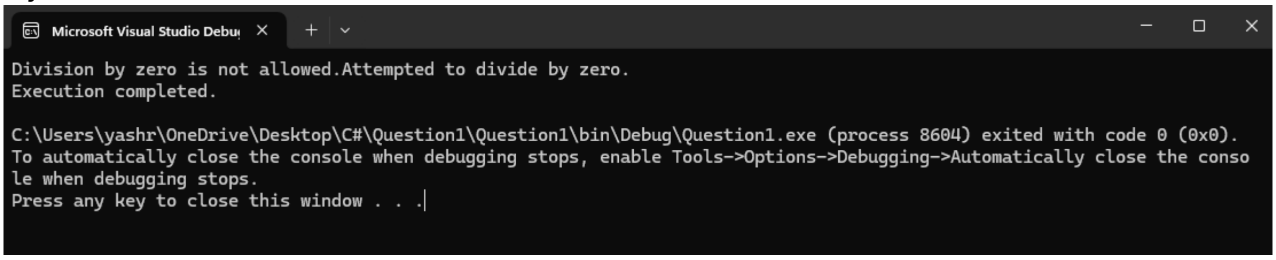
Assignment 5: Exception Handling in C#

Q1. Handling Division by Zero

Read two numbers and perform division. Use try-catch-finally. Catch DivideByZeroException and display “Division by zero is not allowed.” In the finally block display “Execution completed.” Ensure finally executes regardless of exceptions.

Solution:

```
using System;

public class NumberDivision
{
    public static void Main(string[] args)
    {
        try
        {
            int num = 10; int den = 0;
            int result = num / den; Console.WriteLine("Result: " + result);
        }
        catch (DivideByZeroException ex)
        {
            Console.WriteLine("Division by zero is not allowed." + ex.Message);
        }
        finally
        {
            Console.WriteLine("Execution completed.");
        }
    }
}
```



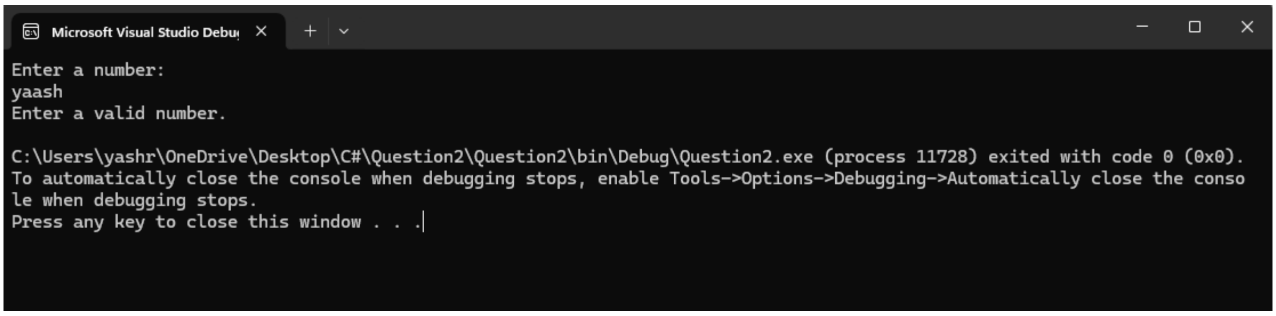
```
Division by zero is not allowed.Attempted to divide by zero.
Execution completed.

C:\Users\yashr\OneDrive\Desktop\C#\Question1\Question1\bin\Debug\Question1.exe (process 8604) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Q2. Multiple Catch Blocks

Read console input and convert to int. Handle FormatException, OverflowException, and a generic Exception, with distinct messages.

Solution:

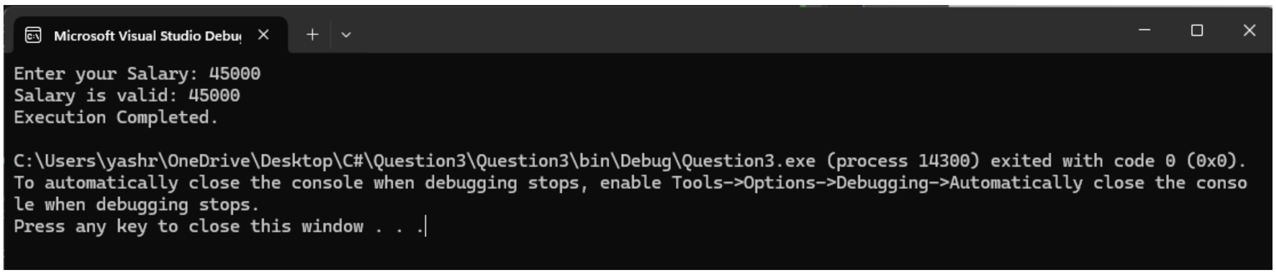
```
using System;

public class NumberFormation
{
    public static void Main(string[] args)
    {
        try
        {
            Console.WriteLine("Enter a number: ");
            int num = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("Number: " + num);
        }
        catch (FormatException ex)
        {
            Console.WriteLine("Enter a valid number.");
        }
        catch (OverflowException ex)
        {
            Console.WriteLine("Overflow Occured: " + ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine("An error Occured: " + ex.Message);
        }
    }
}
```



```
Enter a number:
yaash
Enter a valid number.

C:\Users\yashr\OneDrive\Desktop\C#\Question2\Question2\bin\Debug\Question2.exe (process 11728) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Q3. Custom Exception — NegativeSalaryException
Define NegativeSalaryException : Exception. If entered salary < 0, throw it and handle with a clear error message.

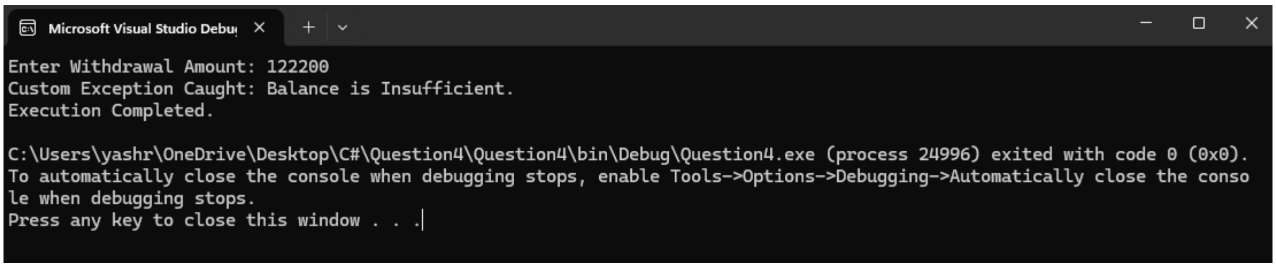
Solution:
```
using System;

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

class Program
{
    static void CheckSalary(int num)
    {
        if (num < 0)
        {
            // Throw user-defined exception
            throw new NegativeSalaryException("Salary cannot be negative!");
        }
        else
        {
            Console.WriteLine("Salary is valid: " + num);
        }
    }

    static void Main()
    {
        try
        {
            Console.Write("Enter your Salary: ");
            int n = Convert.ToInt32(Console.ReadLine()); CheckSalary(n);
        }
        catch (NegativeSalaryException ex)
        {
            Console.WriteLine("Custom Exception Caught: " + ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine("General Exception: " + ex.Message);
        }
        finally
        {
            Console.WriteLine("Execution Completed.");
        }
    }
}
```



```
Enter your Salary: 45000
Salary is valid: 45000
Execution Completed.

C:\Users\yashr\OneDrive\Desktop\C#\Question3\Question3\bin\Debug\Question3.exe (process 14300) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Q4. Banking Scenario — InsufficientBalanceException
Simulate withdrawal: if withdrawal > balance, throw custom InsufficientBalanceException; otherwise print remaining balance.

Solution:
```
using System;

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

class Program
{
    static void Balance(int withdrawal)
    {
        int balance = 2000;
        if (withdrawal > balance)
        {
            // Throw user-defined exception
            throw new InsufficientBalanceException("Balance is Insufficient.");
        }
        else
        {
            Console.WriteLine("Credited & Remaining Balance: " + (balance -
withdrawal));
        }
    }
    static void Main()
    {
        try
        {
            Console.Write("Enter Withdrawal Amount: "); int n =
Convert.ToInt32(Console.ReadLine()); Balance(n);
        }
        catch (InsufficientBalanceException ex)
        {
            Console.WriteLine("Custom Exception Caught: " + ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine("General Exception: " + ex.Message);
        }
        finally
        {
            Console.WriteLine("Execution Completed.");
        }
    }
}
```



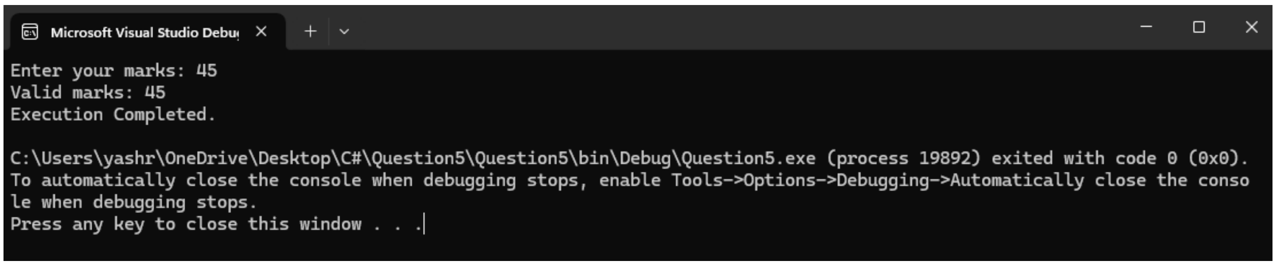
```
Enter Withdrawal Amount: 122200
Custom Exception Caught: Balance is Insufficient.
Execution Completed.

C:\Users\yashr\OneDrive\Desktop\C#\Question4\Question4\bin\Debug\Question4.exe (process 24996) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Q5. Student Marks Validation

Student class stores marks (0–100). If input outside range, throw InvalidMarksException. Demonstrate validation and handling in Main().

Solution:

```
using System;

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

class Program
{
    static void SubjectMarks(int marks)
    {
        if (marks < 0 || marks > 100)
        {
            // Throw user-defined exception
            throw new InvalidMarksException("Invalid marrks.");
        }
        else
        {
            Console.WriteLine("Valid marks: " + marks);
        }
    }

    static void Main()
    {
        try
        {
            Console.Write("Enter your marks: ");
            int n = Convert.ToInt32(Console.ReadLine()); SubjectMarks(n);
        }
        catch (InvalidMarksException ex)
        {
            Console.WriteLine("Custom Exception Caught: " + ex.Message);
        }
        catch (Exception ex)
        {
            Console.WriteLine("General Exception: " + ex.Message);
        }
        finally
        {
            Console.WriteLine("Execution Completed.");
        }
    }
}
```



```
Enter your marks: 45
Valid marks: 45
Execution Completed.

C:\Users\yashr\OneDrive\Desktop\C#\Question5\Question5\bin\Debug\Question5.exe (process 19892) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

MCQ Questions

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

A. throw B. try **C. catch** D. finally

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

A. Executes only when no exception occurs
B. Executes only when exception occurs
C. Executes always, whether exception occurs or not
D. Executes only for system exceptions

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

A. ApplicationException B. Exception C. SystemException D. RuntimeException.

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

A. The program terminates abnormally
B. The compiler throws an error
C. CLR ignores it
D. It restarts automatically

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

A. raise **B. throw** C. throws D. raiseException

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

A. Infinity
B. NaN
C. DivideByZeroException
D. ArithmeticException

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

A. The order of catch blocks does not matter
B. More specific exceptions must appear before general ones
C. Only one catch block is allowed
D. Catch blocks cannot be nested

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

A. No **B. Yes** C. Only in static methods D. Only with throw

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");
    }    }}
```

A. Result: 0
B. Division by zero not allowed | Finally block executed
C. Compile-time error
D. Program terminates abnormally

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

A. IndexOutOfRangeException
B. ArrayLimitException
C. OverflowException
D. ArgumentException

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

A. Rethrows the same exception
B. Throws a new exception
C. Terminates the program
D. Ignores the 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");
}
```

A.
Divide by zero
End of program

B.
Index error End of program

C. Only End of program **D.** Program terminated abnormally

13. What is the use of ApplicationException class?

A. Used for system exceptions
B. Used for user-defined exceptions
C. Used for compilation errors
D. Used by CLR internally

14. Predict the output

```
try{
   int x = int.Parse("123A"); Console.WriteLine("Number:
   " + x);
}
catch (FormatException){ Console.WriteLine("Invalid
   number format");
}
```

- **A.** Number: 123A
- **B. Invalid number format**
- **C.** Compile-time error
- **D.** Program terminates abnormally

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

A. try B. finally **C. catch** D. throw

Q16. 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?

A. Forwarding the exception to the next statement
B. **Passing an exception up the call stack until caught**
C. Ignoring the exception
D. Retrying code execution

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

A. try B. catch C. finally **D. Both B and C**

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

A. try's return executes
B. **finally's return overrides try's**
C. Both execute sequentially
D. Compile-time error

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

A. **Must inherit from Exception or ApplicationException**
B. Cannot include constructors
C. Cannot be thrown
D. Handled only by CLR