Assignment No:01

1. Generation of ASP.Net

Ans:

- Classic ASP:
  - ➢ Introduced in 1998
  - ➢ Microsoft’s first server side scripting language

  Drawbacks:

  1. Loosely typed code and used to mix script code with html code
  2. Updates the files only when the server is working properly

- ASP .Net:
  - ➢ Released in 2002

  Drawbacks:

  1. Leaky Abstraction
  2. Poor security feartures

- ASP .Net Webpages:
  - ➢ ASP.NET is an single page application model.

  Drawback:

  1. Not good for large project

- ASP .Net MVC:

- ➢ ASP.NET is an MVC application model
- ➢ Merged into .NET Core

Drawback: The complexity is high to develop the applications using this **pattern**.

- ASP .Net API:

  Drawbacks:
  1. Hosting dependency
  2. Protocol dependency
  3. Serialisation dependency
  4. Data transformation

- ASP.Net Web Forms:

  Drawbacks:

  1. Complex Pages with Performance issues
  2. Limited support for testing

- ASP .NET Core:

  ASP.**NET Core** is a cross-platform, high-performance, open-source framework for building modern, cloud-enabled, Internet-connected apps. With **ASP**.**NET Core**, you can: Build web apps and services, Internet of Things (IoT) apps, and mobile backends

2. Advantages of ASP .NET

Ans:

- It allows for separation of concern

ASP.NET follows the MVC architecture, which allows for separate input, process and output of the application. This three-tier architecture, Model-View- Controller has interconnected parts, and can handle specific development aspects of software applications.

- Reduces coding time

The framework technology is a big help in reducing coding time, especially when you are developing big applications.

- Cross-platform migration

The framework language allows for easy cross-platform migration, configuration and deployment services.

- Manageability

The excellent manageability feature of the framework is contributed through its text based hierarchical configuration system.

- Delivers power and flexibility

The framework language is based on common language runtime, so all the web application developers can enjoy flexibility and power of that entire platform. It is also language independent, so you can choose the language for your application or even divide your application across several languages.

- Simplicity

Each task can be performed easily, even the most common ones to the complicated and tricky ones.

3. Disadvantages of ASP .NET

Ans:

- Security

Yes, we did mention security as one of the advantages of the framework. But apparently, extra care should be taken to protect the applications.

- Costly

When compared to open source alternatives, ASP.NET is expensive, as you have expenses like SQL Server licenses, Visual Studio licenses, Windows server licenses, etc.

- Making changes in the app

Changes might not work in the next version of your app. What works in the current version, works, but in the next, you might have to get help from GitHub.

- Porting ASP application from one server to another is expensive

4. Difference Between Asp.Net VS Asp.Net Core

Ans:

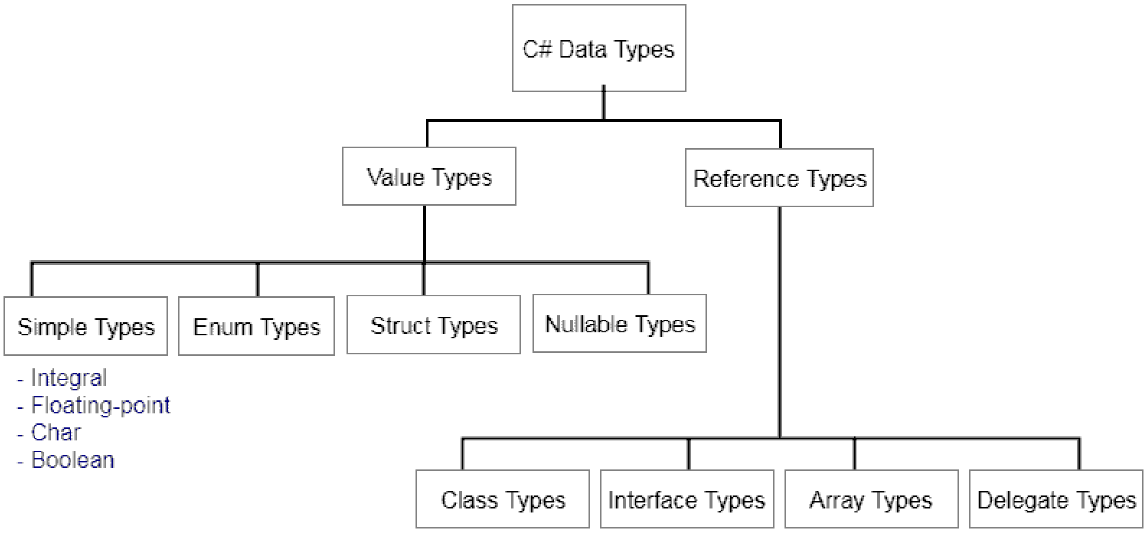
ASP.NET	**ASP.NET CORE**
Asp.Net Build for Windows	Asp.Net Core Build for Windows, Mac and Linux
Asp.Net has a Good Performance	ASP.Net core has higher performances than ASP.Net 4x.
It runs on .Net Framework or commonly called as full .Net Framework	It runs on .Net Core and Full .Net Framework.
Asp.Net Supports WebForm, Asp.Net MVC and Asp.Net WebAPI.	Asp.Net Core does not support WebForm. It supports MVC, Web API and Asp.Net Web pages originally added in .Net Core 2.0.

Asp.Net used the only IIS with dependant on System.web.dll.	Asp.Net Core has not dependant System.web.dll and so the IIS.
Support C#, VB and many other languages and also support WCF, WPF and WF	Support only C#, F# language. VB support to added a short time and no support WCF, WPF and WF but support for WCF client libraries are available.
Asp.Net MVC application added Web.config, Global.asax, Application Start.	Core did not support Web.config and Global.asax files. It is supporting appsettings.json.
Container support not more than better as the ASP.Net Core application.	Container support best suited for deployments like Docker.
All major versions supported	Support Core from Visual Studio 2015 update 3 and current version VS 2017.
We Need to re-compile after the code change.	Core Browser refresh will compile and executed the code no need for re-compile.

Assignment No2

1) Datatypes and Range of datatypes:

Data Types	Memory Size	Range
char	1 byte	-128 to 127
signed char	1 byte	-128 to 127
unsigned char	1 byte	0 to 127
short	2 byte	-32,768 to 32,767
signed short	2 byte	-32,768 to 32,767
unsigned short	2 byte	0 to 65,535
int	4 byte	-2,147,483,648 to -2,147,483,647
signed int	4 byte	-2,147,483,648 to -2,147,483,647
unsigned int	4 byte	0 to 4,294,967,295
long	8 byte	?9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
signed long	8 byte	?9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
unsigned long	8 byte	0 - 18,446,744,073,709,551,615
float	4 byte	1.5 * 10^{-45} - 3.4 * 10^{38}, 7-digit precision
double	8 byte	5.0 * 10^{-324} - 1.7 * 10^{308}, 15-digit precision
decimal	16 byte	at least -7.9 * $10^{?28}$ - 7.9 * 10^{28}, with at least 28-digit precision



2) **Operators:**

Arithmetic Operators

These are used to perform arithmetic/mathematical operations on operands. The **Binary Operators** falling in this category are :

- Addition: The '+' operator adds two operands. For example, x+y.
- Subtraction: The '-' operator subtracts two operands. For example, x-y.
- Multiplication: The '*' operator multiplies two operands. For example, x*y.
- Division: The '/' operator divides the first operand by the second. For example, x/y.
- Modulus: The '%' operator returns the remainder when first operand is divided by the second. For example, x%y.

Unary Operators

- Increment
- Decrement

Relational Operators:

- == Equal To
- != Not Equal to
- > Greater than
- < Less than
- >= Greater than Equal to
- <= Less than equal to

Logical Operators

- && Logical AND
- || Logical OR
- ! Logical Not

Bitwise Operators

- & Binary AND
- | Binary OR
- ^ Binary XOR
- ~ Binary Ones Complement
- << Binary Left Shift Operator
- >> Binary Right Shift Operator

Assignment Operators:

- = assignment operator
- += Add AND assignment operator
- -= Subtract AND assignment operator
- /= Divide AND assignment operator
- %= Modulus AND assignment operator
- <<= Left shift AND assignment operator

- >>= Right shift AND assignment operator
- &= Bitwise AND assignment operator
- ^= bitwise exclusive OR and assignment operator

Conditional Operator

It is ternary operator which is a shorthand version of if-else statement. It has three operands and hence the name ternary. It will return one of two values depending on the value of a Boolean expression.

Syntax:
Condition? First_Expression: Second_Expression;

3)Local And Global Variables:

Local Variable is defined as a type of variable declared within programming block or subroutines. It can only be used inside the subroutine or code block in which it is declared. The local variable exists until the block of the function is under execution. After that, it will be destroyed automatically.

Global Variable in the program is a variable defined outside the subroutine or function. It has a global scope means it holds its value throughout the lifetime of the program. Hence, it can be accessed throughout the program by any function defined within the program

ASSIGNMENT NO:03

1. **Mention all different data types we can pass in switch ()**

Ans:

The switch expression is of integer type such as int, char, byte, or short, or of an enumeration type, or of string type. The expression is checked for different cases and the one match is executed

2. **Practice switch/break removing break from various cases and also remove default and see the output in programs.**

Ans:

- The use of break statement in switch is essential
- Instead of break we can use goto or return statement
- The default statement is optional and it can be used anywhere inside the switch statement.
- Multiple default statements are not allowed.

3. **When to use else if ladder and when to use switch case?**

We use a switch statement instead of if-else statements because if-else statement only works for a small number of logical evaluations of a value. If you use if-else statement for a larger number of possible conditions then, it takes more time to write and also become difficult to read. switch statement to save time and write optimized code. Using switch statement will provide a better readability of code.

4.**Difference between Else if ladder and Switch case**

Ans:

else if ladder	**Switch Case**
In else if ladder, the control goes through the every else if statement until it finds true value of the statement or it comes to the end of the else if ladder.	In case of switch case, as per the value of the switch, the control jumps to the corresponding case.
Not Readable	switch is considered to be more readable.

there is no need of use of break in else if ladder.	The use of break statement in switch is essential
in else if ladder accepts integer type as well as character	The variable data type that can be used in expression of switch is integer only
In case of else if ladder, the code needs to be processed in the order determined by the programmer.	Each case in switch statement is independent of the previous one
else if ladder works on the basis of true false(zero/non-zero) basis.	Switch case statement work on the basis of equality operator
Low speed than Switch	High speed

5. What if we missed last else in else if ladder
Ans: Error Doesn’t Occur

6. Arrays, String, Sting buffer and String builder
Ans:
Array: Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.
int arr[]={10,20,30};

String: String String is immutable (once created cannot be changed) object.
String once assigned cannot be changed.
Every immutable object is thread safe, that implies String is also thread safe. String cannot be used by two threads simultaneously.

StringBuffer: StringBuffer is mutable means one can change the value of the object. The object created through StringBuffer is stored in the heap. StringBuffer has the same methods as the StringBuilder, but each method in StringBuffer is synchronized that is StringBuffer is thread safe.

StringBuilder: StringBuilder is same as the StringBuffer, that is it stores the object in heap and it can also be modified . The main difference between the StringBuffer and StringBuilder is that StringBuilder is also not thread safe. StringBuilder is fast as it is not thread safe.

ASSIGNMENT 04

OOP Concepts:

1. Class:

- It Contains variables and methods.
- Without creating object there is no meaning to class.
- It is user defined data type

2. Object:

- It is run time entity
- An object is instance of the class
- Class does not occupy any memory but when object is created then memory will allocate to class.

3. Encapsulation:

- Class Boundary is Encapsulation
- Wrapping the data members and member functions in single unit.

4. Abstraction:

- It is the technique of showing only essential details without representing the implementation details.
- Ex: ATM Machine

5. Inheritance:

-When a class includes a property of another class it is known as inheritance.

-Inheritance is a process of object reusability.

6. Data Binding:

- Binding is process of binding the application UI and Business logic. Any changes made in business logic will reflect directly to the application UI.

7. Polymorphism:

- Polymorphism means multiple forms.
- Having more than one function with same name but different functionality.

Types of polymorphism:

1) Static Polymorphism
2) Dynamic Polymorphism

8. Interface:

- It is like abstract class because all the methods which are declared inside the **interface** are abstract methods. It cannot have method body and cannot be instantiated. It is used to achieve multiple inheritance which can't be achieved by class.

ASSIGNMENT No:05

1. Difference Between abstract class and Interface

Abstract Class	Interface
Multiple inheritance is not achieved by abstract class.	Multiple inheritance is achieved by interface.
It contain constructor.	It does not contain constructor.
It can contain static members.	It does not contain static members.
It can contain different types of access modifiers like public, private, protected etc.	It only contains public access modifier because everything in the interface is public.
The performance of an abstract class is fast.	The performance of interface is slow because it requires time to search actual method in the corresponding class.
It is used to implement the core identity of class.	It is used to implement peripheral abilities of class.
A class can only use one abstract class.	A class can use multiple interface.

2. Abstract class and sealed class

Abstract Class	**Sealed Class**
A class that contains one or more methods is known as abstract class	A Class which it is not possible to derive a class is known as a sealed

	class
The abstract class contains abstract and non-abstract methods	It cannot contain abstract and virtual methods
Creating new class from abstract class is compulsory to consume it	It is not possible to create a new class from sealed class
An abstract class cannot be instantiated directly, we need to create the object for its child classes to consume an abstract class	We should create an object for sealed class to consume its memory
We need to use the keyword abstract to make any class as abstract	We need to use the keyword sealed to make any class as sealed
As abstract class cannot be the bottom most class within the inheritance hierarchy	The sealed class should be the bottommost class within the inheritance hierarchy

3. Constructor:

A constructor is a special method of the class which gets automatically invoked whenever an instance of the class is created. Like methods, a constructor also contains the collection of instructions that are executed at the time of Object creation. It is used to assign initial values to the **data members** of the same class.

- Constructor of a class must have the same name as the class name in which it resides.

- A constructor can not be abstract, final, and Synchronized.
- A constructor doesn't have any return type, not even void.
- A class can have any number of constructors.
- Access modifiers can be used in constructor declaration to control its access i.e which other class can call the constructor.

Types of Constructor:

1. Default Constructor
2. Parameterized Constructor
3. Copy Constructor
4. Private Constructor
5. Static Constructor

4. Static Constructor

Static Constructor has to be invoked only once in the class and it has been invoked during the creation of the first reference to a static member in the class. A static constructor is initialized static fields or data of the class and to be executed only once.

Points To Remember :

- It can't be called directly.
- When it is executing then the user has no control.
- It does not take access modifiers or any parameters.
- It is called automatically to initialize the class before the first instance created.

5. Get and set in c#

The get set accessor or modifier mostly used for storing and retrieving the value from the private field. The get accessor must return a value of property type where set accessor returns void. The set accessor uses an implicit parameter called value. In simple word, the get method used for retrieving the value from private field whereas set method used for storing the value in private variables.

static is a modifier in C# which is applicable for the following:

- Classes
- Variables
- Methods
- Constructor

Static Class

A static class is declared with the help of *static* keyword. A static class can only contain static data members, static methods, and a static constructor. It is not allowed to create objects of the static class. Static classes are ***sealed***, means one cannot inherit a static class from another class.

Static Variable

A static variable is declared with the help of static keyword. When a variable is declared as static, then a single copy of the variable is created and shared among all objects at the class level. Static variables are accessed with the name of the class, they do not require any object for access.

Static Method

A static method is declared with the help of static keyword. Static methods are accessed with the name of the class. A static method can access static and non-static fields, static fields are directly accessed by the static method without class name whereas non-static fields require objects.

Sealed classes are used to restrict the users from inheriting the class. A class can be sealed by using the ***sealed*** keyword. The keyword tells the compiler that the class is sealed, and therefore, cannot be extended. No class can be derived from a sealed class.

Assignment No:06

1) Difference Between Method and constructor

Constructors	**Methods**
A Constructor is a block of code that initializes a newly created object.	A Method is a collection of statements which returns a value upon its execution.
A Constructor is invoked implicitly by the system.	A Method is invoked by the programmer.
A Constructor is invoked when a object is created using the keyword **new**.	A Method is invoked through method calls.
A Constructor doesn’t have a return type.	A Method must have a return type.
A Constructor’s name must be same as the name of the class.	A Method’s name can be anything.
A Constructor cannot be inherited by subclasses.	A Method can be inherited by subclasses

2) Copy Constructor:

A constructor that creates an object by copying variables from another object or that copies the data of one object into another object is termed as the **Copy Constructor**. It is a parameterized constructor that contains a parameter of the same class type. The main use of copy constructor is to initialize a new instance to the values of an existing instance. Normally, C# does not provide a copy constructor for objects, but if you want to create a copy constructor in your program you can create according to your requirement.

3) Difference between sorted list and hashmap

SortedList Class:

- A SortedList element can be accessed by its key or by its index.
- A SortedList object internally maintains two arrays to store the elements of the list, i.e, one array for the keys and another array for the associated values.
- A key cannot be null, but a value can be.
- The capacity of a SortedList object is the number of elements the SortedList can hold.
- A SortedList does not allow duplicate keys.
- Operations on a SortedList object tend to be slower than operations on a Hashtable object because of the sorting.
- Elements in this collection can be accessed using an integer index. Indexes in this collection are zero-based.

Characteristics of Hashtable Class:

- In Hashtable, key cannot be null but value can be.
- In Hashtable, key objects must be immutable as long as they are used as keys in the Hashtable.
- The capacity of a Hashtable is the number of elements that Hashtable can hold.
- A hash function is provided by each key object in the Hashtable.

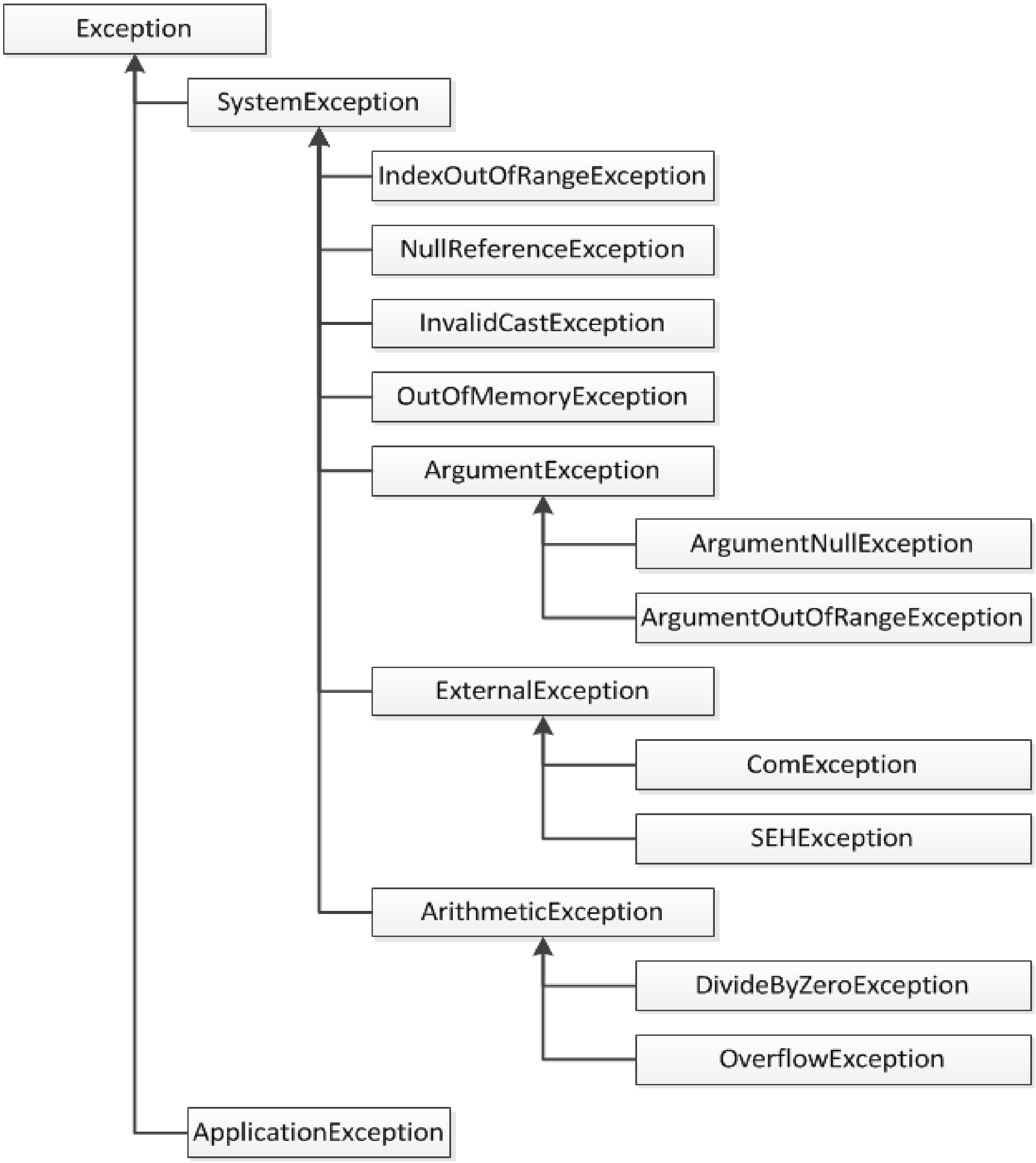
4) Delegate

A delegate is an object which refers to a method or you can say it is a reference type variable that can hold a reference to the methods. Delegates in C# are similar to the function pointer in C/C++. It provides a way which tells which method is to be called when an event is triggered.

For example, if you click an *Button* on a form (Windows Form application), the program would call a specific method. In simple words, it is a type that represents references to methods with a particular parameter list and return type and then calls the method in a program for execution when it is needed.

Assignment N0 :07

1) Hierarchy of exception classes



2) **Exception:** An exception is defined as an event that occurs during the execution of a program that is unexpected by the program code. The actions to be performed in case of occurrence of an exception is not known to the program. In such a case, we create an exception object and call the exception handler code. The execution of an exception handler so that the program code does not crash is called exception handling. Exception handling is important because it gracefully handles an unwanted event, an exception so that the program code still makes sense to the user.

Keyword	Definition
try	Used to define a try block. This block holds the code that may throw an exception.
catch	Used to define a catch block. This block catches the exception thrown by the try block.
finally	Used to define the finally block. This block holds the default code.
throw	Used to throw an exception manually.

3) Types of exception under checked and unchecked exception:

Unchecked Exception

Checked Exceptions occur at the runtime of the program. These are also known as Runtime Exceptions. It is not a requirement to handle or catch them at compile time. These are mainly due to the mistakes in the program.

Examples Unchecked Exceptions

- Arithmetic Exception

- Null Pointer Exception
- Array Index Out of Bounds Exception
- Unsupported Operation Exception
- Security Exception
- System Exception
- Missing Resource Exception
- No Such Element Exception
- Undeclared Throwable Exception
- Empty Stack Exception

Checked Exceptions

Checked Exceptions occur at the compile time of the program. These exceptions should be either caught or handled during compile time. If we do not catch or handle them then the compiler will throw a compilation error. These are the sub classes of the Exception class.

Examples of Checked Exceptions

- No Such Field Exception
- File Not Found Exception
- Interrupted Exception
- No Such Method Exception
- Class Not Found Exception
- Parse Exception
- Clone Not Supported Exception
- Instantiation Exception

When to use Checked Exceptions

- Where chances of failure are greater.
- Mainly when an exception occurs and we know what we need to do.

When to use Unchecked Exceptions

- Business logic.
- Rules Validation.
- User Input Validation.

Difference between Checked and Unchecked Exception

- Checked Exceptions are the sub-class of the Exception class, on the other hand Unchecked Exceptions are Runtime Exceptions.
- In Checked Exceptions the Java Virtual Machine requires the exception to be caught or handled, while in Unchecked Exceptions the Java Virtual Machine does not require the exception to be caught or handled.
- Checked Exceptions are checked at runtime of the program, while Unchecked Exceptions are checked at the compile time of the program.
- Checked Exceptions and Unchecked Exceptions both can be created manually.
- Checked Exceptions and Unchecked Exceptions both can be handled using try, catch and finally.
- Unchecked Exceptions are mainly programming mistakes.
- Unchecked Exceptions can be ignored in a program but Unchecked Exceptions cannot be ignored in a program.