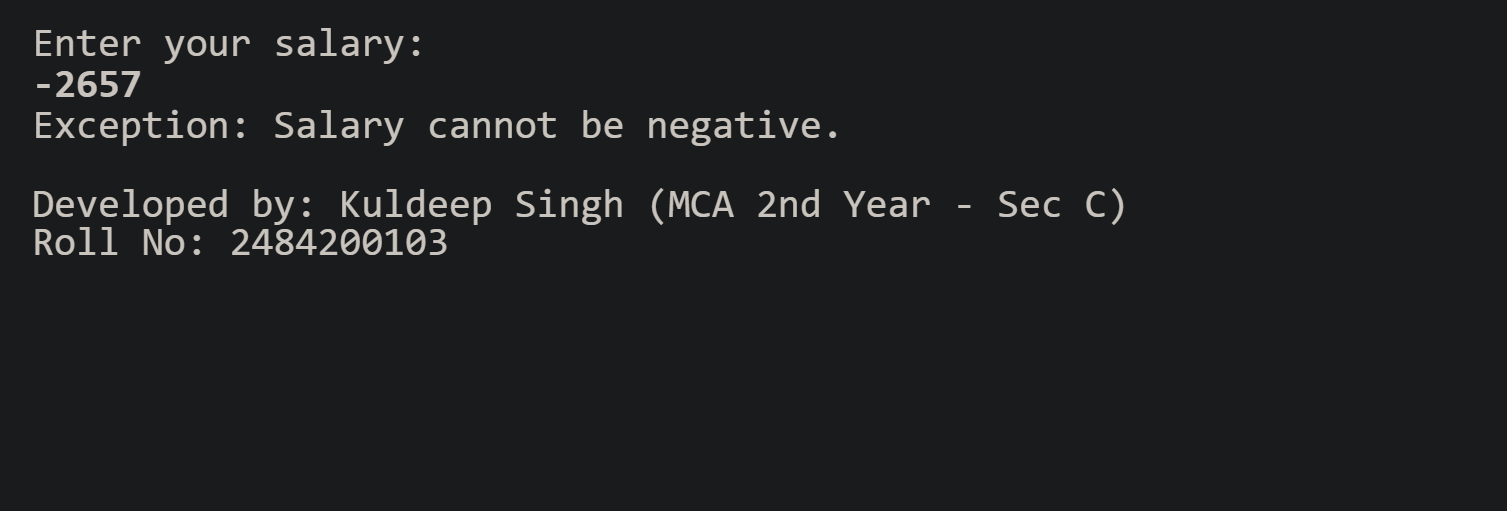
               Console.WriteLine("Exception: " + ex.Message);

           }

           Console.WriteLine("\nDeveloped by: Kuldeep Singh (MCA 2nd Year - Sec C)\nRoll No: 2484200103");

        }

    }



Q4:

internal class Q4

    {

        class InsufficientBalanceException : Exception

        {

           public InsufficientBalanceException(string message) : base(message) { }

        }

        static void Main(string[] args)

        {

           int bal = 15000;

           Console.WriteLine("enter withdrawal");

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

           try

           {

               if (bal < wthd)

               {

                   throw new InsufficientBalanceException("Balance is low.");

               }

               else

               {

                   Console.WriteLine("Your current balance is: " + (bal-wthd));

               }

           }

           catch (InsufficientBalanceException ex)

           {

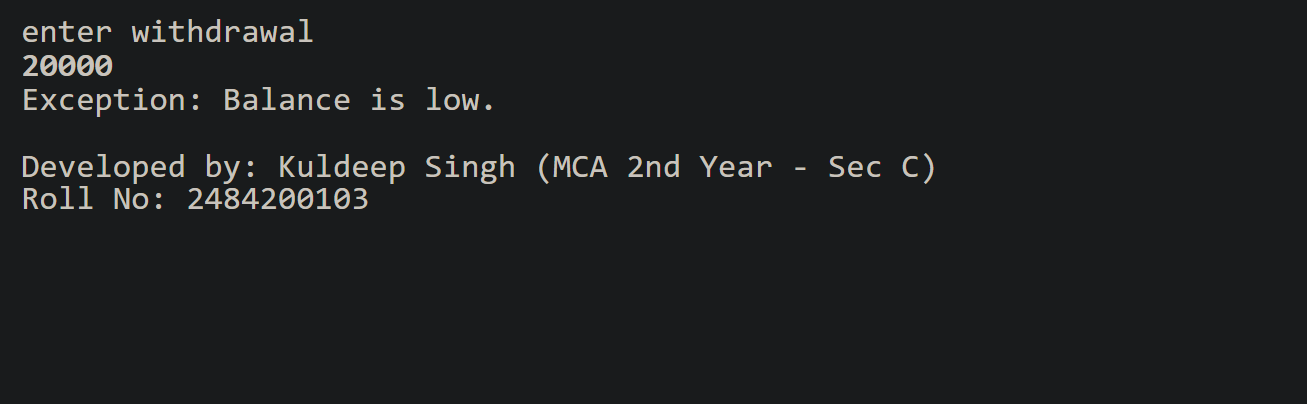
               Console.WriteLine("Exception: " + ex.Message);

           }

           Console.WriteLine("\nDeveloped by: Kuldeep Singh (MCA 2nd Year - Sec C)\nRoll No: 2484200103");

        }

    }



Q5:

internal class Q5

    {

        class InvalidMarksException : Exception

        {

            public InvalidMarksException(string message) : base(message) { }

        }

        static void Main(string[] args)

        {

            try

            {

                Console.WriteLine("Enter your marks:");

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

                if (0 > marks || marks > 100)

                {

                    throw new InvalidMarksException("inputed marks is out of range");

                }

                else

                {

                    Console.WriteLine("Your marks is: " + marks);

                }

            }

            catch (InvalidMarksException ex)

            {

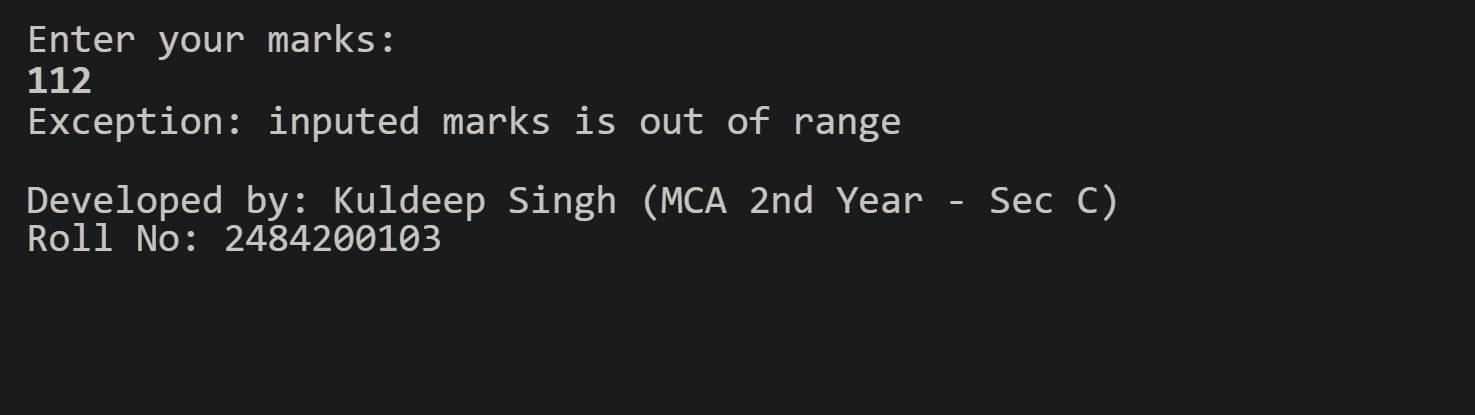
                Console.WriteLine("Exception: " + ex.Message);

            }

            Console.WriteLine("\nDeveloped by: Kuldeep Singh (MCA 2nd Year - Sec C)\nRoll No: 2484200103");

        }

    }



**MCQ Questions:**

    1. Which of the following keywords is used to handle exceptions in C#?

    C. catch

    2. What does the finally block do in C#?

    C. Executes always, whether exception occurs or not

    3. Which class is the base for all exceptions in C#?

    B. Exception

    4. What happens if an exception is not handled in any method?

    A. The program terminates abnormally

    5. Which statement is used to manually raise an exception?

    B. throw

    6. What will be the output of dividing by zero in C#?

    C. DivideByZeroException

    7. Which of the following is true about multiple catch blocks?

    B. More specific exceptions must appear before general ones

    8. Can a finally block be used without a catch block?

    B. Yes

    9. Predict the output

        using System;

        class Test

        {

            static void Main()

            {

                try

                {

                    int x = 10, y = 0;

                    int z = x / y;

                    Console.WriteLine("Result: " + z);

                }

                catch (DivideByZeroException)

                {

                    Console.Write("Division by zero not allowed |");

                }

                finally

                {

                Console.Write(" Finally block executed");

                }

            }

        }

B. Division by zero not allowed | Finally block executed

10. Which exception occurs when you access an array element beyond its limit?

A. IndexOutOfRangeException

11. What does the keyword throw; inside a catch block do?

A. Rethrows the same exception

12. Predict the output

try {

    int[] arr = { 10, 20, 30 };

    Console.WriteLine(arr[3]);

    }

    catch (DivideByZeroException){

    Console.WriteLine("Divide by zero");

    }

    catch (IndexOutOfRangeException){

    Console.WriteLine("Index error");

    }

    finally{

    Console.WriteLine("End of program");

    }

    B.

    Index error

    End of program

13. What is the use of ApplicationException class?

B. Used for user-defined exceptions

14. Predict the output

    try{

    int x = int.Parse("123A");

    Console.WriteLine("Number: " + x);

    }

    catch (FormatException){

    Console.WriteLine("Invalid number format");

    }

B. Invalid number format

15. Which block executes when an exception occurs in the try block?

C. catch

16. True or False

    In C#, every user-defined (custom) exception class must directly inherit from the

    System.Exception class or one of its derived classes.

True

17. What is exception propagation?

B. Passing an exception up the call stack until caught

18. Which block is optional in try-catch-finally structure?

D. Both B and C

19. What will happen if both try and finally blocks have return statements?

B. finally’s return overrides try’s

20. Which of the following statements about custom exceptions is correct?

A. Must inherit from Exception or ApplicationException