In this exercise you will be practicing exception handling.

Remember Exception is generated whenever your application has been placed an abnormal state. There is a huge performance penalty when an exception is thrown so you should only go into an exception state as a last option.

# You must follow the specifications exactly

In a console application, you will code six method for this lab and then call them individually from your main method. The specifications are given below.

The following method is given:

static int Division(int top, int bottom) => top / bottom;

#### Description of methods

static void DivisionNoHandling()

{

//call the Division method with argument 1 and 0

//this will terminate your application

}

static void DivisionWithExceptionHandling()

{

//call the Division method with argument 1 and 0

//You will catch the exception and do nothing

//Your application should not crash

}

static void DivisionWithExceptionHandlingMoreInfo()

{

//call the Division method with argument 1 and 0

//You will catch the exception and print the associated message

//your application should not crash

}

static void DivisionWithExceptionHandlingThrow()

{

//call the Division method with argument 1 and 0

//You will catch the exception and print the associated message

//You should create and throw a new exception

//your application will crash

}

static void GeneratingException()

{

//You should create and throw a new exception

//your application will crash

}

static void HandlingRandomException ()

{

//You should call the method below and catch all the possible exceptions individually and display a sensible output

//your application will not crash

}

static void GeneratingRandomException()

{

int exceptionType = new Random().Next() % 6;

switch(exceptionType)

{

case 0:

throw new IndexOutOfRangeException();

case 1:

throw new NullReferenceException();

case 2:

throw new InvalidOperationException();

case 3:

throw new ArithmeticException();

case 4:

throw new FormatException();

case 5:

throw new InvalidCastException();

case 6:

throw new OutOfMemoryException();

}

}