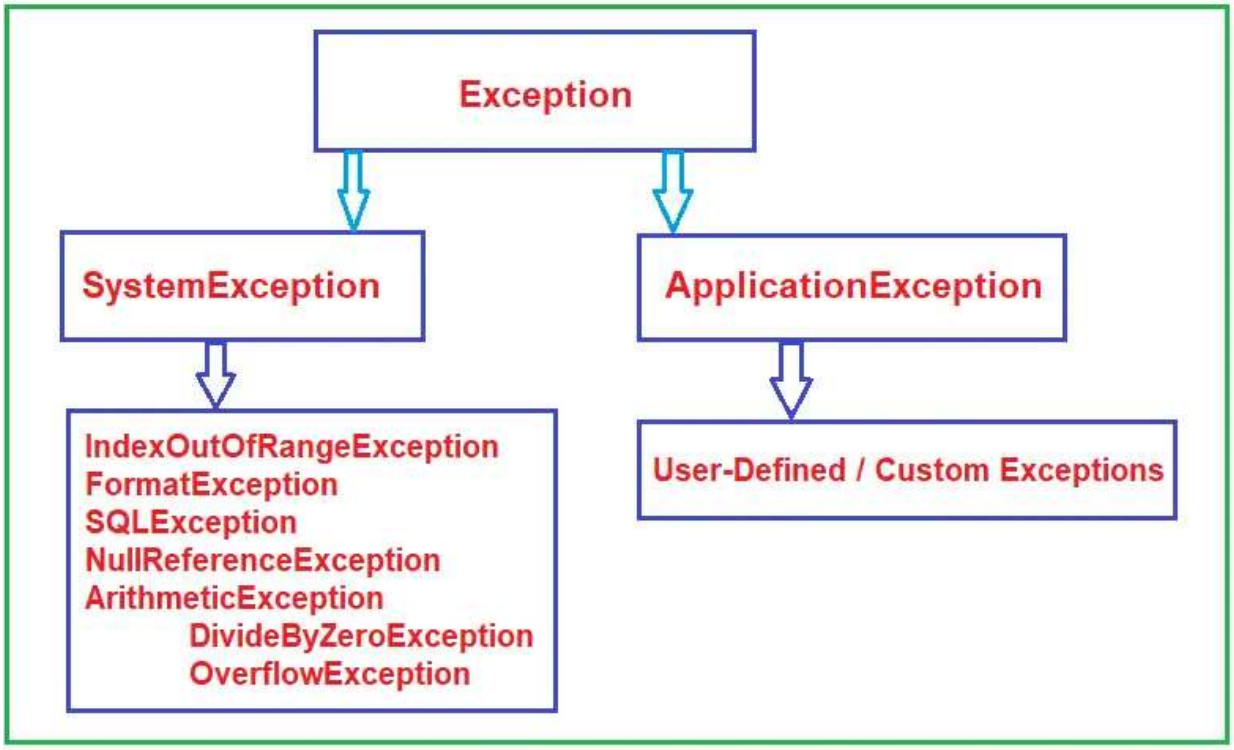
Create Custom Exceptions in C#

Types of Exceptions in C#:

Before understanding how to create Custom Exceptions or used Defined Exceptions in C#, let us first understand what the different types of Exceptions are available. In C#, the exceptions are divided into two types. They are as follows:

1. **System Exception:** These exceptions are caused by the CLR.
2. **Application Exception:** These exceptions are caused by the programmer.

For a better understanding, please have a look at the below image. The Exception is the parent class of all Exception classes. From the Exception class, SystemException and ApplicationException classes are defined.



As you can see, for both SystemException and ApplicationException, the parent is the Exception class only. By default, all the System Exception classes are inherited from the SystemException class which is inherited from the Exception class and if we are creating any Application Exception i.e. Custom Exception or user-defined exception, then that class should and must be inherited from the Exception class, even we can also create Custom Exception

classes inherit from ApplicationException class. Previously we used to create CustomException classes inheriting from the ApplicationException class, but currently, Microsoft recommended to use the Exception class.

What are System Exceptions in C#?

An exception that is raised implicitly under a program at runtime by the Exception Manager (Component of CLR) because of some logical mistakes (some predefined conditions) is known as System Exception. For example, if you are diving an integer number by zero, then one system exception is raised called DivideByZero. Similarly, if you are taking an alphanumeric string value from the user and trying to store that value in an integer variable, then one system exception is raised called FormatException. So, in C#, there are lots of System Exception classes are available. Some of them are as follows:

1. **DivideByZeroException**
2. **IndexOutOfRangeException**
3. **FormatException**
4. **SQLException**
5. **NullReferenceException, Etc.**

What are Application Exceptions in C#?

An exception that is raised explicitly under a program based on our own condition (i.e. user-defined condition) is known as an application exception. As a programmer, we can raise application exceptions at any given point in time. For example, our requirement is that while performing the division operation, we need to check that if the second number is an odd number, then we need to throw an exception. This cannot be handled automatically by the CLR. Then as a user, we need to create our Custom Exception and we need to create an instance of our Custom Exception class and we need to throw that Custom Exception instance using the throw keyword explicitly based on our business requirement.

To raise an Application Exception in C#, we need to adopt the following process. First, we need to create a custom Exception class by inheriting it from the Parent Exception class and then we need to create an instance of the Custom Exception class and then we need to throw that instance.

Step1: Create one Custom Exception Class

```
public class OddNumberException : Exception
{}
```

Step2: Create an instance of the Custom Exception Class

```
OddNumberException ONE = new OddNumberException();
```

Step3: Throw the Custom Exception Instance

```
throw ONE;
throw  new OddNumberException();
```