PT – Tutorial 2

1. Explain the Seven key pillars of ASP.net.

Ans The Seven Key Pillars of Asp.net are as Follows:-

a) Asp.NET is integrated With the .NET Framework:

The .NET Framework is Consists of thousands of numbers of classes, structures, interfaces, and other Core Programming ingredients which are grouped into a logical, hierarchical Container Called a namespace.

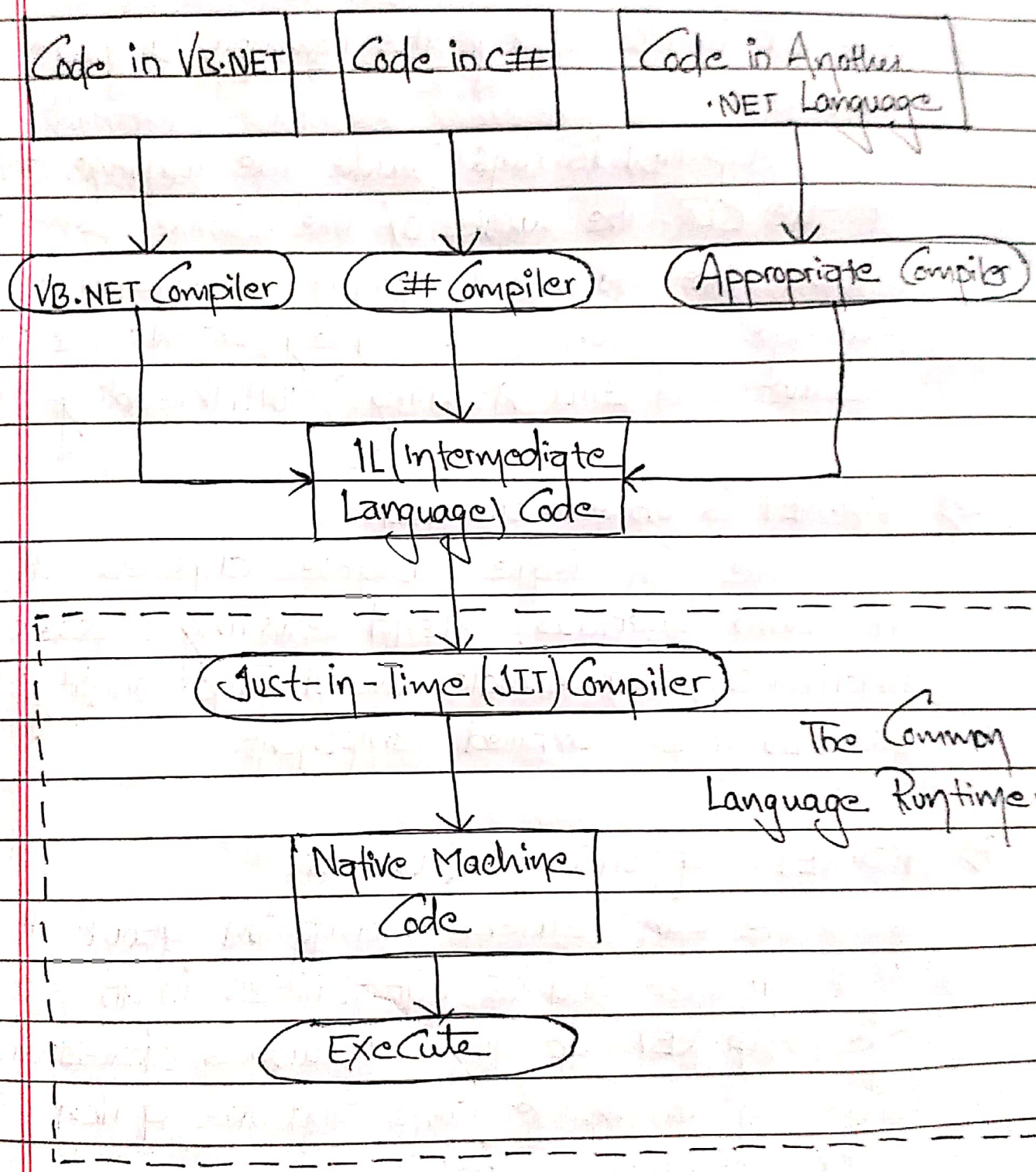
b) Asp.NET is Compiled, Not Interpreted

Asp.NET applications, like all .NET applications, are always Compiled. .NET applications actually go through two stages of Compilation.

- In the First stage, the C# Code is Compiled into an intermediate language Called MicroSoft Intermediate Language (MSIL), or just IL.
- The Second level of Compilation happens just before the page is actually executed. At this Point, the IL Code is Compiled into low-level native machine Code. This stage is known as just in time (JIT) Compilation, and it takes place in the same way For all .NET applications.

Scanned with CamScanner

Compilation of asp.net Web Page



c) ASP.NET is Multilanguage:

Asp.net Support multiple language like C#, VB.net etc. No matter what language we use, the

Code is Compiled into IL.

d) ASP.NET is Hosted by the Common Language Runtime:

ASP.NET is runs inside the runtime environment of the CLR. The whole of the Some benefits of CLR are Automatic memory management and garbage Collection, type Safety, Extensible metadata, Structured error Handling, Multithreading etc.

e) ASP.NET is Object-oriented:

We Can Create reusable classes, Standardize Code With interfaces, extend existing Classes With inheritance, and bundle useful Functionality in a distributable, Compiled Component.

F) ASP.NET Supports all Browsers:

We Can retrieve information about the Client browser and its Capabilities in an Asp.NET Page, Asp.NET actually encourages developers to ignore these Considerations and use a rich Suite of Web Server Controls.

g) ASP.NET is Easy to Deploy and Configure:

Every installation of the .NET Framework

Scanned with CamScanner

Provides the Same Core Classes. As a result, deploying an Asp.NET application is relative Simple. We Simply need to Copy all the Files to a virtual directory on a Production Server.

2. Clarify your understanding and uses of Asp.Net File types and Asp.net application directories.

Ans

File Types	Description
.aspx	These Asp.NET Web Pages Contains the user interface and, optionally, the underlying application Code. users request or navigate directly to one of these Pages to Start web application.
.ascx	These are Asp.NET User Controls. User Controls are Similar to web Pages, except that the user Can't access these Files directly. Instead, they must be hosted inside an Asp.NET Web Page. User Controls allow us to develop a Small Piece of user interface and reuse it in as many web Forms.

Scanned with CamScanner

Web.Config — This is the XML-based Configuration File for Asp.NET application. It includes Settings for Customizing Security, Sta Management, memory management and much more.

global.asax — This is the global application File. We can use this File to define global Variables and react to global events (such as when a Web application First Starts).

.cs — These are Code-behind Files that contai C# Code. They allow us to Separate the application logic from the user interfa of a Web Page.

.asmx — An XML Web Services file that Contains Classes and methods that are available to other Web applications by way of SOAP

.browser — A browser definition file used to identify the Features of Client browsers.

Scanned with CamScanner

- .dll A Compiled Class library File (assembly).
- .master A master page that defines the layout for other web pages in the application.
- .mdF SQL database File for use with SQL Server Express.
- .Sitemap A Site-map File that Contains the Structure of the web Site.

ASP.NET Application Directories

Directory	Description
App_Browsers	Contain .browser files that ASP.NET uses to identify the browsers that are using our applications and determine their Capabilities. This Folder is optional.
App_Code	Contain Source Code Files that are dynamically Compiled for use in our Application.

Scanned with CamScanner

App_GlobalResources — Stores global resources that are accessible to every page in the web Application. This directory is used in localization scenarios, when we need to have a website in more than one language.

App_LocalResources — Serves the same purpose as App_GlobalResources, except these resources are accessible to a specify page only.

App_WebReferences — Stores references to web services, which are remote code routines that a web Application can call over a network or the internet.

App_Data — Stores data, including SQL Server Express database files

App_Themes — Stores the themes that are used to standardize and reuse formatting in your web application.

Bin — Contains all the compiled .NET components

Scanned with CamScanner

(DLLs) that the Asp.NET web application uses.

3. What are the Server Controls in Asp.net? Explain the life Cycle of asp.net Page.

Ans Controls are small building blocks of the graphical user interface, which include text boxes, buttons, check boxes, list boxes, labels, and numerous other tools. Using these tools, the users can enter data, make selections and indicate their preferences.

They run on the web server and they automatically provide their own HTML output. Even better, server controls behave like their windows counterparts by maintaining state and raising events that we can react to in code.

Asp.NET actually provides two sets of server-side controls.

- HTML Server Control
  - These are server-based equivalents for standard HTML elements.
  - They generates their own interface.
  - They retain their state.

Scanned with CamScanner

- They Fire Server-Side events.

- Web Controls
- These are Similar to the HTML Server Controls, but they Provide a richer object model with a variety of Properties for Style and Formatting details.
- They Provide a rich user interface.
- They Provide a Consistent Object model.
- They tailor their Output automatically.
- They Provide high-level Features.

ASP.NET Page Life Cycle

Stage	Description
Page request	The page request occurs before the page life cycle begins. When the page is requested by a user, ASP.NET determines whether the page needs to be Parsed and Compiled or whether a Cached Version of the page Can be Sent in response without running the page.

Scanned with CamScanner

Start — In the Start Stage, Page Properties Such as Request and Response are Set. At this Stage, the Page also determines whether the request is a Postback or a new request and Sets the IsPostBack Property.

Initiallization — During page Initiallization, Controls on the Page are available and each Control's UniqueID Property is Set. A MasterPage and themes are also applied to the Page if applicable. If the Current request is a Postback, the Postback data has not yet been loaded and Control Property Values have not been restored to the Values from View State.

Load — During Load, if the Current request is a Postback, Control Property are loaded With information recovered from ViewState and Control State.

Postback event handling — If the request is a Postback, Control event handlers are Called. After that, the Validate method of all validator

Controls is called, which sets the IsValid property of individual Validator controls and of the page.

Rendering Before rendering, View State is saved for the page and all controls. During the rendering stage, the page calls the Render method for each control, providing a text writer that writes its output to the OutputStream object of the page's Response property.

Unload. The Unload event is raised after the page has been fully rendered, sent to the client, and is ready to be discarded. At this point, page properties such as Response and Request are unloaded and cleanup is performed.

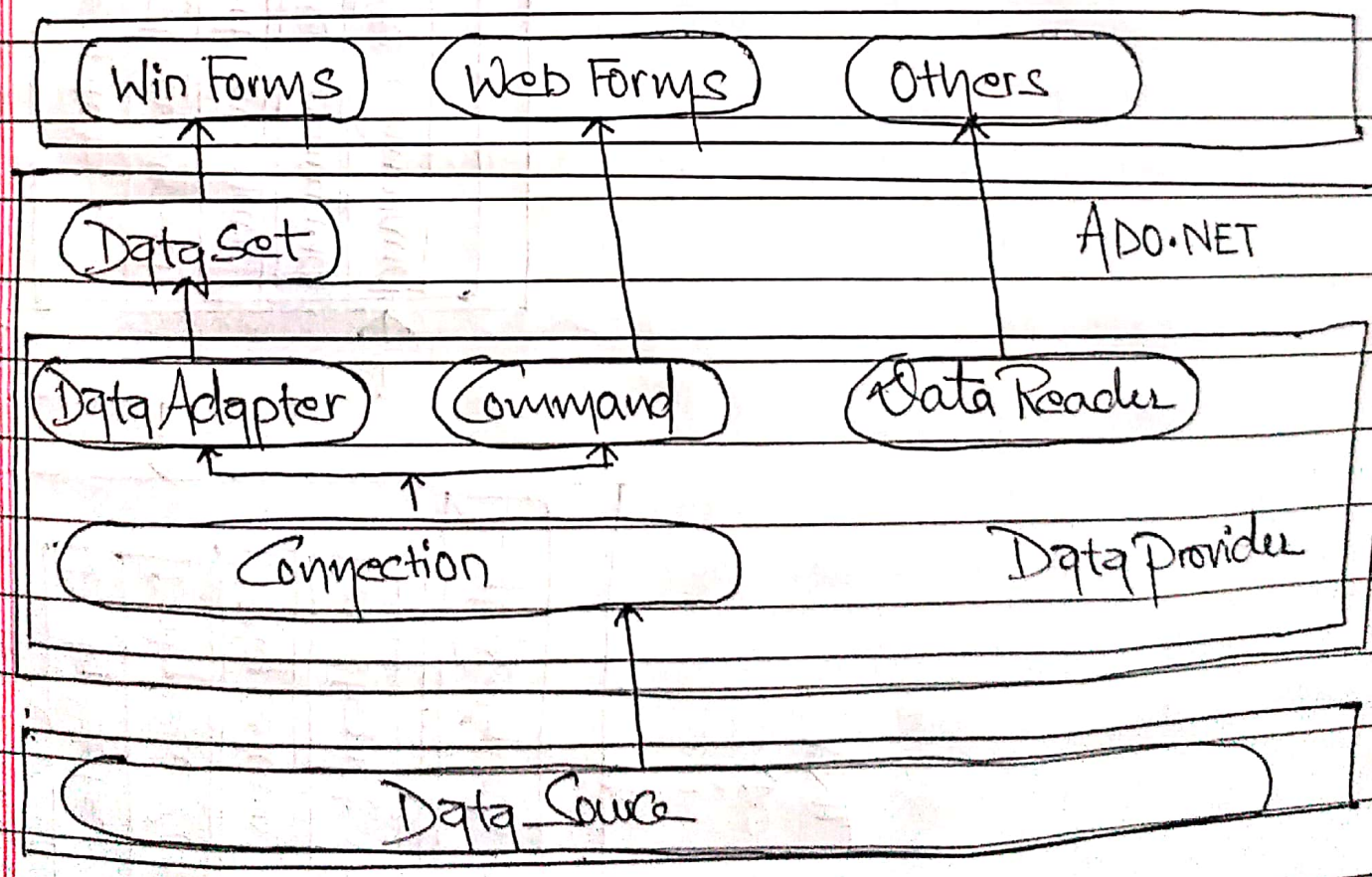
4. What is Connection string? Explain the architecture of ADO.NET?

Ans In computing, a connection string is a string that specifies information about a data source and the means of connecting to it. It is

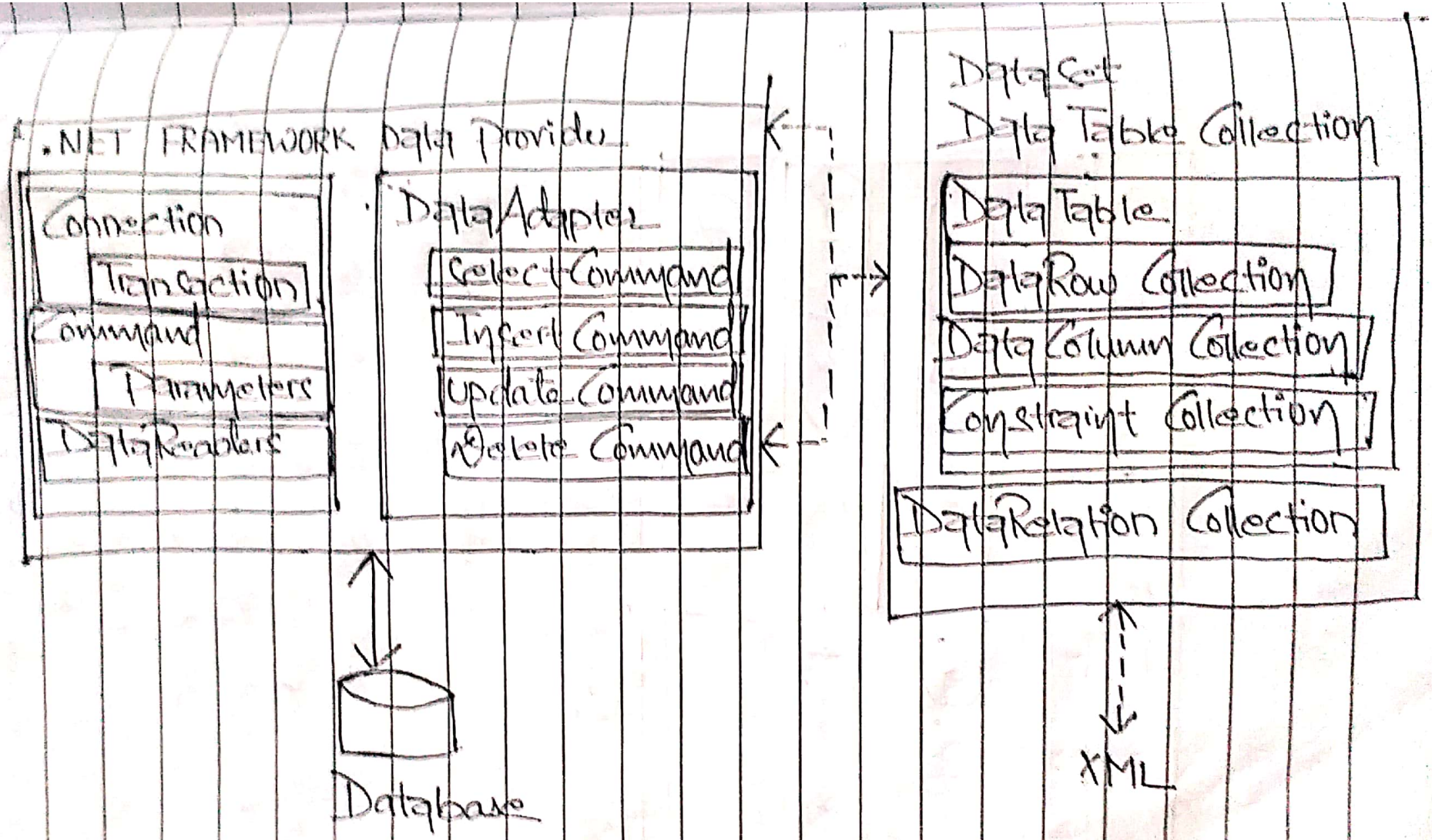
passed in code to an underlying driver or provider in order to intiqlite the connection. whilist commonly used for a database connection, the data source could also be a spreadsheet or text file.

The connection string may include attributes such as the name of the driver, server and database, as well as security information such as user name and password.

ADO.NET Architecture



Scanned with CamScanner



Scanned with CamScanner

Object	Description
Connection	Establishes a Connection to a specific data Source
Command	Executes a Command against a data Source. Exposes Parameters and Can execute within the Scope of a Transaction from a Connection.
DataReader	Reads a Forward-only, read-only stream of data from a data Source
DataAdapter	populates a DataSet and resolves updates with the data Source.
DataTable	Has a Collection of DataRows and DataColumns representing table data, used in disConnected model.
DataSet	Represents a Cache of data. Consist of a Set of DataTables and relations among them.

Scanned with CamScanner

5. Write a Simple asp.net Program in order to demonstrate the Connectivity to the database Server (use Connection String).

Ans

```
System.Data.SqlClient namespace.
String ConnectionString = WebConfigurationManager.
ConnectionStrings["DBConnectionString"].
ConnectionString;
SqlConnection Con = new SqlConnection(ConnectionString);
try
{
Con.Open();
lblInfo.Text = "<b>ServerVersion:</b>" + Con.ServerVersion;
lblInfo.Text += "<br/><b>Connection Is:</b>" + Con.State.ToString();
}
Catch (Exception err)
{
lblInfo.Text = " Error reading the database." +
    err.Message;
}
Finally
{
```

Scanned with CamScanner

```
Con.Close();
lblInfo.Text += "<br/><b>Now Connection Is:</b>" +
Con.State.ToString();
}
```

6. What is JIT Compilation ? Differentiate JVM, JRE and JDK ?

Ans Initially Java has been accused of poor performance because it's both compiles and interprets instruction. Since compilation or Java File to Class File is independent of execution of Java Program. Here compilation word is used for byte code to machine instruction translation. JIT are advanced part of Java virtual machine which optimize byte code to machine instruction conversion part by compiling similar byte codes at same time and thus reducing overall execution time. JIT is part of Java Virtual Machine and also performs several others optimization such as in-lining function.

In short here are few differences between JRE, JDK and JVM

a) JRE and JDK come as installer while JVM are bundled with them.

b) JRE only contain environment to execute java program but doesn't contain other tool for compiling java program.

c) JVM comes along with both JDK and JRE and created when we execute java program by giving "java" command.

7. How error is differ from exception. Write a java program to demonstrate the concept of illustrates the use of try, catch, throw, throws and finally keyword.

Ans Both Error & Exception are derived from java.lang.Throwable in java but main difference between Error and Exception is kind of error they represent. java.lang.Error represent errors which are generally can not be handled and usually refer catastrophic failure eg. running out of system resources, some examples of Error in java are java.lang.OutOfMemoryError or java.lang.NoClassDefFoundError and java.lang.UnSupportedClassVersionError. on the other hand java.lang.Exception represents errors which can

be Catch and dealt e.g. IOException which Comes while performing I/O operations i.e reading Files and directories.

using try/Catch

```
Class execq{
Public Static Void main (String[] args){
int d,a;
try{
d=0.
a=42/d;
System.out.Println("This will not be Printed");
} Catch (Airthemetic Exception e){
System.out.Println("Division by zero"+e
Print Stack Trace());
}
System.out.Println("After try/Catch block");
}
}
```

throw

```
Class ThrowDemo{
Static Void demoproc(){
try{
```

Scanned with CamScanner

```
throw new NullPointerException("demo");
} Catch (NullPointerexception e){
System.out.println(" Caught inside demoproc");
    throwe;
}
}
Public Static Void main (String[] args){
try{
demoproc();
} Catch (NullPointerexception e){
System.out.Println("ReCaught" + e);
}
}
}
```

<u>throws</u>

```
Class ThrowsDemo{
Static Void throwone() throws IllegalAccess
exception {
System.out.Println("Inside throwone");
throw new IllegalAcess Exception("demo");
}
Public Static Void main (String[] args){
try{
    throwone();
} Catch (IllegalAcess Exception e){
```

Scanned with CamScanner

```
System.out.Println("Caught"+e);
    }
  }
}
```

Finally

```
Class FinallyDemo {
Static Void ProcA(){
try {
System.out.Println("Inside ProcA");
  throw new RuntimeException("demo");
} finally {
System.out.Println("ProcA's finally");
  }
}
Static Void ProcB(){
try {
System.out.Println("Inside ProcB");
  return;
}
finally {
  System.out.Println("Proc B's Finally");
}
}
Static Void Proc C(){
```

Scanned with CamScanner

```
try{
System.out.Println("Inside Proc e");
}
finally{
System.out.Println("Proc's finally");
}
}
Public Static Void main (String [] args) {
  try{
Proc A();
  } Catch (exception e){
System.out.Println(" Exception Caught "+e);
  }
Proc B();
  Proc C();
  }
}
```

8. Write a Java Program to demonstrate the Concept of user defined exception taking any example Scenario.

Ans

```
Class MyException extend Exception {
  Private int detail;
  My Exception (int a){
```

Scanned with CamScanner

```
detail = a;
}
Public String toString (){
return "MyException[" + detail + "]";
}
}
Class ExceptionDemo{
Static void Compute (int a) throws MyException {
System.out.println ("Called Compute (" + a + ")");
if (a>10)
throw new MyException (a);
System.out.Println ("Normal Exit");
}
Public Static void main (String [] args){
try {
Compute (1);
Compute (20);
} Catch (My Exception e){
System.out.Println ("Caught" + e);
}
}
}
```

O/P

```
Called Compute (1)
Normal Exit
Called Compute (20)
Caught MyException[20]
```

Scanned with CamScanner

9. What is debugging? Explain different debugging techniques with suitable examples. Section of Code.

Ans Debugging is the process of detecting and removing of existing and potential errors (also called as 'bugs') in a software code that can cause it to behave unexpectedly or crash. To prevent incorrect operation of a software or system, debugging is used to find and resolve bugs or defects.

The different debugging techniques are as follows:-

a) Loop Bug

The two most common kinds of loop errors are unintended infinite loops and off-by-one errors

Infinite Loop

A while loop, do-while loop, or for loop does not terminate as long as the controlling Boolean expression evaluates to true. However, if we make a mistake and write our program so that the Boolean expression is always true, then the loop will run forever. A loop that runs forever is called an infinite loop.

Scanned with CamScanner

OFF-by-one error: The loop repeats the loop body one too many or one too few times. These sorts of errors can result from carelessness in designing a controlling Boolean expression. For example, if we use less-than when we should use less-than-or-equal, this can easily make our loop iterate the body the wrong number of times.

b) Tracing Variables

one good way to discover errors in a loop or any kind of code is to trace some key variables. Tracing variables means watching the variables change value while the program is running. Most programs do not output each variable's value every time the variable changes, but being able to see all of these variable changes can help us to debug your program.

c) General Debugging Techniques

Tracing errors can sometimes be a difficult and time-consuming task. It is not uncommon to spend more times debugging a piece of code then it took to write the code

Scanned with CamScanner

in the First Place. If we are having difficulties Finding the Source of our errors, then there are Some general debugging techniques to Consider. If the Program is giving incorrect Output Values, then we should examine the Source Code, different test Cases using a range of input and Output Values, and the logic behind the algorithm itself.

10. Explain each of the Following with relevant example Program: How Java Programming Provides the basic Object Oriented Features like encapsulation, abstraction, polymorphism and inheritance.

Ans Encapsulation in Java is a mechanism of wrapping the data (variables) and Code acting on the data (methods) together as a Single Unit. In encapsulation, the Variables of a Class will be hidden From other Classes, and Can be accessed only through the methods of their Current class. Therefore, it is also known as data hiding.

Eg:-

```
Public Class RunEncap {
Public static void main(String args[]){
```

```
EnCapTest enCap = new EnCapTest();
enCap.SetName("Huuchuu");
enCap.SetAge(25);
  enCap.SetIdNum("7777");
System.out.Println("Name:"+enCap.getName()+
"Age:"+enCap.getAge());
  }
  }
```

In Java, abstraction is achieved using Abstract Classes and interfaces.

```
Public abstract class Employee {
  Private String Name;
  Private String address;
  Private int number;

Public Employee(String name, String address, int
    number) {
System.out.Println("Constructing an Employee");
    this.name = name;
    this.address = address;
    this.number = number;
}
  Public double Computepay() {
    System.out.Println("Inside Employee Computepay");
```

Scanned with CamScanner

```
return 0.0;
}
Public Void mailcheck(){
  System.out.Println("Mailing a Check to" + this.
name + " " + this.address);
}
Public String toString(){
   return name + " " + address + " " + number;
}
Public String toString(){
 return name + " " + address + " " + number;
}
Public String getName(){
   return name;
}
Public String getAddress(){
 return address;
}
Public Void SetAddress(String newAddress){
   address = newAddress;
}
Public int getNumber(){
   return number;
}
}
```

Scanned with CamScanner

Any Java object that can pass more than one IS-A test is considered to be polymorphic. In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class object.

```
Public class Salary extends Employee {
  Private double Salary;

Public Salary (String name, String address, int number,
  double Salary){
    Super( name, address, number);
    Set_Salary (Salary);
    }
  Public Void mailcheck() {
System.out.Println("Within mailcheck of Salary Class");
System.out.Println("Mailing Check to" + getName() +
    " with Salary" + Salary);
  }
  Public double getSalary() {
    Return Salary;
}
Public Void SetSalary (double newSalary) {
    if (newSalary >= 0.0) {
      Salary = newSalary;
```

Scanned with CamScanner

```
}
}
Public double Computepay() {
System.out.Println(" Computing Salary Pay For "
getName());
    return Salary/52;
}
}
```

Inheritance Can be defined as the Process wh
one Class acquires the Properties (methods
and Fields) of another. With the use of
inheritance the information is made
manageable in a hierarchical order.

```
Class Calculation {
  int z;
Public Void addition (int x, int y) {
    Z = x+y;
System.out.Println ("The Sum of the given
Numbers : "+z);
  }
Public Void Substraction (int x, int y) {
    Z = x-y;
System.Out.Println (" The difference between
```

Scanned with CamScanner

```
the given numbers: "+z);
}
}
Public Class My_Calculation extends Calculations{
Public Void multiplication (int x, int y) {
z = x * y
System.out.Println ("The Product of the given
numbers: "+z);
}
Public Static Void main(String args[]) {
int a = 20, b = 10;
My_Calculation demo = new My_Calculation();
demo.addition (a,b);
demo.Substraction (a,b);
demo.multiplication (a,b);
}
}
```

11. What are the Significance of using interface in Java? Explain With Suitable example program.

Ans An interface is a reference type in Java. It is Similar to Class. It is a Collection of abstract methods. A Class implements an interface, thereby inheriting the abstract methods of

Scanned with CamScanner

the ~~inheritance~~ interface.

An interface is different from a class in several ways

- You cannot instantiate an interface.
- An interface does not contain any constructors.
- All of the methods in an interface are abstract.
- An interface cannot contain instance fields. The only fields that can appear in an interface must be declared both static and final
- An interface is not extended by a class; it is implemented by a class.
- An interface can extend multiple interfaces.

Eg:-

```
Public Class MammalInt implements Animal {
public Void eat () {
    System.Out.Println ("Mammal eats");
}
Public Void travel () {
    System.Out.Println("Mammal travels");
}
Public int noOfLegs () {
    return 0;
}
```

Scanned with CamScanner

```
Public Static Void main (String args[]) {
 MammalInt m = new MammalInt();
   m.eat();
   m.travel();
   }
 }
```

OUTPUT

Mammal eats

Mammal travels

12. Write a Program to implement following functionality of Bank account. Assume necessary member variable required. (Consider exception handling too)

Class Name : BankAccount

Member Methods:

Public BankAccount (long accountNumber): use Parameterized Constructor

Public Void deposit (double amount): deposit an amount

Public Void deposit (double amount): deposit an amount

Public Void withdraw (double amount): withdraw an amount (check balance before withdraw)

Public double getBalance (): Return an available

balance.
Public Void transfer (double amount, BankAccount targetAccount): Transfer amount to targetAccount.

Ans

```
Package TutorialTwo;
Class InSufficientBalanceException extends Exception{
  Private double amountDetail;
InSufficientBalanceException(double amountDetail){
  this.amountDetail = amountDetail;
}
}

Class BankAccount{
  Private long accountNumber;
  Private double totalAmount;
Public BankAccount(long accountNumber){
    this.accountNumber = accountNumber;
  }
Public Void deposit(double amount){
    totalAmount = totalAmount + amount;
  }
Public Void Withdraw(double amount) throws
  InSufficientBalanceException{
  if( amount > totalAmount)
    throw new InSufficientBalanceException(amount)
```

Scanned with CamScanner

```
else
    totalAmount = totalAmount - amount;
}
Public double getTotalAmount(){
    return totalAmount;
}
Public Void transfer(double amount, BankAccount
    targetAccount){
totalAmount = totalAmount - amount;
targetAccount.totalAmount = targetAccount.totalAmount
+ amount;
}
Public Void display(){
System.Out.Println("The total balance in the
account is" + getTotalAmount());
}
}

Public Class QuestionTwelve {
    Public Static Void main(String[] args){
BankAccount accountone = new BankAccount(123456
789);
    accountOne.deposit(10000);
BankAccount accountTwo = new BankAccount(987654
321);
    accountTwo.deposit(15000);
```

Scanned with CamScanner

```
try {
 accountOne.withdraw(5000);
} Catch (InsufficientBalanceException e) {
 System.out.Println("Insufficient balance");
}

accountTwo.transfer(5000, accountOne);
accountOne.display();
accountTwo.display();
}
}
```

13. Make a Java application with these Features

- An interface Called RegularPolygon with two abstract methods: getNumSides and getSideLength
- A Class EquilateralTriangle that implements the interface, has getNumSides return 3 and getSideLength return an instance Variable that is set by the Constructor.
- A Class Square that implements the interface, has getNumSides return 4 and getSideLength return an instance Variable that is set by the Constructor.

13.1 Add a Static totalSides method, that given a RegularPolygon[], returns the Sum of the number of sides of all the elements.

13.2 Add two default methods:

- getperimeter (n* length, where n is the number of sides)
- get InteriorAngle ((n-2)π/n in radians)

13.3 Make a few test cases.

Ans

```
Package tutorialTwo;
interface RegularPolygon {
    Public int getNumSide();
    Public int getSideLength();
    Public int getperimeter();
    Public int getInteriorAngle();
    Public static int totalSides(){
        return 4+3;
    }
}
Class EquilateralTriangle implements RegularPolygon {
 Private int SideLength;
EquilateralTriangle (int SideLength){
    this.SideLength = SideLength;
}
@override
Public int getNumSide(){
    return 3;
}
```

Scanned with CamScanner

```
@override
Public int getSideLength(){
    return SideLength;
}
@Override
Public int getperimeter(){
    return 3*getSideLength();
}
@override
 Public int getInteriorAngle(){
    return ((getNumSide()-2)*180)/getNum
 Side();
}
}
Class Square implements RegularPolygon{
 Private int SideLength;
 Square(int SideLength){
 this.SideLength = SideLength;
}
@Override
 Public int getNumSide(){
    return 4;
}
@override
Public int getSideLength(){
```

Scanned with CamScanner

```
return SideLength;
}
@override
Public int getperimeter (){
  return 4*getSideLength();
}
@override
Public int getInteriorAngle (){
  return ((getNumSide()-2)*180)/getNumSide();
}
}
Public Class QuestionThirteen {
Public Static Void main (String[] args){
Equilateral Triangle equilateralTriangle = new
  EquilateralTriangle(5);
System.out.Println("The perimeter of an
Equilateral triangle is "+equilateralTriangle.get
  Perimeter());
System.out.Println("The total number of Sides of
an Equilateral triangle is "+equilateral Triangle
.get ~~perimeter()~~ NumSide());
System.out.Println("The interior angle of an eq.
triangle is "+equilateralTriangle.getInterior
  Angle());
Square Square = new Square(4);
```

Scanned with CamScanner

```
System.Out.Println("The Perimeter of Square is"
  + Square.getPerimeter());
System.Out.Println("The total number of Sides
  of Square is" + Square.getNumSide());
System.Out.Println("The interior angle of Square
  is " + Square.getInteriorAngle());
System.Out.Println("The Sum of the total
    number of Sides of all elements is"
  + RegularPolygon.totalSides());
  }
}
```

Scanned with CamScanner