**Day 2: Assignments**

**1.Write 3 different java programs to print the following patterns**

**a) 1**

**12**

**123**

**12345**

Solution :-

**public** **class** Q1\_Part1 {

**public** **static** **void** main(String[] args) {

**for**(**int** i=1;i<=5;i++){

**for**(**int** j=1;j<=i;j++){

System.***out***.print(j);

}

System.***out***.println();

}

}

}

**b) 54321**

**5432**

**543**

**54**

**5**

Solution :-

**public** **class** Q1\_Part2 {

**public** **static** **void** main(String[] args){

**for**(**int** i=1;i<=5;i++){

**for**(**int** j=5;j>=i;j--){

System.***out***.print(j);

}

System.***out***.println();

}

}

}

**c)     x**

**xxx**

**xxxxx**

**xxxxxxx**

**xxxxx**

**xxx**

**x**

              Note: Shape will be Rhombus.

 Solution :-

**public** **class** Q1\_Part3 {

**public** **static** **void** main(String[] args){

**int** rows=10;

**int** space=rows-1;

**for**(**int** i=1;i<=rows;i++){

**for**(**int** j=1;j<=space;j++){

System.***out***.print(" ");

}

space=space-1;

**for**(**int** k=1;k<=2\*i-1;k++){

System.***out***.print("x");

}

System.***out***.println();

}//outer loop end

**for**(**int** i=1;i<=rows-1;i++){

**for**(**int** j=1;j<=i;j++){

System.***out***.print(" ");

}

**for**(**int** k=1;k<=2\*(rows-i)-1;k++){

System.***out***.print("x");

}

System.***out***.println();

}

}

}

**2. Write a java program to take the input from user and determine if it is a prime number or not.**

Solution :-

**import** java.util.Scanner;

**public** **class** Q2 {

**public** **static** **void** main(String[] args){

Scanner sc=**new** Scanner(System.***in***);

System.***out***.println("Enter a number : ");

**int** num=sc.nextInt();

**int** flag=0;

**for**(**int** i=2;i<num/2;i++){

**if**(num%i==0){

flag=1;

**break**;

}

}

**if**(flag==1){

System.***out***.println(num+" is not a prime number");

}

**else** {

System.***out***.println(num+" is a prime number");

}

}

}

**3. Write a java program to display the fibonacci series till less than 200 using only 2 variables.**

Solution :-

**public** **class** Q3\_WithRecursion {

**static** **int** fib(**int** term){

**if**((term==1)||(term==2)){

**return** 1;

}

**else** {

**return** *fib*(term-1)+*fib*(term-2);

}

}

**public** **static** **void** main(String[] args) {

**for**(**int** i=1;i<=200;i++){

System.***out***.print(*fib*(i)+"+");

}

}

}

**4.Write Java program to check if a name is palindrome.**

Solution :-

**public** **class** Q4 {

**public** **static** **void** main(String[] args){

Scanner sc=**new** Scanner(System.***in***);

System.***out***.println("Enter a string");

String str=sc.nextLine();

String rev="";

**for**(**int** i=str.length()-1;i>=0;i--){

rev=rev+str.charAt(i);

}

**if**(str.equals(rev)){

System.***out***.println("String is palindrome");

}

**else** {

System.***out***.println("String is not palindrome...");

}

}

}

**5.Write Java program to check if a number is Armstrong number or not? (input 153 output true,  123 output false)**

 Solution :-

**import** java.util.Scanner;

**public** **class** Q5 {

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

System.***out***.println("Enter a number");

**int** num=sc.nextInt(); //153

**int** temp=num;

**int** r;

**int** cube=0;

**while**(num>0){

r=num%10;

num=num/10;

cube=cube+(r\*r\*r);

}

**if**(temp==cube){

System.***out***.println("Given no. "+temp+" is an armstrong number");

}

**else** {

System.***out***.println(temp+" is not an armstrong number");

}

}

}

**6.How to find factorial of number in Java using iteration?**

Solution :-

**import** java.util.Scanner;

**public** **class** Q6 {

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

System.***out***.println("Enter any number : ");

**int** num=sc.nextInt();

**int** fact=1;

**for**(**int** i=num;i>=1;i--){

fact=fact\*i;

}

System.***out***.println("Factorial of "+num+" is : "+fact);

}

}

**7.Write a Java code to take a character as a input from user and determine if it is a vowel or a consonant using conditional construct.**

Solution :-

**import** java.util.Scanner;

**public** **class** Q7 {

**public** **static** **void** main(String[] args){

Scanner sc=**new** Scanner(System.***in***);

System.***out***.println("Enter any character : ");

**char** ch=sc.next().charAt(0);

**char** ch2=String.*valueOf*(ch).toLowerCase().charAt(0);

**if**((ch2=='a')||(ch2=='e')||(ch2=='i')||(ch2=='o')||(ch2=='u')){

System.***out***.println("Entered Character is a vowel");

}

**else** {

**if**((ch2>='a')&&(ch2<='z')){

System.***out***.println("Entered character '"+ch+"' is a consonant");

}

**else** {

System.***out***.println("Not an alphabet");

}

}

}

}

**8. Write a switch case java code to create calculator with + - / \* functionalities only.**

Solution :-

**import** java.util.Scanner;

**public** **class** Q8 {

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

System.***out***.println("Enter the first number : ");

**int** firstNumber=sc.nextInt();

System.***out***.println("Enter the second number : ");

**int** secondNumber=sc.nextInt();

System.***out***.println("Enter for choice for the operation you want to perform (+,-,\*,/)");

String choice=sc.next();

**switch**(choice){

**case** "+" : System.***out***.println("Addition = "+(firstNumber+secondNumber));

**break**;

**case** "-" : System.***out***.println("Subtraction = "+(firstNumber-secondNumber));

**break**;

**case** "\*" : System.***out***.println("Multiplication = "+(firstNumber\*secondNumber));

**break**;

**case** "/" : System.***out***.println("Division = "+(firstNumber/secondNumber));

**break**;

**default**: System.***out***.println("Invalid Choice....");

}

}

}

**9. Write a java code to copy one array into another.**

Solution :-

**public** **class** Q9 {

**public** **static** **void** main(String[] args) {

**int**[] arr={23,1,98,8,12};

**int**[] arr2=**new** **int**[arr.length];

System.*arraycopy*(arr, 0, arr2, 0, arr.length);

System.***out***.println("======Array Copy=======");

**for**(**int** i:arr2){

System.***out***.println("Element : "+i);

}

}

}

**10. Write a java code to compare the length of two arrays and display the longer array.**

Solution :-

**public** **class** Q10 {

**public** **static** **void** main(String[] args) {

String[] arr1={"Ram","Sita","Luv","Kush"};

String[] arr2={"Radha","Krishna"};

**if**(arr1.length>arr2.length){

**for**(String s:arr1){

System.***out***.println(s);

}

}

**else** {

**for**(String s:arr2){

System.***out***.println(s);

}

}

}

}

**11. Write a java code to display a reverse String array.**

Solution :-

**public** **class** Q11 {

**public** **static** **void** main(String[] args) {

String[] books={"Complete Reference","OCJP Preparation","Black Book","Effective Java","Thinking in Java"};

String temp="";

**for**(**int** i=0,j=(books.length-1);i<=books.length/2;i++,j--){

temp=books[i];

books[i]=books[j];

books[j]=temp;

}

System.***out***.println("=====Reversed Array is==========");

**for**(String book:books){

System.***out***.println(book);

}

}

}

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

Ans : In Java , exceptions can be categorized into two types :- Checked Exceptions and Unchecked exceptions which are as follows :-

1. Checked Exceptions :- Checked Exceptions are the exceptions which are known to the Compiler . e.g. ClassNotFoundException , SQLException , FileNotFoundException , IOException etc. All Checked Exceptions are the child classes of java.lang.Exception class.
2. Unchecked Exception : - Unchecked exceptions are the exceptions which are unknown to the compiler . All the unchecked exceptions are the child of RuntimeException class and they itself are known as runtime exceptions . e.g. ArithmeticException , NullPointerException , ArrayIndexOutOfBoundsException etc.

**13.   Write the difference between throw and throws with example code**

**Ans :-** throw :- throw keyword is used to throw an exception explicity . It is mainly used when we want to throw the object of user-defined exception classes in java.

throws : throws keyword is used when we want to propogate the exception to the caller of the method . Throws keyword is used in the signature of the method.

e.g.

**import** java.util.Scanner;

**class** VotingException **extends** Exception {

**public** VotingException() {

}

**public** VotingException(String msg){

**super**(msg);

}

**public** String toString(){

**return** **super**.toString();

}

}

**class** VotingCheck {

**public** **void** checkEligibility(**int** age,String nationality) **throws** VotingException{

**if**((age>=18)&&(nationality.equals("Indian"))){

System.***out***.println("Eligible for Voting");

}

**else** {

**throw** **new** VotingException("Not Eligible for voting. Ur age must be >= 18 and nationality must be indian");

}

}

}

**public** **class** ThrowThrowsDemo {

**public** **static** **void** main(String[] args) {

Scanner sc=**new** Scanner(System.***in***);

VotingCheck obj=**new** VotingCheck();

System.***out***.println("Enter ur age : ");

**int** a=sc.nextInt();

System.***out***.println("Enter ur nationality : ");

String nationality=sc.next();

**try** {

obj.checkEligibility(a, nationality);

}

**catch**(VotingException e){

e.printStackTrace();

}

}

}

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

Ans :- When a try catch block is present in another try block then it is called the nested try catch block . Each time a try doesn’t have a catch handler for a particular exception , then the catch blocks of parent try block are inspected for that exception , if match is found that the catch block executes.

If neither catch block nor parent catch block handles exception then the system generated message would be shown for the exception .

**public** **class** NestedTryCatchExample {

**public** **static** **void** main(String[] args){

//parent try block

**try**{

//child try block 1

**try**{

System.***out***.println("Inside block 1");

**int** b=45/0;

System.***out***.println(b);

}

**catch**(ArithmeticException arithmeticException){

System.***out***.println("Exception 1");

}

//child try block 2

**try**{

System.***out***.println("Inside block 2");

**int** b=45/0;

System.***out***.println(b);

}

**catch**(ArrayIndexOutOfBoundsException ex){

System.***out***.println("Exception 2");

}

System.***out***.println("Just other statement");

}

**catch**(ArithmeticException e1){

System.***out***.println("ArithmeticException");

System.***out***.println("Inside parent catch");

}

**catch**(ArrayIndexOutOfBoundsException e2){

System.***out***.println("ArrayIndexOutOfBoundsException");

System.***out***.println("Inside parent catch");

}

**catch**(Exception e){

System.***out***.println("Exception ");

System.***out***.println("Inside parent try catch block");

}

System.***out***.println("Next statement...");

}

}

**15.   Write a note on MultiThreading and MultiTasking**

Ans : - Multitasking – The process of executing more than one task is called Multitasking . Multitasking can be categorized into two types:-

1. Process Based Multitasking :- The process of executing more than one process (any program in execution mode is called process) at a time is called Process based Multitasking . It is also be called as Multiprocessing. E.g. We are writing assignments in Ms-Word , at the same time we are listening songs on VLC Media player as well as we are using Google browser. Here we are working with 3processes at a time so it is called process based multitasking.
2. Thread Based Multitasking : The process of executing more than one thread (A smallest individual unit of a process who is responsible for its execution) in a process is called Thread based multitasking . Thread based multitasking is also known as Multithreading. E.g. In Ms – World , one thread is responsible for writing , one thread is responsible for checking spellings , one thread is responsible for checking grammer mistake. So this is an example of Multithreading.

So , we can say , Multithreading is a type of Multitasking.

**16.   Write a short note on Deque and give example code.**

Ans : The java.util.Deque interface is a subtype of the [java.util.Queue](https://www.geeksforgeeks.org/queue-interface-java/) interface. The Deque is related to the double-ended queue that supports addition or removal of elements from either end of the data structure, it can be used as a [queue (first-in-first-out/FIFO)](https://www.geeksforgeeks.org/queue/) or as a [stack (last-in-first-out/LIFO)](https://www.geeksforgeeks.org/stack/).

public class DequeExample

{

public static void main(String[] args)

{

Deque deque = new LinkedList<>();

// We can add elements to the queue in various ways

deque.add("Element 1 (Tail)"); // add to tail

deque.addFirst("Element 2 (Head)");

deque.addLast("Element 3 (Tail)");

deque.push("Element 4 (Head)"); //add to head

deque.offer("Element 5 (Tail)");

deque.offerFirst("Element 6 (Head)");

deque.offerLast("Element 7 (Tail)");

System.out.println(deque + "\n");

// Iterate through the queue elements.

System.out.println("Standard Iterator");

Iterator iterator = deque.iterator();

while (iterator.hasNext())

System.out.println("\t" + iterator.next());

// Reverse order iterator

Iterator reverse = deque.descendingIterator();

System.out.println("Reverse Iterator");

while (reverse.hasNext())

System.out.println("\t" + reverse.next());

// Peek returns the head, without deleting

// it from the deque

System.out.println("Peek " + deque.peek());

System.out.println("After peek: " + deque);

// Pop returns the head, and removes it from

// the deque

System.out.println("Pop " + deque.pop());

System.out.println("After pop: " + deque);

// We can check if a specific element exists

// in the deque

System.out.println("Contains element 3: " +

deque.contains("Element 3 (Tail)"));

// We can remove the first / last element.

deque.removeFirst();

deque.removeLast();

System.out.println("Deque after removing " +

"first and last: " + deque);

}

}

**17.   Write a short note on Generics an all types of Parameters used in Generics with example code.**Ans : Generics in Java is similar to templates in C++. The idea is to allow type (Integer, String, … etc and user defined types) to be a parameter to methods, classes and interfaces. For example, classes like HashSet, ArrayList, HashMap, etc use generics very well. We can use them for any type.

Generic Class

Like C++, we use <> to specify parameter types in generic class creation. To create objects of generic class, we use following syntax.

// To create an instance of generic class

BaseType <Type> obj = new BaseType <Type>()

Note: In Parameter type we can not use primitives like

'int','char' or 'double'.

**18.   Write a short note on Map Interface.**

Ans : The java.util.Map interface represents a mapping between a key and a value. The Map interface is not a subtype of the [Collection](https://www.geeksforgeeks.org/collections-in-java-2/)interface. Therefore it behaves a bit different from the rest of the collection types.

A Map cannot contain duplicate keys and each key can map to at most one value. Some implementations allow null key and null value ([HashMap](https://www.geeksforgeeks.org/hashmap-treemap-java" \t "_blank)and [LinkedHashMap](https://www.geeksforgeeks.org/linkedhashmap-class-java-examples" \t "_blank)) but some do not ([TreeMap](https://www.geeksforgeeks.org/hashmap-treemap-java" \t "_blank)).

The order of a map depends on specific implementations, e.g [TreeMap](https://www.geeksforgeeks.org/hashmap-treemap-java" \t "_blank)and [LinkedHashMap](https://www.geeksforgeeks.org/linkedhashmap-class-java-examples" \t "_blank)have predictable order, while [HashMap](https://www.geeksforgeeks.org/hashmap-treemap-java" \t "_blank)does not.  
Exampled class that implements this interface is HashMap, TreeMap and LinkedHashMap.

Why and When Use Maps:  
Maps are perfectly for key-value association mapping such as dictionaries. Use Maps when you want to retrieve and update elements by keys, or perform lookups by keys. Some examples:

A map of error codes and their descriptions.

A map of zip codes and cities.

A map of managers and employees. Each manager (key) is associated with a list of employees (value) he manages.

A map of classes and students. Each class (key) is associated with a list of students (value).

**19.   Write the difference between LinkedList and ArrayList.**

Ans :-

LinkedList :-

1. LinkedList will create doubly linked list in memory.
2. LinkedList is best suitable when we want to do frequent insertion and deletion.
3. Not suitable when we want to do random access.
4. LinkedList implements java.util.List and java.util.Deque interface.
5. LinkedList can be iterated either in backward or in forward direction.

ArrayList :-

1. ArrayList will create dynamic array in memory.
2. It is best suitable when we want to do Random Access.
3. ArrayList is not suitable for frequent insertion and deletion.
4. ArrayList can be iterated only in forward direction.
5. ArrayList implements List and RandomAccess interface.

**20.   Write a note on Dynamic array in java.**

Ans :

When the size of an array is unknown at the run time we need to create dynamic array. In java there

are plenty of ways in which we can build array dynamically . In that, i am choosing one of the easy way.

It’s through ‘Arraylist’. Here we will take a tour from ‘How to create an Arraylist’ to ‘How to use

arraylist’.

Syntax for creating an arraylist is

|  |  |
| --- | --- |
| 1 | ArrayList arraylist\_name = new ArrayList(); |

Here is a simple code which creates an arraylist named fruits and adds fruit names to it.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | import java.util.ArrayList;  public class fruits {        public static void main(String[] args) {            ArrayList fruits = new ArrayList(); //creating a new arraylist          fruits.add("apple"); // adding fruit names to it          fruits.add("orange");          fruits.add("mango");      }  } |

**21.   What is the purpose of the System class?**

Ans : 1. System class is provided with useful fields (static members) pertaining to the environment.  
  
2. Standard input,output and error output streams are provided with System class. These are used to access the externally defined properties and environment variables.

Example :-

System.in – external property for input device

System.out – external property for output device.

**22.   Which is the abstract parent class of FileWriter ?**

Ans : java.io.Writer is the abstract parent class of FileWriter ?

**23.   Which class is used to read streams of characters from a file?**

Ans : FileReader class is used to read streams of characters from a file?

**24.   Which class is used to read streams of raw bytes from a file?**

Ans : FileWriter class is used to read streams of raw bytes from a file?

**25.   What are the differences between FileInputStream/FileOutputStream and RandomAccessFile**

Ans : RandomAccessFile treats the file as an array of bytes where it has the internal pointer. The fact that it treats it like a large array of bytes is what is unique about this class. FileInputStream however just reads the stream and returns the data. It is more suited to reading raw data like images etc. It does not treat the file as a large array, it just keeps tabs of where in the file it has read so far. With FileInputStream you would actually have to read the data and place it into an array to get the same style of access as RandomAccessFile.

**26.   Write a note on Channels and Buffer with example.**

Ans : Buffers provide a mechanism to store a fixed amount of primitive data elements in an in-memory container. In the NIO, all data is handled with buffers. When data is read, it is read directly into a buffer. When data is written, it is written into a buffer.

Buffers work with channels. Channels are portals through which I/O transfers take place, and buffers are the sources or targets of those data transfers.

import java.io.file.Paths;

import java.nio.file.Path;

import java.io.IOException;

import java.nio.ByteBuffer;

import java.nio.channels.FileChannel;

import java.io.FileReader;

import java.io.BufferedReader;

public class FileChannelRead {

public static void main (String [] args)

throws Exception {

new FileChannelRead().readFile();

}

private void readFile()

throws IOException {

String filePath = "readfile.txt";

printFileContents(filePath);

Path path = Paths.get(filePath);

FileChannel fileChannel = FileChannel.open(path);

ByteBuffer buffer = ByteBuffer.allocate(6);

int noOfBytesRead = fileChannel.read(buffer);

while (noOfBytesRead != -1) {

System.out.println("Number of bytes read: " + noOfBytesRead);

buffer.flip();

System.out.print("Buffer contents: ");

while (buffer.hasRemaining()) {

System.out.print((char) buffer.get());

}

System.out.println(" ");

buffer.clear();

noOfBytesRead = fileChannel.read(buffer);

}

fileChannel.close();

}

private void printFileContents(String path)

throws IOException {

FileReader fr = new FileReader(path);

BufferedReader br = new BufferedReader(fr);

String textRead = br.readLine();

System.out.println("File contents: ");

while (textRead != null) {

System.out.println(" " + textRead);

textRead = br.readLine();

}

fr.close();

br.close();

}

}

**34.   Write a note on PreparedStatement and ResultSetMetaData interfaces with code snippets.**

# Ans : PreparedStatement interface

The PreparedStatement interface is a subinterface of Statement. It is used to execute parameterized query.

Let's see the example of parameterized query:

1. String sql="insert into emp values(?,?,?)";

As you can see, we are passing parameter (?) for the values. Its value will be set by calling the setter methods of PreparedStatement.

Why use PreparedStatement?

Improves performance: The performance of the application will be faster if you use PreparedStatement interface because query is compiled only once.

How to get the instance of PreparedStatement?

The prepareStatement() method of Connection interface is used to return the object of PreparedStatement. Syntax:

1. public PreparedStatement prepareStatement(String query)throws SQLException{}

e.g.

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:@localhost: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);}

}

}

**ResultSetMetaData :-**

The metadata means data about data i.e. we can get further information from the data.

If you have to get metadata of a table like total number of column, column name, column type etc. , ResultSetMetaData interface is useful because it provides methods to get metadata from the ResultSet object.

import java.sql.\*;

class Rsmd{

public static void main(String args[]){

try{

Class.forName("oracle.jdbc.driver.OracleDriver");

Connection con=DriverManager.getConnection(

"jdbc:oracle:thin:@localhost:1521:xe","system","oracle");

PreparedStatement ps=con.prepareStatement("select \* from emp");

ResultSet rs=ps.executeQuery();

ResultSetMetaData rsmd=rs.getMetaData();

System.out.println("Total columns: "+rsmd.getColumnCount());

System.out.println("Column Name of 1st column: "+rsmd.getColumnName(1));

System.out.println("Column Type Name of 1st column: "+rsmd.getColumnTypeName(1));

con.close();

}catch(Exception e){ System.out.println(e);}

}

}

**35.   Write a note on DDL, DML, DQL, DDL with code snippets.**

Ans : **Data Definition Language (DDL)**  
  
The commands of SQL that are used to create database objects, alter the structure of the database objects and delete database objects from database are collectively called as DDL. Examples include Create, Alter , Drop, Truncate, Rename and Comment Commands.

Create  
  
Create command is used to create database and its Objects like tables, index, stored procedure, views , triggers, functions and etc.  
  
Example  
  
To create Employee table.

create table tblEmployee(

   Id int primary key identity(1,1) not null,

   Name nvarchar(50) ,

   Gender nvarchar(50) ,

   Salary int ,

   DepartmentId int ,

)

Alter  
  
Alter command is used to create database and its Objects.

Drop  
  
Drop command is used to delete objects from database.  
  
Truncate  
  
Trunctae Table command is used to remove all records from a table, including all spaces allocated for records are removed.  
  
Rename  
  
It is used to rename the objects.  
  
Comment  
  
// -> Single line Comments, /\* --Multi Line Comments-- \*/ used to comment the sql statements.   
  
Data Manipulation Language (DML)  
  
The commands of SQL that are used to insert data into the database, modify the data of the database and to delete data from the database are collectively called as DML. Examples include Insert, Update and Delete.  
  
Insert

To insert date into a table.  
  
**Update**  
  
To update the existing data in a table.  
  
**Delete**  
  
delete all records from a table.   
  
**Data Query Language (DQL)**  
  
The commands of SQL that are used to retrieve data from the database are collectively called as DQL. So all Select statements comes under DQL.  
  
**Select**  
  
To retreive data from the database table. 

**36.   Write a note on HTML , CSS and Javascript.**

Ans :

HTML :-

* HTML is the standard markup language for creating Web pages.
* HTML stands for Hyper Text Markup Language
* HTML describes the structure of Web pages using markup
* HTML elements are the building blocks of HTML pages
* HTML elements are represented by tags
* HTML tags label pieces of content such as "heading", "paragraph", "table", and so on
* Browsers do not display the HTML tags, but use them to render the content of the page

A simple HTML Document :-

* <!DOCTYPE html>  
  <html>  
  <head>  
  <title>Page Title</title>  
  </head>  
  <body>  
    
  <h1>My First Heading</h1>  
  <p>My first paragraph.</p>  
    
  </body>  
  </html>

CSS :-

* CSS stands for Cascading Style Sheets
* CSS describes how HTML elements are to be displayed on screen, paper, or in other media
* CSS saves a lot of work. It can control the layout of multiple web pages all at once
* External stylesheets are stored in CSS files
* CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

CSS and HTML :-

* HTML was NEVER intended to contain tags for formatting a web page!
* HTML was created to describe the content of a web page, like:
* <h1>This is a heading</h1>
* <p>This is a paragraph.</p>
* When tags like <font>, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large websites, where fonts and color information were added to every single page, became a long and expensive process.
* To solve this problem, the World Wide Web Consortium (W3C) created CSS.
* CSS removed the style formatting from the HTML page!

Javascript:-

* JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform.

<html>

<body>

<script language="javascript" type="text/javascript">

<!--

document.write("Hello World!")

//-->

</script>

</body>

</html>

**37.   Write a code to fetch the data from H2 and put it in any collection object and display it.**

Ans : **public** **class** JDBCDemo1 {

**public** **static** **void** main(String[] args) {

String url="jdbc:h2:tcp://localhost/~/Project";

String username="sa";

String password="";

String query="select \* from Student";

**try**(

Connection conn=DriverManager.*getConnection*(url,username,password);

Statement st=conn.createStatement();

ResultSet rs=st.executeQuery(query);

){

**while**(rs.next()){

System.***out***.println(rs.getInt(1)+" has name "+rs.getString(2));

}

}

**catch**(Exception e){

e.printStackTrace();

}

}

}

**38.   Describe the different approaches of String processing.**

Ans : In java , String is a class of java.lang package which represents group of characters . String is immutable in Java which means once created we cannot modify the object of the String class . There are two ways to create object of String class in Java :-

1. By Using String Literal :- Anything written within double quotes is string literal .

String str=”Sachin”;

When we will create the object of String class using String literal , String object will gets created in String constant pool instead of heap . Before creating the object , JVM will check if object representing the same sequence of characters is already available . If it is available then JVM will return the reference of already existing object , it wont create new object.

1. By using new keyword :-

String str=new String(); //will create empty string

String str=new String(“Sachin”);

String class has overloaded constructors . We can choose the constructor as per our requirements.

When we will create the object of String class using new keyword , String object will be created in heap. Every time we will write new , new object will be created in the memory.

**39.   What is the difference between System.out ,System.err and System.in?**

Ans : System.out and System.err both are represents OutputStream and are by default connected to Console. System.out is used for displaying normal messages whereas System.err is used for displaying error messages .

System.in represents InputStream . It is connected to keyboard by default that’s why it is used to take input from the user.