1. Write the difference between checked and unchecked exception with example code

Checked exceptions are exceptions that occur at the time of compiling the program. Unchecked exceptions are that exceptions that occur at the run time.

Eg for checked exception:

import java.io.\*;

class Main {

     public static void main(String[] args) {

         FileReader file = new FileReader("C:\\test\\a.txt");

         BufferedReader fileInput = new BufferedReader(file);

         // Print first 3 lines of file "C:\test\a.txt"0

         for (int counter = 0; counter < 3; counter++)

             System.out.println(fileInput.readLine());

         fileInput.close();

     }

}

Eg for unchecked exception:

import java.io.\*;

public class main {

public static void main(String args[]) {

int x = 0;

int y = 10;

int z = y/x;

}}

Output:

Exception in thread "main" java.lang.ArithmeticException: / by zero

2. Write the difference between throw and throws with example code  
 Throw is a keyword used in java exception to explicitly thow an exception instance (object). This keyword 'throw' can be used in both checked (compile time) exception and unchecked (runtime) exception.

In java, throws keyword is used to declare an exception and throw it to the calling environment (throw it to the code unit from where method is getting called for execution at runtime). This keyword should only be used at runtime (unchecked).

Eg for throw:

Void A(){

Try{

Throw new ArthmeticException(“In a/b, b cannot be 0”);

}

Catch(Exception e)

{

System.out.println(“e.getMessage()”);

}

}

Eg for throws:

Void A()throws ArthmeticException{

//Statement

}

3. Write a note or nested try…catch block with example code

When try block is used then catch block must be implemented. The try-catch block inside a nested try block.

Eg:

Class A{

Public static void main(String[] Args)

{

try{

try{

int array[]={1,2,3,4};

System.out.println(“array[10]”);

}

Catch(ArithmeticException e)

{ System.out.println(“ArithmeticException”);

}

Catch(ArrayIndexOutofBoundException e){

System.out.println(“ArrayIndexOutofBound”)

}

}

}

4. Write a note on MultiThreading and MultiTasking  
 MultiTasking refers to any task taking place in CPU in asynchronous execution. It can refer to multiple threads, multiple programs and multiple processes.

MultiThreading is specific to multiple thread execution by time-sharing in both synchronous and asynchronous logic.

5. Write a short note on Deque and give example code.

Deque is a double ended queue from where we can enter and remove the elements from first and last i.e both the ends.  
Eg

import java.util.\*;

public class ArrayDequeExample {

    public static void main(String[] args) {

    Deque<String> deque = **new** ArrayDeque<String>();

    deque.add("Rini");

    deque.add("Vini");

    deque.add("Mini");

**for** (String str : deque) {

   System.out.println(str);

    }

    }

}

6. Write a short note on Generics an all types of Parameters used in Generics with example code.  
 Generics are used in collections and in generics we can specify the specific datatype to the variables that are initialized (i.e, which type of variable to be inserted).

Eg

public class test{

public static < E > void printArray(E[] elements) {

for ( E element : elements){

System.out.println(element );

}

System.out.println();

}

public static void main( String args[] ) {

Integer[] intArray = { 10, 20, 30, 40, 50 };

Character[] charArray = { 'A', 'B', 'C', 'D' };

System.out.println( "Printing Integer Array" );

printArray( intArray );

System.out.println( "Printing Character Array" );

printArray( charArray );

}

}

7. Write a short note on Map Interface.

A map contains values on the basis of key i.e. key and value pair. Each key and value pair is known as an entry. Map contains only unique keys.Map is useful if you have to search, update or delete elements on the basis of key.

8. Write the difference between LinkedList and ArrayList.

LinkedList internally uses **doubly linked list**to store the elements. Manipulation with LinkedList is faster than ArrayList because it uses doubly linked list so no bit shifting is required in memory. LinkedList class can **act as a list and queue** both because it implements List and Deque interfaces. LinkedList is better for manipulating data.

ArrayList internally uses **dynamic array** to store the elements. Manipulation with ArrayList is slow because it internally uses array. If any element is removed from the array, all the bits are shifted in memory. ArrayList class can act as a **list** only because it implements List only. ArrayList is better for storing and accessing data

9. Write a note on Dynamic array in java.

Array lists are called as dynamic arrays in java.