PBLJ MODEL PAPER SOLUTION

Q1. What is the significance of using this keyword in constructor?

Ans. The this keyword refers to the current object in a method or constructor. The most common use of the this keyword is to eliminate the confusion between class attributes and parameters with the same name (because a class attribute is shadowed by a method or constructor parameter).

Q2. List various test fixtures.

Ans. Test fixtures can be set up three different ways: in-line, delegate, and implicit. In-line setup creates the test fixture in the same method as the rest of the test. While in-line setup is the simplest test fixture to create, it leads to duplication when multiple tests require the same initial data.

Q3. What is the purpose of DTD in XML?

Ans. The purpose of a DTD is to define the legal building blocks of an XML document. It defines the document structure with a list of legal elements. A DTD can be declared inline in your XML document, or as an external reference.

Q4. List various implicit objects of JSP.

Ans. Request, response, out, session, application, config, pagecontext, page, exception.

Q5. Differentiate GenericServlet and HttpServlet.

Ans.

GenericServlet	HttpServlet
It is defined by javax.servlet package.	It is defined by javax.servlethttp package.
It describes protocol-independent servlet	It describes protocol-dependent servlet.
GenericServiet is not dependent on any particular protocol. It can be used with any protocol such as HTTP, SMTP, FTP, and so on.	HttpServlet is a dependent protocol and is only used with HTTP protocol.

GenericServlet	HttpServlet
All methods are concrete except the service() method. service() method is an abstract method.	All methods are concrete (non-abstract). service() is non-abstract method. service() can be replaced by doGet() or doPost() methods.
The service method is abstract.	The service method is non-abstract
It forwards and includes a request and is also possible to redirect a request.	It forwards and includes a request but it is not possible to redirect the request.
GenericServlet doesn't allow session management with cookies and HTTP sessions.	HTTPServlet allows session management with cookies and HTTP sessions.
It is an immediate child class of Servlet interface.	It is an immediate child class of GenericServlet class.
GenericServlet is a superclass of HttpServlet class.	HttpServlet is a subclass of GenericServlet class.

Q6. Write a program to differentiate between throw and throws keyword.

Ans.

THROW	THROWS
A throw is used to throw an exception explicitly	A throws to declare one or more exceptions, separated by commas.
Can throw a single exception using throw	Multiple can be thrown using Throws
This keyword is used in the method	Signature method is used with keyword throws
Only unchecked exceptions propagated using throw keyword.	To raise an exception throws keyword followed by the class name and checked exception can be propagated.
Throw keyword is followed by the instance variable	Throws keyword is followed by the exception class

Q7. What is thread synchronization. Write a program to study thread synchronization in java.

Ans. Synchronization in Java is the capability to control the access of multiple threads to any shared resource.

Java Synchronization is better option where we want to allow only one thread to access the shared resource.

```
class Table{
 synchronized void printTable(int n){
   for(int i=1;i<=5;i++){
     System.out.println(n*i);
     try{
      Thread.sleep(400);
     }catch(Exception e){System.out.println(e);}
   }

 }
}

class MyThread1 extends Thread{
Table t;
MyThread1(Table t){
this.t=t;
}
public void run(){
t.printTable(5);
}

}
class MyThread2 extends Thread{
Table t;
MyThread2(Table t){
this.t=t;
```

```
}
public void run(){
t.printTable(100);
}
}

public class TestSynchronization2{
public static void main(String args[]){
Table obj = new Table();
MyThread1 t1=new MyThread1(obj);
MyThread2 t2=new MyThread2(obj);
t1.start();
t2.start();
}
}
```

Q8. Explain various XML parsers and importance of parsing.

Ans.

i. **Dom Parser** − Parses an XML document by loading the complete contents of the document and creating its complete hierarchical tree in memory.
ii. **SAX Parser** − Parses an XML document on event-based triggers. Does not load the complete document into the memory.
iii. **JDOM Parser** − Parses an XML document in a similar fashion to DOM parser but in an easier way.
iv. **StAX Parser** − Parses an XML document in a similar fashion to SAX parser but in a more efficient way.
v. **XPath Parser** − Parses an XML document based on expression and is used extensively in conjunction with XSLT.
vi. **DOM4J Parser** − A java library to parse XML, XPath, and XSLT using Java Collections Framework. It provides support for DOM, SAX, and JAXP.

Q9. Differentiate Servlet and JSP and also write code for both.

Ans.

Servlet	JSP
Servlet is a java code.	JSP is a HTML based code.
Writing code for servlet is harder than JSP as it is HTML in java.	JSP is easy to code as it is java in HTML.
Servlet plays a controller role in the hasMVC approach.	JSP is the view in the MVC approach for showing output.
Servlet is faster than JSP.	JSP is slower than servlet because the first step in the hasJSP lifecycle is the translation of JSP to java code and then compile.
Servlet can accept all protocol requests.	JSP only accepts HTTP requests.
In servlet, we can override the service() method.	In JSP, we cannot override its service() method.

Q10. Create a class named 'Animal' which includes methods like eat() and sleep(). Create a child class of Animal named 'Bird' and override the parent class methods. Add a new method named fly(). Create an instance of Animal class and invoke the eat and sleep methods using this object. Create an instance of Bird class and invoke the eat, sleep and fly methods using this object.

Ans. class Animal {

```
public void eat()
   {
      System.out.println("eat method");
   }
   public void sleep()
   {
```

```
        System.out.println("sleep method");
    }

}
class Bird extends Animal{
    @Override
    public void eat() {
        super.eat();
        System.out.println("overide eat");
    }
    @Override
    public void sleep() {
        super.sleep();
        System.out.println("override sleep");
    }
    public void fly()
    {
        System.out.println("in fly method");
    }
}
class Animals{
    public static void main(String[] args) {
        Animal a =new Animal();
```

```
        Bird b = new Bird();
        a.eat();
        a.sleep();
        b.eat();
        b.sleep();
        b.fly();
    }
}
```

Q11. Explain various ArrayList class methods with the aid of program.

Ans. Java ArrayList add()

Java ArrayList addAll()

Java ArrayList clear()

Java ArrayList clone()

Java ArrayList contains()

Java ArrayList removeAll()

Java ArrayList remove()

Java ArrayList isEmpty()

Java ArrayList set()

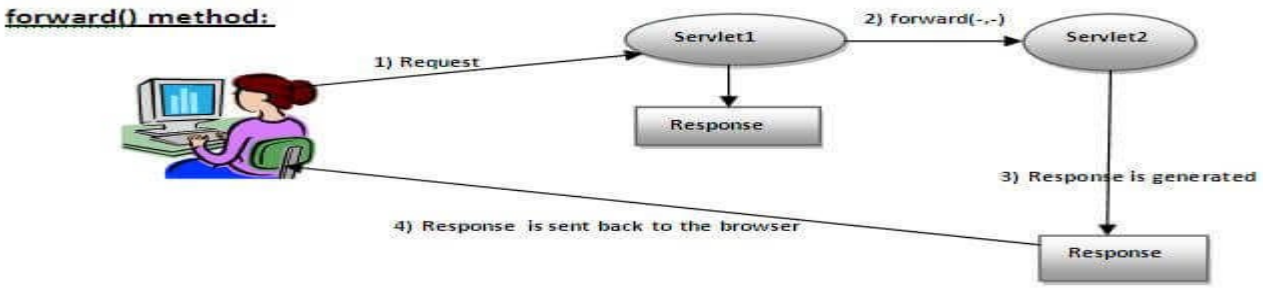
Java ArrayList sort().

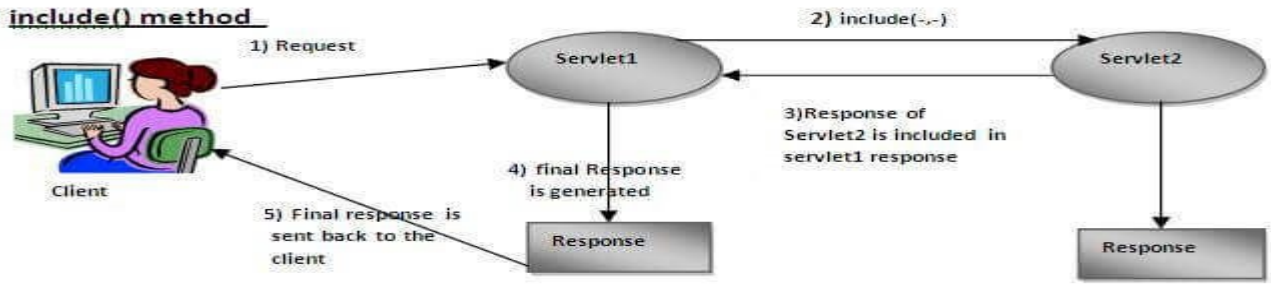
Q12. Explain Request Dispacher, its methods with appropriate diagrams. Also give example to illustrate difference between them.

Ans.

i. **public void forward(ServletRequest request,ServletResponse response)throws ServletException,java.io.IOException:**Forwards a request from a servlet to another resource (servlet, JSP file, or HTML file) on the server.

ii. **public void include(ServletRequest request,ServletResponse response)throws ServletException,java.io.IOException:**Includes the content of a resource (servlet, JSP page, or HTML file) in the response.





Q13. Differentiate private and protected access modifier?

Ans.

Private	Protected
The class members declared as private can be accessed only by the functions inside the class.	Protected access modifier is similar to that of private access modifiers.

Private members keep implementation details in a program.	Protected members enhanced access for derived classes.
Private members are not inherited in class.	Protected members are inherited in class.

Q14. What is difference between wait() and sleep() method?

Ans.

Wait()	Sleep()
Wait() method belongs to object class.	Sleep() method belongs to thread class.
Wait() method releases lock during synchronization.	Sleep() method does not release the lock on object during synchronization.
Wait() is not a static method.	Sleep() is a static method.
Wait() has 3 overloaded methods:- 1. wait() 2. Wait(long timeout) 3. Wait(long timeout, int nanos)	Sleep() has 2 overloaded methods:- 1. Sleep(long millis) millis: milliseconds 2. Sleep(long millis, int nanos) nanos: Nanoseconds

Q15. Difference between forward() method and sendRedirect() method ?

Ans.

forward() method	sendRedirect() method
The forward() method works at server side.	The sendRedirect() method works at client side.
It sends the same request and response objects to another servlet.	It always sends a new request.
It can work within the server only.	It can be used within and outside the server.
Example: request.getRequestDispacher("servlet 2").forward(request,response);	Example: response.sendRedirect("servlet2");

Q16. What is DOM Parser and its interfaces?

Ans. The DOMParser interface provides the ability to parse XML or HTML source code from a string into a DOM Document . You can perform the opposite operation—converting a DOM tree into XML or HTML source—using the XMLSerializer interface.

Q17. Recall the working of init() method of a servlet?

Ans. Called by the servlet container to indicate to a servlet that the servlet is being placed into service. The servlet container calls the init method exactly once after instantiating the servlet. The init method must complete successfully before the servlet can receive any requests.

Q18. Write program for default constructor, parametrized constructor. How we can access default and parameterized constructor of parent class write code to justify your answer.

Ans. Default Constructor:-

```
//Java Program to create and call a default constructor
class Bike1{
//creating a default constructor
Bike1(){System.out.println("Bike is created");}
//main method
public static void main(String args[]){
//calling a default constructor
Bike1 b=new Bike1();
}
}
```

Parametrized Constructor:-

```
//Java Program to demonstrate the use of the parameterized constructor.
class Student4{
    int id;
    String name;
```

```
    //creating a parameterized constructor
    Student4(int i,String n){
    id = i;
    name = n;
    }
    //method to display the values
    void display(){System.out.println(id+" "+name);}
    public static void main(String args[]){
    //creating objects and passing values
    Student4 s1 = new Student4(111,"Karan");
    Student4 s2 = new Student4(222,"Aryan");
    //calling method to display the values of object
    s1.display();
    s2.display();
   }
}
```

Q19. Write the steps to connect to the database in java. Write a program using prepared statement interface.

Ans. Java Database Connectivity with 5 Steps

i. Register the driver class.
ii. Create the connection object.
iii. Create the Statement object.
iv. Execute the query.
v. Close the connection object.

```
import java.sql.*;
class InsertPrepared{
public static void main(String args[]){
try{
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con=DriverManager.getConnection("jdbc:oracle:thin:@local
host:1521:xe","system","oracle");
```

```
PreparedStatement stmt=con.prepareStatement("insert into Emp values(?,
?)");
stmt.setInt(1,101);//1 specifies the first parameter in the query
stmt.setString(2,"Ratan");
int i=stmt.executeUpdate();
System.out.println(i+" records inserted");
con.close();
}
catch(Exception e){ System.out.println(e);}
  }
}
```

Q20. Elaborate XML, its role in web development. Explain the structure of XML program with suitable example.

Ans. XML stores data in plain text format. This provides a software- and hardware-independent way of storing, transporting, and sharing data. XML also makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

```
<!DOCTYPE website [
<!ELEMENT website (name,company,phone)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT company (#PCDATA)>
<!ELEMENT phone (#PCDATA)>
]>
<website>
<name>GeeksforGeeks</name>
<company>GeeksforGeeks</company>
<phone>011-24567981</phone>
</website>
```

Q21. Explain various scriplets tags in JSP with the help of program.

Ans.

Expression Tag:-

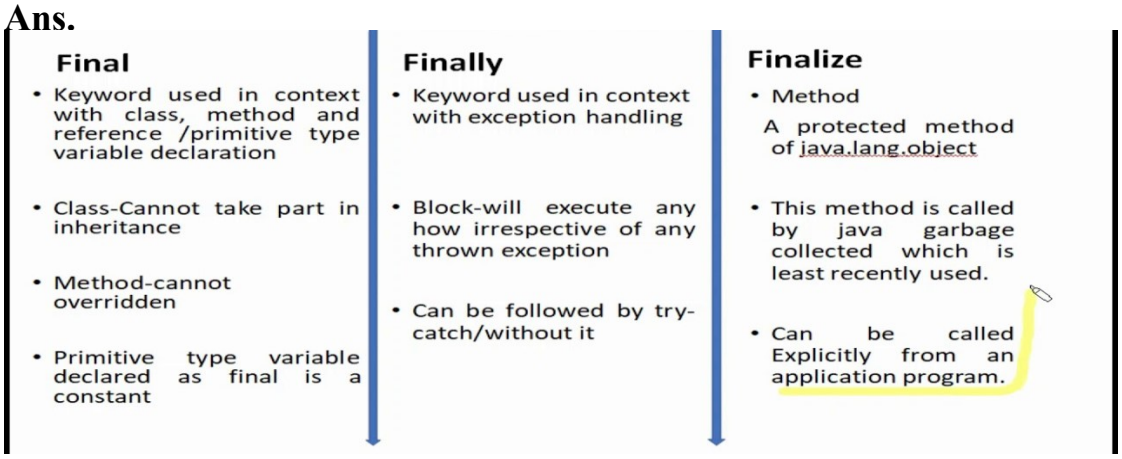
```
<html>
<body>
<form action="welcome.jsp">
<input type="text" name="uname"><br/>
<input type="submit" value="go">
</form>
</body>
</html>
```

Declaration Tag:-

```
<html>
<body>
<%!
int cube(int n){
return n*n*n*;
}
%>
<%= "Cube of 3 is:"+cube(3) %>
</body>
</html>
```

Q22. Distinguish between final, finally and finalize. Write code to differentiate between final, finally and finalize in Java?

Ans.



Final:-

```
public class FinalExampleTest {
   //declaring final variable
   final int age = 18;
   void display() {
   // reassigning value to age variable
   // gives compile time error
   age = 55;
   }
     public static void main(String[] args) {
     FinalExampleTest obj = new FinalExampleTest();
   // gives compile time error
   obj.display();
   }
}
```

Finally:-

```
public class FinallyExample {
     public static void main(String args[]){
     try {
       System.out.println("Inside try block");
     // below code throws divide by zero exception
      int data=25/0;
```

```
      System.out.println(data);
    }
    // handles the Arithmetic Exception / Divide by zero exception
    catch (ArithmeticException e){
      System.out.println("Exception handled");
      System.out.println(e);
    }
    // executes regardless of exception occurred or not
    finally {
      System.out.println("finally block is always executed");
    }
    System.out.println("rest of the code...");
    }
  }
```

Finalized:-

```
public class FinalizeExample {
    public static void main(String[] args)
    {
      FinalizeExample obj = new FinalizeExample();
      // printing the hashcode
      System.out.println("Hashcode is: " + obj.hashCode());
      obj = null;
      // calling the garbage collector using gc()
      System.gc();
      System.out.println("End of the garbage collection");
    }
  // defining the finalize method
    protected void finalize()
    {
      System.out.println("Called the finalize() method");
    }
```

```
}
```

Q23. Explain various inputStream classes. Write program for FileInputStream and FileOutputStream?

Ans. InputStream class is the superclass of all the io classes i.e. representing an input stream of bytes. It represents input stream of bytes. Applications that are defining subclass of InputStream must provide method, returning the next byte of input.

FileInputStream:-

```
  public static void main(String args[]) throws IOException {
    //Creating a File object
    File file = new File("D:/images/javafx.jpg");
    //Creating a FileInputStream object
    FileInputStream inputStream = new FileInputStream(file);
    //Creating a byte array
    byte bytes[] = new byte[(int) file.length()];
    //Reading data into the byte array
    int numOfBytes = inputStream.read(bytes);
    System.out.println("Data copied successfully...");
  }}
```

FileOutputStream:-

```
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
public class FileInputStreamExample {
  public static void main(String args[]) throws IOException {
    //Creating a File object
    File file = new File("D:/images/javafx.jpg");
    //Creating a FileInputStream object
```

```
    FileInputStream inputStream = new FileInputStream(file);
    //Creating a byte array
    byte bytes[] = new byte[(int) file.length()];
    //Reading data into the byte array
    int numOfBytes = inputStream.read(bytes);
    System.out.println("Data copied successfully...");
    //Creating a FileInputStream object
    FileOutputStream outputStream = new
FileOutputStream("D:/images/output.jpg");
    //Writing the contents of the Output Stream to a file
    outputStream.write(bytes);
    System.out.println("Data written successfully...");
  }
}
```

Q24. What is ServletConfig and ServletContext, its advantages. Differentiate between ServletConfig and ServletContext with the help of program.

Ans. The ServletConfig parameters are specified for a particular servlet and are unknown to other servlets. It is used for intializing purposes. The ServletContext parameters are specified for an entire application outside of any particular servlet and are available to all the servlets within that application.

```
<web-app>
    <servlet>
        <servlet-name>recruiter</servlet-name>
        <servlet-class>Recruiter</servlet-class>
        <init-param>
        <param-name>Email</param-name>
        <param-value>forRecruiter@xyz.com</param-value>
        </init-param>
</servlet>
```

```
    <servlet-mapping>
        <servlet-name>recruiter</servlet-name>
        <url-pattern>/servlet1</url-pattern>
</servlet-mapping>
    <servlet>
        <servlet-name>applicant</servlet-name>
        <servlet-class>Applicant</servlet-class>
        <init-param>
        <param-name>Email</param-name>
        <param-value>forApplicant@xyz.com</param-value>
        </init-param>
</servlet>
    <servlet-mapping>
        <servlet-name>applicant</servlet-name>
        <url-pattern>/servlet2</url-pattern>
</servlet-mapping>
    <context-param>
        <param-name>Website-name</param-name>
        <param-value>NewWebsite.tg</param-value>
</context-param>
    </web-app>
```

Q25. List 5 checked and unchecked exception.

Ans. 5 checked exception:-

- ClassNotFoundException
- InterruptedException
- InstantiationException
- IOException
- SQLException

5 unchecked exceptions:-

- ArithmeticException
- ClassCastException
- NullPointerException

- ArrayIndexOutOfBoundsException
- NegativeArraySizeException

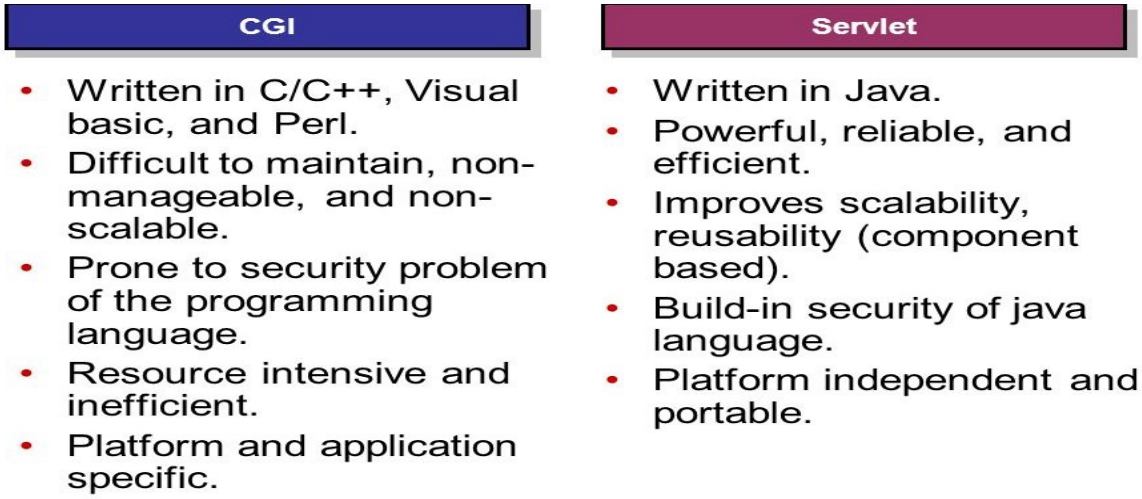
Q26. What is difference between user Thread and daemon Thread?

Ans.

Non-Daemon Threads (User Thread)	Daemon Thread
High priority threads.	Low priority threads.
Always run on foreground	Run on background
Designed to do some specific task.	Designed to support the user threads.
Created by the user.	Mostly created by the JVM (Ex: Garbage collector). User can also create Daemon Threads
JVM wait until user threads to finish their work. It never exit until all user threads finish their work.	If all user threads have finished their work JVM will force the daemon threads to terminate

Q27. Difference between servlet and CGI.

Ans.



Q28. Differentiate forward and include method.

Ans. The forward() method is used to transfer the client request to another resource (HTML file, servlet, jsp etc). When this method is called, the control is transferred to the next resource called. On the other hand, the include() method is used to include the content of the calling file into the called file.

Q29. What is RMI?

Ans. RMI (Remote Method Invocation) is a way that a programmer, using the Java programming language and development environment, can write object-oriented programming in which objects on different computers can interact in a distributed network.

Q30. How user defined exceptions are defined? When it is used? Write code to create user defined exception.

Ans. User-defined exceptions provide the flexibility to customize the exception as per our use case. A custom exception class must extend Exception class from java. lang package. Error message of a custom exception can be configured by passing message to super class constructor, or by overriding the toString() method.

```
import java.util.Scanner;

public class ExceptionExample {
```

```
  public static void main(String args[]) {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter first number: ");
    int a = sc.nextInt();
    System.out.println("Enter second number: ");
    int b = sc.nextInt();
    int c = a/b;
    System.out.println("The result is: "+c);
  }
}
```

Q31. What is Junit, its advantages?

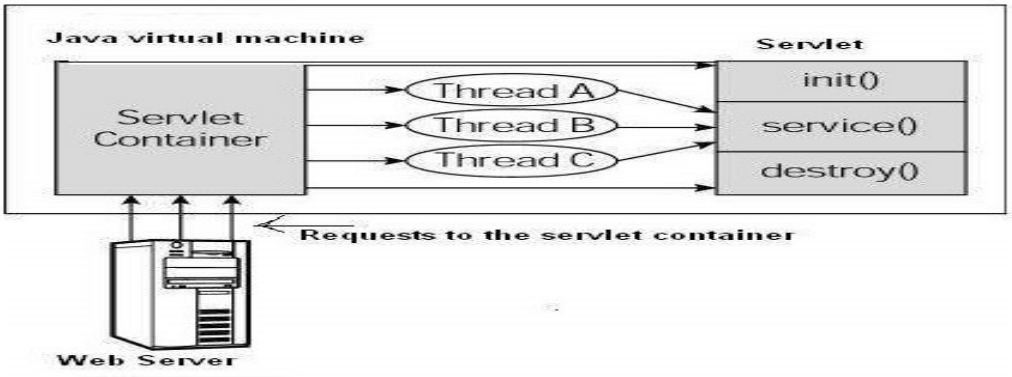
Ans. JUnit is a unit testing open-source framework for the Java programming language. Java Developers use this framework to write and execute automated tests. In Java, there are test cases that have to be re-executed every time a new code is added. This is done to make sure that nothing in the code is broken.

Advantages of JUnit are:-

- Unit is an open source framework, which is used for writing and running tests.
- Provides annotations to identify test methods.
- Provides assertions for testing expected results.
- Provides test runners for running tests.
- JUnit tests allow you to write codes faster, which increases quality.

Q32. Explain the servlet life cycle with diagram and explain its life cycle methods in detail.

Ans. A servlet life cycle can be defined as the entire process from its creation till the destruction. The following are the paths followed by a servlet. The servlet is initialized by calling the init() method. The servlet calls service() method to process a client's request.



Servlet class is loaded first when the Web container receives a new request. Then the web container creates an instance of the servlet. This instance is created only once in the whole life cycle of a servlet. The servlet is initialized by the calling init() method.

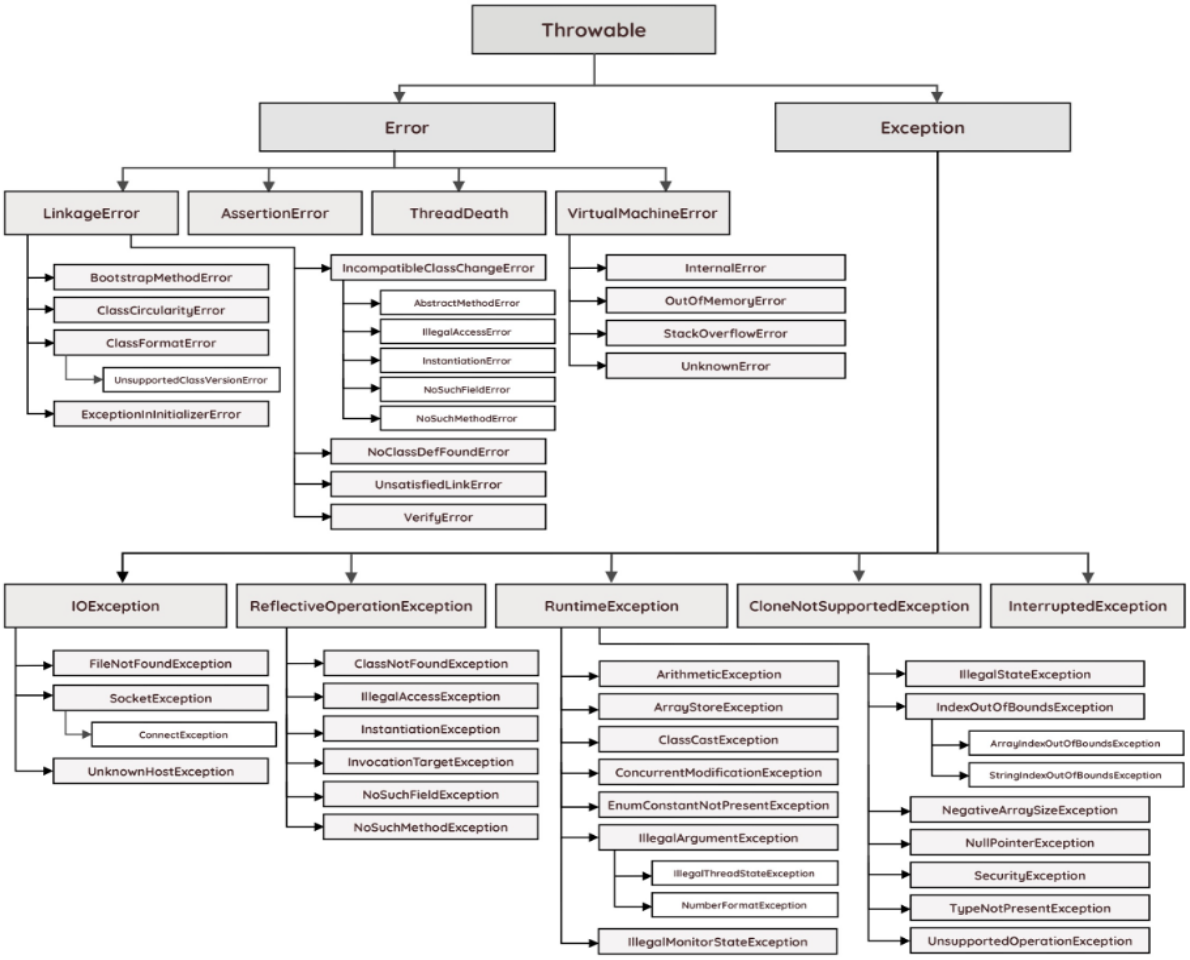
Q33. Explain JSTL and differentiate between SAX parser and DOM parser?

Ans. JSTL stands for JSP Standard Tag Library. JSTL is the standard tag library that provides tags to control the JSP page behavior. JSTL tags can be used for iteration and control statements, internationalization, SQL etc.

SAX	DOM
Simple API for XML	Document Object Model
SAX is an event driven parsing method.	DOM is tree based parsing method.
Useful for parsing large XML document.	It is useful for smaller applications.
Traversing is done in Top to Bottom approach.	Traversing can be done in any direction.
It requires less memory.	It requires more memory.
Event Driven API	Tree based API.

Q34. Draw Java Exception class Hierarchy and Write program to differentiate between number format exception and input mismatch exception?

Ans.



Number format exception:-

```
class Wrap
{
   public static void main(String...args)
   {
      Integer j=new Integer("s");
      System.out.println(j);
   }
}
```

Input mismatch exception:-

```
import java.util.Scanner;
class User
{
```

```
    public static void main(String...args)
    {
       Scanner obj=new Scanner(System.in);
       int i=obj.nextInt();
       int j=obj.nextInt();
       System.out.println("sum of numbers input by user");
       System.out.println(i+j);
    }
}
```

Q35. Distinguish between HashSet and HashMap. Write program to show the difference between HashSet and HashMap?

Ans.

Basic	**HashSet**	**HashMap**
Implements	Set interface	Map interface
Duplicates	No	Yes duplicates values are allowed but no duplicate key is allowed
Dummy values	Yes	No
Objects required during an add operation	1	2
Adding and storing mechanism	HashMap object	Hashing technique
Speed	It is comparatively slower than HashMap	It is comparatively faster than HashSet because of hashing technique has been used here.
Null	Have a single null value	Single null key and any number of null values
Insertion Method	Add()	Put()

HashSet

```
import java.util.*;
public class HashSetExample
{
public static void main(String args[])
{
//creating object of HashSet
HashSet<String> hs= new HashSet<String>();
//adding values to HashSet
hs.add("Java");
```

```
hs.add("Python");
hs.add("C++");
hs.add("C");
System.out.println("Before adding duplicate and null values: ");
System.out.println(hs);
//adding duplicate values
hs.add("Python");
hs.add("C");
System.out.println("After adding duplicate values: ");
System.out.println(hs);
//adding null values
hs.add(null);
hs.add(null);
System.out.println("After adding null values: ");
System.out.println(hs);
}
}
```

HashMap

```
import java.util.*;
public class HashMapExample
{
public static void main(String args[])
{
//creating object of HashMap
HashMap<String, Integer> hm= new HashMap<String, Integer>();
//adding key-value pair
hm.put("John", 23);
hm.put("Monty", 27 );
hm.put("Richard", 21);
hm.put("Devid", 19);
System.out.println("Before adding duplicate keys: ");
System.out.println(hm);
//adding duplicate keys
hm.put("Monty", 25);    //replace the Monty's previous age
hm.put("Devid", 19);
System.out.println("After adding duplicate keys: ");
System.out.println(hm);
}
}
```

Q36. What is the purpose of DTD in XML. How to use DTD in xml code.

Ans. The purpose of a DTD is to define the legal building blocks of an XML document. It defines the document structure with a list of legal elements. A DTD can be declared inline in your XML document, or as an external reference.

The working of DTD is performed by the following steps:

- First, create a DTD file for the respective XML Document.
- Next outline the structure of the document.
- Initiate with the root node which is the same as DOCTYPE.
- Include all the elements, attributes, entities for the file.