Module 4 Theory

*** For detailed theory/explanation along with examples/execution code , refer to the commented code in the practicals of the module ***

1) An **enumeration** is a set of named integer constants. An enumerated type is declared using the enum keyword.C# enumerations are value data type. In other words, enumeration contains its own values and cannot inherit or cannot pass inheritance.Each of the symbols in the enumeration list stands for an integer value, one greater than the symbol that precedes it. By default, the value of the first enumeration symbol is 0.

2) An **exception** is a problem that arises during the execution of a program. A C# exception is a response to an exceptional circumstance that arises while a program is running, such as an attempt to divide by zero.

Exceptions provide a way to transfer control from one part of a program to another. C# exception handling is built upon four keywords: try, catch, finally, and throw.

try − A try block identifies a block of code for which particular exceptions is activated. It is followed by one or more catch blocks.

catch − A program catches an exception with an exception handler at the place in a program where you want to handle the problem. The catch keyword indicates the catching of an exception.

finally − The finally block is used to execute a given set of statements, whether an exception is thrown or not thrown. For example, if you open a file, it must be closed whether an exception is raised or not.

throw − A program throws an exception when a problem shows up. This is done using a throw keyword.

3) An **interface** is defined as a syntactical contract that all the classes inheriting the interface should follow. The interface defines the 'what' part of the syntactical contract and the deriving classes define the 'how' part of the syntactical contract.Interfaces define properties,methods, and events, which are the members of the interface. Interfaces contain only the declaration of the members.It is the responsibility of the deriving class to define the members.It often helps in providing a standard structure that the deriving classes would follow.Abstract classes to some extent serve the same purpose, however, they are mostly used when only few methods are to be declared by the base class and the deriving class implements the functionalities.Interfaces are declared using the interface keyword. It is similar to class declaration. Interface statements are public by default.

4) **Inheritance** allows us to define a class in terms of another class, which makes it easier to create and maintain an application. This also provides an opportunity to reuse the code functionality and speeds up implementation time.

When creating a class, instead of writing completely new data members and member functions, the programmer can designate that the new class should inherit the

members of an existing class. This existing class is called the base class, and the new class is referred to as the derived class.

- Single inheritance - [one base,one child]
- Multi level inheritance - [one base->child->grand child->....]
- Multiple inheritance - [two or more base->one child]-not supported so implemented using interfaces
- Hierarchical inheritance - [one base,two or more child]

5) A **file** is a collection of data stored in a disk with a specific name and a directory path. When a file is opened for reading or writing, it becomes a stream.

The stream is basically the sequence of bytes passing through the communication path. There are two main streams: the input stream and the output stream. The input stream is used for reading data from file (read operation) and the output stream is used for writing into the file (write operation).

The System.IO namespace has various classes that are used for performing numerous operations with files, such as creating and deleting files, reading from or writing to a file, closing a file etc.

Generally, the file is used to store the data. The term File Handling refers to the various operations like creating the file, reading from the file, writing to the file, appending the file, etc. There are two basic operation which is mostly used in file handling is reading and writing of the file. The file becomes stream when we open the file for writing and reading. A stream is a sequence of bytes which is used for communication. Two stream can be formed from file one is input stream which is used to read the file and another is output stream is used to write in the file. In C#, System.IO namespace contains classes which handle input and output streams and provide information about file and directory structure.