4.1 Exception handling report

NullReferenceException

This exception is thrown when you try to use something as reference, but it is not yet initialized.

This exception should be thrown by the Runtime system not by programmer, as it reflects the programmer’s error rather than the user-side error. This exception should be aware and avoided by programmer.

As this exception reflects the programmer’s error, it should include the variable that is being null, the message should be like this: “There is a null value that is being used as reference!”. No parameter because the parameter that have this exception is being null and the debug console has provided the line that created this exception.

This exception can be caught and therefore can be handled.

This exception should be dealt by programmer as it occurs by programmer’s logic and should be fix by programmer.

Programmer should avoid this exception by finding the reason why the parameter is being null and fix it. Usually the parameter is set to null before any reference, just set it to something else that suitable to the working project.

IndexOutOfRangeException

This exception is thrown when you are trying to access an item in an array using an invalid index, usually the index is out of array’s range. For example, the array has length of 4 (0 to 3) but you are trying to access the 6th position.

System is in charge of throwing this exception since it mostly because of programmer’s fault.

The message that comes along with the exception is: $“You are trying to access the position {postion} of the array that have the length of {array.Length}” The parameters that included are the index position that is being access to and the length of the array.

This exception can be generally caught and be handled.

This exception should not be passed to user because it is devepoler’s fault.

This exception should be avoided by user by using a loop to set the limit to user access to the array’s range. If the exception is caused by the programmer, then it should be related to the logic of the program. Such as the array has the length of 4, then the maximum index should be 3. If programmer use a loop to work with this array, the condition should be “i<array.Length” or “i<=array.Length-1”, not “i<=array.Length”.

StackOverflowException

This exception is thrown when there is an infinite loop or an infinite recursion

The system is in charge of throwing this exception to programmer as it usually causes of programmer's error.

If this exception occurs, it should be displayed as message as this exception is vital to the programmer's program usability. The message should be the exception's message itself. As the system throws this exception, not programmer.

This exception can not be catch since .NET Framework 2.0, therefore the programmers must handle it themselves, by debugging for any unwanted infinite loop or infinite recursion.

OutOfMemoryException

This exception is thrown when the memory of the device is not enough to run the program

The system is in charge of throwing this exception because it is not programmer's fault.

If this exception occurs, the system should display the message, without any parameter because the programmer can hardly handle it.

This exception can be caught.

This exception is not the kind that you want to avoid because it's related to hardware, the way to fix it would be optimize the program to use less resources or run the program with a device that have enough memory.

InvalidCastException

This exception is thrown when the conversion of 2 types of data is not supported, for example: conversion from string to date and time.

The runtime system is in charge of throwing this exception because it reflects programmer's fault.

If this message needed to be thrown, the message would be the exception itself, because it is mainly used for debug process.

This exception should not be catch in the try catch block as it is because of programmer

This exception should be handled by programmer rather than catch or pass to user.

This exception can be avoided by checking datatype carefully before any conversion.

DivideByZeroException

This exception is thrown when there is a number (int, double, decimal) that divided by zero

The system should be in charge of throwing this exception because it is usually caused by programmer.

The message would be: "There is a number that divided to zero"

This exception can be caught and handled, and it should not pass to user because it's developer's fault.

This exception can be avoided by debugging to check if any parameter that is being used to divide by zero.

ArgumentException

This exception is thrown when there is an argument provide an invalid method

This exception should be thrown by programmer as it reflects errors during the execution of program, caused by user.

The message should be relevant to the error that caused this exception, for example: if user put an even number while program required an odd, it should say: $“{number} is not an even number!”.

This exception can be caught and handled.

If this exception occurs, it is better to pass it to user, because it reflects user’s fault in running program.

Programmer should try to handle and cover any errors that is potentially caused by user.

ArgumentOutOfRangeException

This exception is thrown when an argument that pass to a method that is not null and contain a value that is out of the defined range.

This exception should be thrown by programmer if there is an error caused by user, for example, input a negative number to when the program asked for human age (from 0 to 120), or by system if it is the case of developer’s fault, such as delete an item in an empty array, or not in the range of array.

The message would be $“Expected input from {from} to {to}.”

This exception can be catch by the system if it is caused by user.

Developer should avoid this exception by setting the limit to user input.

SystemException

This is the parent class of any other exception class, which means that any error can leads to this exception.

System is in charge to throw this exception. I don’t recommend throwing this because it doesn’t contain any useful information.

The message would be ”Something went wrong”

This exception can be caught. And should not pass to user since it doesn’t provide useful info.

As mentioned above, this is the base class of other exception so the way to avoid it should be handle the other subclass exceptions.