**Introduction to JavaScript(CH-1)**

**Program:-**

Program is a collection of set of instructions to be executed to produce the output using a special type of compiler or software.

**Programming Language:-**

The language which is used to represent the instructions of a program to execute it is called a programming language.

Types:-

1. structural/procedural language:-it is based on structure or procedures.

ex-c

1. partial object oriented language:-which may or may not use the concepts of oops to write a program.

ex-c++, php

1. fully object oriented language:-without the help of oops we cannot write a program.

ex-java,.net

1. object based language:-based on objects.

ex-javascript, vbscript

**What is java-script:-**

JavaScript is a programming language that can be included on web pages to make them more interactive. You can use it to check or modify the contents of forms, change images, open new windows and write dynamic page content.JavaScript is a very powerful **client-side scripting language**. JavaScript is used mainly for enhancing the interaction of a user with the webpage. In other words, you can make your webpage more lively and interactive, with the help of JavaScript. JavaScript is also being used widely in game development and  mobile application development. JavaScript was developed by Brendan Eich in 1995, which appeared in Netscape, a popular browser of that time.

**Features of JS:-**

1. client-side

2. interpreted

3. object oriented

4. high level language

5. dynamic

6. easy to write

**Advantages of JS:-**

* **Interpreted languages**: JavaScript is an interpreted language. It requires no compilation process so no compiler is required. The browser interprets JavaScript as it HTML tags.
* **Easy to learn**: The syntax of JavaScript is very easy. Any person can learn it very easily and use it to develop dynamic and attractive  websites.
* **Easy to Debug and Test**: JavaScript code is interpreted line by line. The errors are indicated along with line number. It is very easy to find error in the code, correct it and test it gain.
* **Event-Based Programming**: JavaScript is an event-based language. It means that different code segment are executed when certain event occur. For example, a code segment may execute when the user clicka button or moves a mouse over an object etc.
* **Procedural Capabilities**: JavaScript provides all capabilities of a procedural language. It provides condition checking, loops and branching facilities that can be executed in a web page.
* **Platform Independence**:JavaScript is platform independed language. Any JavaScript-enabled browser can understand and interpreted JavaScript code. Any JavaScript code is executed on different types of hardware a JavaScript program written for.

**Disadvantages of JS:-**

* **Security Issues**: Javascript snippets, once appended onto web pages execute on client servers immediately and therefore can also be used to exploit the user's system. While a certain restriction is set by modern web standards on browsers, malicious code can still be executed complying with the restrictions set.
* **Javascript rendering varies**: Different layout engines may render Javascript differently resulting in inconsistency in terms of functionality and interface. While the latest versions of javascript and rendering have been geared towards a universal standard, certain variations still exist. [**Website Usability Consultants all over the world**](http://www.nextprise.com/website-usability-consulting.php) make a living on these differences, but it enrages thousands of developers on a daily basis.

**Uses of JS:-**

* **Input Validation**: JavaScript can be used to validate the input. Data entered informs should be validated before it is processed.
* **Mouse Rollover Effects**: JavaScript can used to create different buttons with interesting mouse rollover effects. It makes browsing more interesting and attractive.
* **Popup Windows**: JavaScript can be used to create popup windows. These windows are normally used to display important announcements, offers and news etc.
* **Dynamic Contents**: JavaScript can be used to generate dynamic contents in a website. Different HTML tags can be generated based on the user input etc.
* **User Interaction**: JavaScript can be used to interest with the user. The input entered by the user can be processed and proper message can be displayed to the user. The interactive capabilities of a website makes it more interesting and productive for the users.

**Elements of JavaScript(CH-2)**

**Character-set**:-

It combines small alphabets,capital alphabets,digits and special symbols to write a program.

* a-z(ascii code->97-122)
* a-z(ascii code->65-90)
* 0-9(ascii code->48-57)

**Constants:-**

Constants are the fixed quantity whose value can never be change throughout the program execution.

Types:-

1. Integer constant-

-may be +ve or –ve.

-examples are 9,-346

2. Float/real constant-

-may be +ve or \_ve

-examples are 1/3,-4.89

3. Character/string constant-

-these constants are enclosed within a pair of double quote and size is not fixed. ex-“anil”, ”s”

**Variables:-**

These are the temporary named memory locations where the constant values are stored. Every constant value must be associated with a variable inside the memory.

Rules for constructing a variable:-

1. variable name must starts with an alphabet or an underscore(\_).
2. it cannot starts with a digit.
3. but digits can be placed after first place.
4. no blank space is allowed while declaring a variable.
5. variable name length must not be more then 38 characters.
6. no special symbols except underscore is allowed while declaring a variable.

**Comment Lines:-**

These are used to deactivate line of code(s) which are present but not used inside the program.

Types:

1. single line comment:-used to deactivate a single line using the symbol (//).

ex-//printf(“hello”);

1. multiline comment:-used to deactivate more than one line at a time.(/\* \*/)

ex-/\* int a;

a=5;

printf(“%d”,a); \*/

**Datatypes:-**

Datatype decides which type of dada a variable can take. it always used as a suffix for the variable. There are different types datatypes used in c programming language.

System requirement to write a program:-

1. an editor:-notepad/notepad++/editplus/dreamweaver
2. a browser:- google chrome/mozila/opera/interner explorer

how to write an run java-script programs:-

1. first of all open an editor like notepad and write your program/code.
2. Then save this file just like html means choose any location and give a file name with extension “.html” and save the file.
3. Go to that location and double click on the file to view the output.

**(1)Write a simple program to print your name ?**

<html>

<head>

<title>abc</title>

<script>

document.write(“hello students”);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

**(or)**

<html>

<head>

<title>abc</title>

</head>

<body bgcolor=”pink” text=”violet”>

<script>

document.write(“hello students”);

</script>

</body>

</html>

**Description:-**

“document” is a pre-defined object in java-script which is used to call a pre-defined method.

“write” is a method/function in java-script which is used to print a message on the screen.

**Operators:-**

These are the special symbols which are used to operate or perform task over the operand or variable.

Types:-

1. **Assignment operator:-** used to assign a value to the variable. it is of two types.
2. simple assignment(=)

ex: var a=5;

1. compound assignment(+=,-=,\*=,/=)

ex: sum+=i;

1. **Special operator:-**

(i)comma:-used to separate constants, variable and expressions.

Ex:-

constant separation :-var a=new Array(1,2,3,4,5);

variable separation :-var a,b,c;

expression separation :-var a=5,b=4;

(ii)semicolon:- used to terminate/close a line of code.

Ex:- var a=9;

1. **concatenation operator**:- (+)used to concatenate/join string with variable part or with tags.

Ex:- document.write(“addition value=”+c);

1. **Arithmetic operator:-** used to perform mathematical operation over operands using different types of symbols like +,-,\*,/,%.

**(4)Write a program to add two integer number ?**

<html>

<head>

<title>abc</title>

<script>

var a=5,b=4,c;

c=a+b;

document.write(“addition value=”+c);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

**(5)Write a program to input two integer number and find their addition, subtraction, multiplication, division and modulo-division ?**

<html>

<head>

<title>abc</title>

<script>

var n1=10,n2=2,a,s,m,d,md;

a=n1+n2;

s=n1-n2;

m=n1\*n2;

d=n1/n2;

md=n1%n2;

document.write(“addition value=”+a+”<br>”);

document.write(“subtraction value=”+s+”<br>”);

document.write(“multiplication value=”+m+”<br>”);

document.write(“division value=”+d+”<br>”);

document.write(“modulo division value=”+md);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

**(7)Write a program to find the area and circumference of a circle ?**

<html>

<head>

<title>abc</title>

<script>

var r=6,a,c;

a=3.141\*r\*r;

c=2\*3.141\*r;

document.write(“area of circle=”+a+”<br>”);

document.write(“circumference of circle=”+c);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

**(8)Write a program to input temperature in farhenheit scale and display it in celsius scale ?**

<html>

<head>

<title>abc</title>

<script>

var f=98.7,c;

c=(f-32)\*5/9;

document.write(“temperature in celsius scale is=”+c);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

**(9)Write a program to swap two integer number using third variable ?**

<html>

<head>

<title>abc</title>

<script>

var a=5,b=4,c;

c=a;

a=b;

b=c;

document.write(“after swapping value of a=”+a+”<br>”);

document.write(“after swapping value of b=”+b);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

**(10)Write a program to swap two integer number without using third variable ?**

<html>

<head>

<title>abc</title>

<script>

var a=8,b=7;

a=a+b;

b=a-b;

a=a-b;

document.write(“after swapping value of a=”+a+”<br>”);

document.write(“after swapping value of b=”+b);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

**(12)Write a program to calculate simple interest ?**

<html>

<head>

<title>abc</title>

<script>

var p=5000,r=2,t=3;

si=(p\*r\*t)/100;

document.write(“simple interest=”+si);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

1. **Relational/Comparison operator:-** used to compare between two or more operands using different symbols like >,<,>=,<=,==,!=.

ex:- a>b

1. **Logical/Boolean operator:-** used to perform logical operation like producing true/false value using and,or,not,xor gates.

AND(&&) OR(||) NOT(!) XOR(^)

T T->T T T->T T->F T T->F

T F->F T F->T F->T T F->T

F T->F F T->T F T->T

F F->F F F->F F F->F

1. **Unary operator:-** the operator which is used to perform over a single operand is called unary operator.

ex:- a++

1. **Binary operator:-** the operator which is use to operate over two operand is called binary operator.

ex:- a+b

1. **Ternary/conditional operator:-** the operator which is used to operate over more than two operands are called ternary operator.

syn:- (condition)? print statement1:print statement2;

**(13)Write a program to find the greater number among two inputted number using ternary operator ?**

<html>

<head>

<title>abc</title>

<script>

var a=5,b=6;

(a>b)?document.write(“a is greater”):document.write(“b is greater”);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

**(14)Write a program to check for leap year using ternary operator ?**

<html>

<head>

<title>abc</title>

<script>

var y=2018;

(((y%100!=0)&&(y%4==0))||(y%400==0))?document.write(“leap year”):document.write(“not”);

</script>

</head>

<body bgcolor=”pink” text=”violet”>

</body>

</html>

1. **Bitwise operator:-**the operator which is used to perform bit operation over the operand or variable as 0 and 1 is called bitwise operator.

types:-

1. bitwise and(&)
2. bitwise or(|)
3. bitwise xor(^)
4. bitwise complement(~)
5. bitwise left shift(<<)
6. bitwise right shift(>>)

**(16)Write a program for bitwise left-shift operator ?**

<html>

<head>

<title>abc</title>

<script>

var a=7,b=2,c;

c=a<<b;

document.write(“value of c=”+c);

</script>

</head>

<body bgcolor=”red” text=”green”>

</body>

</html>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0

(11100)2=1x24+1x23+1x22+0x21+0x20=16+8+4+0+0=28

**(17)Write a program for bitwise complement operator ?**

<html>

<head>

<title>abc</title>

<script>

var a=5,b;

b=~a;

document.write(“value of b=”+b);

</script>

</head>

<body bgcolor=”red” text=”green”>

</body>

</html>

**(18)Write a program for bitwise and operator ?**

<html>

<head>

<title>abc</title>

<script>

var a=14,b=7,c;

c=a&b;

document.write(“value of c=”+c);

</script>

</head>

<body bgcolor=”red” text=”green”>

</body>

</html>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0

(110)2=1x22+1x21+0x20=4+2+0=6

1. **Increment and decrement operator:-**
2. Increment operator:- this operator is used to increase the value of the variable by one in two different ways.
3. pre-increment:- first increase the value after that update it.

ex- ++a

1. post-increment:- first update the value after increase it.

ex- a++

**(19)Write a program for increment operator ?**

<html>

<head>

<title>abc</title>

<script>

var a=5,b,c;

b=a++ + ++a;

c=++a + a++;

document.write(“value of b=”+b+”<br>”);

document.write(“value of c=”+c);

</script>

</head>

<body bgcolor=”red” text=”green”>

</body>

</html>

1. Decrement operator:- this operator is used to decrease the value of the variable by one in two different ways.
2. pre-decrement:- first increase the value after that update it.

ex- --a

1. post-decrement:- first update the value after increase it.

ex- a--

**(20)Write a program for decrement operator ?**

<html>

<head>

<title>abc</title>

<script>

var a=5,b,c;

b=a--;

c=a--;

document.write(“value of b=”+b+”<br>”);

document.write(“value of c=”+c);

</script>

</head>

<body bgcolor=”red” text=”green”>

</body>

</html>

**Control Structure(CH-3)**

Control structure is the control flow of program execution where we can use different types of statements to process our programs. There are different types of statements available in control structures that are:

1. conditional statements
2. unconditional statements
3. looping statements
4. **Conditional statements:-** These are the types of statements where we use conditions to process the program using some relational operators to produce true or false value.

Types:-

1. **If/simple if:-**

Here we are able to specify only one condition and one print statement. The demerit of this statement is, it cannot go to the default part.

syn:- if(condition)

print statement;

**(21)Write a program to check whether an inputted number is even or not using simple if ?**

<html>

<head>

<title>abc</title>

<script language=”JavaScript”>

var a=8;

if(a%2==0)

document.write(“number is even”);

</script>

</head>

<body bgcolor=”cyan” text=”red”>

</body>

</html>

1. **If else:-**

We are able to specify only one condition but two print statement means one for true part and another for false part.

syn:- if(condition)

print statement1;

else

print statement2;

**(22)Write a program to input two integer number and find the greater one using if else ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var a=3,b=4;

if(a>b)

document.write(“a is greater”);

else

document.write(“b is greater”);

</script>

</head>

<body bgcolor=”gray” text=”indigo”>

</body>

</html>

1. **Nested if:-**

Here we are able to specify ‘n’ number of conditions and ‘n’ number of print statements. But the problem is that it is a very complex type of statement where we get confuse about the starting and ending of if block because of nested structure.

Syn:- if(condition-1)

if(condition-2)

print statement1;

else

print statement2;

**.**

**.**

else

if(condition-n)

print statement-1;

else

print statement-n;

**(23)Write a program to input a year and check whether it is leap year or not using nested if ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var y=2018;

if(y%100!=0)

if(y%4==0)

document.write(“leap year”);

else

document.write(“not”);

else

if(y%400==0)

document.write(“leap year”);

else

document.write(“not”);

</script>

</head>

<body bgcolor=”green” text=”red”>

</body>

</html>

**(24)Write a program to input three integer number and find the greatest one using nested if ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var a=45,b=23,c=90;

if(a>b)

if(a>c)

document.write(“a is greatest”);

else

document.write(“c is greatest”);

else

if(b>c)

document.write(“b is greatest”);

else

document.write(“c is greatest”);

</script>

</head>

<body bgcolor=”green” text=”red”>

</body>

</html>

1. **Ladder else if:-**

It is the type of conditional statement where we can specify ‘n’ number of conditions and ‘n’ number of print statements. It is very simple as compare to nested if because here no nesting of condition is there and we can specify one condition and one print statement so on.

syn:- if(condition-1)

print statement-1;

else if(condition-2)

print statement-2;

**.**

**.**

else

print statement-n;

**(25)Write a program to input an integer number and check whether it is even or odd and positive or negative using ladder else if ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var n=5;

if(n%2==0&&n>0)

document.write(“number is even and positive”);

else if(n%2!=0&&n<0)

document.write(“number is odd and negative”);

else if(n%2==0&&n<0)

document.write(“number is even and negative”);

else if(n%2!=0&&n>0)

document.write(“number is odd and positive”);

else

document.write(“special number”);

</script>

</head>

<body bgcolor=”green” text=”red”>

</body>

</html>

**(26)write a program to input three integer number and find the greatest one using ladder else if ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var a=5,b=90,c=67;

if(a>b&&a>c)

document.write(“a is greatest”);

else if(b>a&&b>c)

document.write(“b is greatest”);

else

document.write(“c is greatest”);

</script>

</head>

<body bgcolor=”green” text=”red”>

</body>

</html>

**(28)Write a program to find the grade of an emp by finding gross salary where basic,ta,da,hra are given using ladder else if ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var bs=24500,ta,da,hra,gs;

ta=0.05\*bs;

da=0.075\*bs;

hra=0.1\*bs;

gs=bs+ta+da+hra;

if(gs>=100000)

document.write(“a grade employee”);

else if(gs>=75000&&gs<100000)

document.write(“b grade employee”);

else if(gs>=50000&&gs<75000)

document.write(“c grade employee”);

else if(gs>=20000&&<50000)

document.write(“d grade employee”);

else

document.write(“e grade employee”);

</script>

</head>

<body bgcolor=”green” text=”red”>

</body>

</html>

**(29)Write a program to find the daily wages of a worker according to the following conditions using ladder else if statement?**

**duty in hours** **amount in rupees**

within first 8 hours 100 rupees

next 4 hours 20 rs/hr

next 4 hours 40 rs/hr

next 4 hours 60 rs/hr

next 4 hours 80 rs/hr

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var hr=17,amt;

if(hr>=1&&hr<=8)

amt=100;

else if(hr>=9&&hr<=12)

amt=100+(hr-8)\*20;

else if(hr>=13&&hr<=16)

amt=180+(hr-12)\*40;

else if(hr>=17&&hr<=20)

amt=340+(hr-16)\*60;

else if(hr>=21&&hr<=24)

amt=580+(hr-20)\*80;

document.write(“amount incurred by the worker=”+amt);

</script>

</head>

<body bgcolor=”green” text=”red”>

</body>

</html>

1. **Unconditional statements:-** These are the types of statements where we don’t use any condition rather we use no. of cases to produce true or false statement. Cases consist of constants or expressions.

Types:-

1. **Switch-case statement:-**

It is the type of unconditional statement where we are able to specify n no of cases. Case blocks are created by the help of keyword case and exited by the help the keyword break. If no case is satisfied inside the switch block then it goes to the default block specified by default keyword.

syn:- switch(expression)

{

case <case-constant 1>:

statement(s);

break;

.

.

case <case-constant n>:

statement(s);

break;

default:

statement(s);

}

**(31)Write a program to display the day name using switch case ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var day=1;

switch(day)

{

case 1:

document.write(“the day is sunday”);

break;

case 2:

document.write(“the day is monday”);

break;

case 3:

document.write(“the day is tuesday”);

break;

case 4:

document.write(“the day is wednesday”);

break;

case 5:

document.write(“the day is thursday”);

break;

case 6:

document.write(“the day is friday”);

break;

case 7:

document.write(“the day is saturday”);

break;

default:

document.write(“wrong choice”);

}

</script>

</head>

<body bgcolor=”green” text=”red”>

</body>

</html>

**(32)Write a program to input a character and check whether it is vowel or consonant using switch case ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var ch=”u”;

switch(ch)

{

case ‘a’:

case ‘e’:

case ‘i’:

case ‘o’:

case ‘u’:

case ‘A’:

case ‘E’:

case ‘I’:

case ‘O’:

case ‘U:

document.write(“it is a vowel”);

break;

default:

document.write(“it is consonant”);

}

</script>

</head>

<body bgcolor=”green” text=”red”>

</body>

</html>

**(33)Write a program to implement a calculator using switch case ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var a=10,b=2,c;

var choice=”\*”;

switch(choice)

{

case ‘+’:

c=a+b;

break;

case ‘-‘:

c=a-b;

break;

case ‘\*’:

c=a\*b;

break;

case ‘/’:

c=a/b;

break;

case ‘%’:

c=a%b;

break;

default:

document.write(“wrong choice”);

}

document.write(“calculated value=”+c);

</script>

</head>

<body bgcolor=”green” text=”red”>

</body>

</html>

1. **Looping statements:-** These are also called iterative or repeatative statements. When we want to execute or print one statement for more than one time with a single specification then we use the concept of looping. For creating a looping statement we need components.
2. **Initialization**:- from which the variable value starts execution or printing.
3. **Condition**:- upto which the variable value should goes.
4. **Incr/decr**:- used to increase or decrease the value of the variable after each execution.

Types:-

1. **While loop:-**

It is otherwise called as top-tested loop or pre-tested loop or entry control loop. In this type of looping statement first the condition is checked after the statements are get executed and printed. if the condition is false then no statement is executed or printed.

syn:- variable value initialization;

while(condition)

{

statement(s);

incr/decr;

}

print statement(s);

**Wap to print 1 to 10 using while loop ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var i=1;

while(i<=10)

{

document.write(“<h1>”+“value of i=”+i+”</h1>”);

i++;

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Wap to input a number and check whether it is prime or not using while loop ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var i=1,n=7,c=0;

while(i<=n)

{

if(n%i==0)

c++;

}

if(c==2)

document.write(“<h1>”+“number is prime” +”</h1>”);

else

document.write(“<h1>”+“number is not prime” +”</h1>”);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Wap to input a decimal number and find its binary equivalent using while loop ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var num=5,prd=1,sum=0,rem;

while(num!=0)

{

rem=num%2;

sum=sum+rem\*prd;

prd=prd\*10;

num=num/2;

}

document.write(sum);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

1. **Do-while loop:-**

It is otherwise called as bottom-tested loop or post-tested loop or exit control loop. in this type of looping statement first the statements are get executed and printed after that the condition is checked. if the condition is false then no statement is executed or printed.

syn:- variable value initialization;

do

{

statement(s);

incr/decr;

} while(condition);

print statement(s);

**Wap to print the series like 11,22,...,99 using do while loop ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var i=11;

do

{

document.write(“<h1>”+“value of i=”+i+”</h1>”);

i=i+11;

}

while(i<=99);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Wap to input a number and reverse it using do while loop ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var num=123,rev=0,rem;

do

{

rem=num%!0;

rev=rev\*10+rem;

num=num/10;

}

while(num|=0);

document.write(“reverse value=”+rev);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**3. For loop:-**

It is the simplest type of looping statement because here all the three parts of the loop written in one line; so it reduce the line of codes.

syn:- for(initialization; condition; incr/decr)

**Wap to print the series like 50,45,...,5 using for loop?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var i;

for(i=50;i>=5;i=i-5)

{

document.write(“value of i=”+i);

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Wap to find the sum of all even and product of all odd numbers between 1 to 50 using for loop ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var i,sum=0,prd=1;

for(i=1;i<=10;i++)

{

if(i%2==0)

sum=sum+i;

else

prd=prd\*i;

}

document.write(“value of sum=”+sum+”<br>”);

document.write(“value of prd=”+prd);

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Wap to print Fibonacci series like 0,1,1,2,3,5,8,13,21,34 using for loop ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var i,a=0,b=1,sum=0;

document.write(a+”<br>”);

document.write(b+”<br>”);

for(i=1;i<=8;i++)

{

sum=a+b;

document.write(c+”<br>”);

a=b;

b=sum;

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Wap to find hcf and lcm of two numbers using for loop ?**

<html>

<head>

<title>series</title>

<script language=”JavaScript”>

var i,hcf,lcm,a=10,b=20;

for(i=1;i<=a&&i<=b;i++)

{

if(a%i==0&&b%i==0)

hcf=i;

}

lcm=(a\*b)/hcf;

document.write(“value of hcf=”+hcf+”<br>”);

document.write(“value of lcm=”+lcm);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Nested for:-**

When one or more than one for loops are nested inside another for loop then it is called nested for loop.

**(55)Write a program for pyramid series ?**

<html>

1

1. 2

1 2 3

1 2 3 4

1 2 3 4 5

<head>

<title>abc</title>

</head>

<script language=”JavaScript”>

var i,j;

for(i=1;i<=5;i++)

{

for(j=1;j<=i;j++)

document.write(j);

document.write (“<br>”);

}

</script>

</head>

<body>

</body>

</html>

**(56)Write a program for pyramid series ?**

<html>

1 2 3 4 5

1 2 3 4

1 2 3

1 2

1

<head>

<title>abc</title>

<script>

for(i=5;i>=1;i--)

{

for(j=1;j<=i;j++)

document.write(j);

document.write (“<br>”);

}

</script>

</head>

<body>

</body>

</html>

**(58)Write a program for pyramid series ?**

<html>

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

<head>

<title>abc</title>

<script type=”JavaScript”>

var i,j,k=1;

for(i=1;i<=5;i++,k=k+2)

{

for(j=i;j<=4;j++)

document.write(“&nbsp&nbsp&nbsp“);

for(j=1;j<=k;j++)

document.write(“\*”);

document.write (“<br>”);

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**(59)Write a program to print floyd triangle upto five rows ?**

<html>

1

2 3

4 5 6

7 8 9 10

11 12 13 14 15

<head>

<title>abc</title>

</script lang=”JavaScript”>

var i,j,k=1;

for(i=1;i<=5;i++)

{

for(j=1;j<=i;j++)

{

document.write(k);

k++;

}

document.write (“<br>”);

}

</script>

</head>

<body>

</body>

</html>

**Array(CH-4)**

Array is a derived data-type in java-script. It is also an object in java-script. It is used to store homogeneous/similar type of data. Whenever we want to store more than one value of similar type inside a single variable using a subscript/size then we use the concept of array. Every element should be separated by a comma and are accessed by the help of a unique index number which is by default start from zero.

Syn:-

var <variablename>=new Array(values);

**note:-** values are optional here.

Write a program to print array elements ?

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(3,66,12,70,5);

document.write(“first element=”+a[0]+”<br>”);

document.write(“second element=”+a[1]+”<br>”);

document.write(“third element=”+a[2]+”<br>”);

document.write(“fourth element=”+a[3]+”<br>”);

document.write(“fifth element=”+a[4]);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

Write a program to print array elements ?

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(3,66,12,70,5);

var i;

for(i=0;i<5;i++)

{

document.write(a[i]+”<br>”);

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

Write a program to print array elements in reverse order?

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(3,66,12,70,5);

var i;

for(i=4;i>=0;i--)

{

document.write(a[i]+”<br>”);

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Pre-defined methods of array:-**

**1. reverse():-**used to reverse all the elements.

Ex:-

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(3,66,12,70,5);

var b=a.reverse();

document.write(b);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-5,70,12,66,3

**2. sort():-** used to sort array elements in ascending order by default.

Ex:-  
<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(“r”,”e”,”k”,”a”,”s”);

var b=a.sort();

document.write(b);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-a,e,k,r,s

**3.push():-** used to push element at the last index of the array.

Ex:-  
<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(9,33,12,90,6);

var b=a.push(55);

document.write(a);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-9,33,12,90,6,55

**4.pop():-** used to delete last element from the last index of the array.

Ex:-  
<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(9,33,12,90,6);

var b=a.pop();

document.write(a);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-9,33,12,90

**5.join():-** used to join array elements to create a string using delimeter.

Ex:-  
<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(9,33,12,90,6);

var b=a.join(“\*”);

document.write(b);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-9\*33\*12\*90\*6

**6.splice():-** used to removes or replaces or adds elements by specifying position and number of items.

Ex:-  
<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(9,33,12,67);

var b=a.splice(2,0,89,123);

document.write(b);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-

**7.slice():-** used to returns selected elements in an array as new array elements.

Ex:-  
<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(9,33,12,90,6);

var b=a.slice(1,3);

document.write(b);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-

**8.shift():-** used to delete first element from the array.

Ex:-  
<html>

<head>

<title>array</title>

<script>

var a=new Array(9,33,12,90,6);

var b=a.shift();

document.write(a);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-

**9.unshift():-** used to add elements at the beginning of the array.

Ex:-  
<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(9,33,12,90,6);

var b=a.unshift(12,8);

document.write(a);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-

**10.concat():-** used to concatenate/joint two different array to produce a single array.

Ex:-  
<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=new Array(9,33,12,90);

var b=new Array(25,1,123);

var c=a.concat(b); //var c=b.concat(a);

document.write(c);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

o/p-9,33,12,90,25,1,123

**Functions(CH-5)**

**What is function:-**

Functions are the self-contained block of codes which are used to perform specific operation over the data. Functions are used to divide complex programs into simple parts. There are two types of functions available in java-script, that are:

1. User-defined functions

2. Pre-defined functions

**User-defined functions:-**

The functions which are declared and defined by the user at the time of writing a program are called user-defined functions. User-defined functions contain different components like:

i. A keyword “function” to define a function.

ii. A user-defined function name followed by a pair of parenthesis.

iii. Number of arguments to be passed to the function for operations.

iv. Body of the function

v. A return statement to return a value to the function.

Syn:-

function function-name(arguments)

{

//statements;

return statement;

}

**Write a program to add two number using function?**

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

function addition(a,b)

{

c=a+b;

return(c);

}

document.write(“addition value=”+addition(5,4));

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Write a program to find the factorial of a number using function ?**

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

function factorial(n)

{

var f=1;

while(n!=0)

{

f=f\*n;

n--;

}

return(f);

}

document.write(“factorial value=”+factorial(5));

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Pre-defined functions:-**

The functions/methods which are already available inside java-script library and carry some meaning for the java-script interpreter are called pre-defined functions. All the java-script pre-defined functions are accessed by the help of pre-defined object.

1. math functions

2. conversion functions

3. user input functions

4. date functions

**Math Functions:-**

All math functions are called by the help of pre-defined object “Math” object by the help of dot(.) operator.

1.**pow(m,e):-** used to calculate power value by providing mantissa and exponent.

Ex:- document.write(Math.pow(5,2)); **o/p-25**

2.**sqrt(value):-** used to calculate square root value of a number.

Ex:- document.write(Math.sqrt(576)); **o/p-24**

3.**ceil(value):-** used to calculate ceiling value by increasing to next higher round-off value.

Ex:- document.write(Math.ceil(10.1)); **o/p-11**

4.**floor(value):-** used to calculate flooring value by decreasing it to next nearest round-off value.

Ex:- document.write(Math.floor(10.9)); **o/p-10**

5.**round(value):-** used to round off a number according to round-off rule.

Ex:- document.write(Math.round(5.7)); **o/p-6**

6.**abs(value):-** used to display absolute value of a number.

Ex:- document.write(Math.abs(-15)); **o/p-15**

7.**sin(radian):-** used to display sin theta value.

Ex:- document.write(Math.sin(0)); **o/p-0**

8.**cos(radian):-** used to display cos theta value.

Ex:- document.write(Math.cos(0)); **o/p-1**

9.**tan(radian):-** used to display tan theta value.

Ex:- document.write(Math.tan(45)); **o/p-1**

10.**max(value1,...,valuen):-** used to return maximum value from the list.

Ex:- document.write(Math.max(5,2,34,1,78)); **o/p-78**

11.**min(value1,…,valuen):-** used to return minimum value from the list.

Ex:- document.write(Math.min(12,5,172,0,29)); **o/p-0**

12.**log(value,base):-** used to return logarithmic value from log table.

Ex:- document.write(Math.log(10,2)); **o/p-1**

**Conversion Function:-**

Use to convert one type of data into another type.

1.**parseInt():-** used to convert string type data into integer type.

Syn:- var <variable-name>=parseInt(variable-name);

2.**parseFloat():-** used to convert string value into float value.

Syn:- var <variable-name>=parseFloat(variable-name);

3.**eval():-** used to convert string value into integer/float values.

Syn:- var <variable-name>=eval(variable-name);

4.**toString():-** used to convert integer/float value into string values.

Syn:- var <variable-name>=toString(variable-name);

**User-input Function:-**

**1. prompt():-** used to take user-input from keyboard by creating boxes using this method.

Syn:- var <variable-name>=prompt(“interactive message”);

**Wap to add two integer number using prompt method?**

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=prompt(“enter first number”);

var b=parseInt(a);

var c=prompt(“enter second number”);

var d=parseInt(c);

var e=b+d;

document.write(“<h1>”+“addition value=”+e+”</h1>”);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**2. confirm():-**used to display a confirmation box before printing any message/output. This box contains two buttons; one is “ok” and another is “cancel”.

Syn:- confirm(“interactive message”);

Wap to add two integer numbers using prompt and confirm method?

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=prompt(“enter first number”);

var b=parseInt(a);

var c=prompt(“enter second number”);

var d=parseInt(c);

var e=b+d;

confirm(“would you like to print the output”);

document.write(“<h1>”+“addition value=”+e+”</h1>”);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**3.alert():-**used to display an alert box if the condition is false or anything goes wrong according to user’s requirement.

Syn:- alert(“interactive message”);

**Wap for alert method?**

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

var a=prompt(“enter your age”);

var b=parseInt(a);

document.write(“<h1>”+“addition value=”+e+”</h1>”);

if(b>=18)

document.write(“<h1>”+“you are eligible for vote”+”</h1>”);

else

alert(“you are not eligible”);

</script>

</head>

<body bgcolor=”red” text=”blue”>

</body>

</html>

**Timer Function:-**

1.getDay():- used to catch system day name.

2. getDate():- used to catch system date.

3.getMonth():- used to catch system month.

4.getFullYear():- used to catch system year.

5.getHours():- used to catch system hours from system date.

6.getMinutes():- used to catch system minutes from system date.

7.getSeconds():- used to catch system seconds from system date.

8.getMilliseconds():- used to catch system milliseconds from system date.

9.getTime():- used to catch system time from system date.

**Wap to display a digital clock using date functions ?**

<html>

<head>

<title>array</title>

<script language=”JavaScript”>

function clock()

{

var d=new Date();

var h=d.getHours();

var m=d.getMinutes();

var s=d.getSeconds();

m=checkTime(m);

s=checkTime(s);

document.getElementById(‘txt’).innerHTML=h+”:”+m+”:”+s;

var t=setTimeout(clock,500);

}

function checkTime(i)

{

if(i<10)

{

i=”0”+i;

}

return(i);

}

</script>

</head>

<body onload=”clock()” >

<div id=”txt”></div>

</body>

</html>

**Validation:-**

It is the type of restriction where user/programmer wants to provide some authentication/authorization to the page. It can be provided in two different ways:

**1. empty validation:-**

Wap to validate a login page ?

<html>

<head>

<title>login page validation</title>

<script language=”JavaScript”>

function myvalidate()

{

var id=document.form1.text1.value;

var pwd=document.form1.text2.value;

if(id==null||pwd==null)

alert(“login failed”);

else

document.write(“<h1>”+“your id=”+id+”</h1>”);

document.write(“<h1>”+“your pwd=”+pwd+”</h1>”);

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

<font size=5 face=”lucida handwriting”><marquee bgcolor=”grey”>welcome to login page</marquee></font>

<h1 align=”center”><u>login info</u></h1>

<center>

<form name=”form1” method=”POST” action=””>

<table height=150 width=250>

<tr>

<td>Enter id:</td><td><input type=”text” name=”text1” placeholder=”enter your id”></td>

</tr>

<tr>

<td>Enter pwd:</td><td><input type=”password” name=”text2” placeholder=”enter your password”></td>

</tr>

<tr>

<td align=”center”><input type=”button” value=”submit” onclick=”myvalidate()”></td><td align=”center”><input type=”button” value=”reset”></td>

</tr>

<tr>

<td><input type=”checkbox”>stay sign-in</td><td align=”right”><a href=””>forgotten password ?</a></td>

</tr>

</table>

</form>

</center>

</body>

</html>

**2. inline validation:-**

Wap to validate a login page ?

<html>

<head>

<title>login page validation</title>

<script language=”JavaScript”>

function myvalidate()

{

var id=document.form1.text1.value;

var pwd=document.form1.text2.value;

if(id==null||pwd==null)

alert(“login failed”);

else

document.write(“<h1>”+“your id=”+id+”</h1>”);

document.write(“<h1>”+“your pwd=”+pwd+”</h1>”);

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

<font size=5 face=”lucida handwriting”><marquee bgcolor=”grey”>welcome to login page</marquee></font>

<h1 align=”center”><u>login info</u></h1>

<center><form name=”form1” method=”POST” action=””>

<table height=150 width=250>

<tr>

<td>Enter id:</td><td><input type=”text” id=”text1” placeholder=”enter your id”></td>

</tr>

<tr>

<td>Enter pwd:</td><td><input type=”password” id=”text2” placeholder=”enter your password”></td>

</tr>

<tr>

<td align=”center”><input type=”button” value=”submit” onclick=”myvalidate()”></td><td align=”center”><input type=”button” value=”reset”></td>

</tr>

<tr>

<td><input type=”checkbox”>stay sign-in</td><td align=”right”><a href=””>forgotten password ?</a></td>

</tr>

</table>

</form>

</body>

</html>

Wap to add two number using textbox ?

<html>

<head>

<title>abc</title>

<script>

function add()

{

var a=parseInt(document.getElementById("t1").value);

var b=parseInt(document.getElementById("t2").value);

var c=document.getElementById("t3");

c.value=a+b;

}

</script>

</head>

<body>

Enter first no.:<input type="text" id="t1" name="t">

Enter second no.:<input type="text" id="t2" name="t2">

<input type="submit" name="submit" value="add" onclick="add()">

Addition value:<input type="text" id="t3" name="t3">

</body>

</html>

<html>

<head>

<title>abc</title>

<script>

function add()

{

var a=parseInt(document.f1.t1.value);

var b=parseInt(document.f1.t2.value);

var c=a+b;

document.write(c);

}

</script>

</head>

<body>

<form name="f1" method="post">

Ente first no.:<input type="text" name="t1">

Enter second no.:<input type="text" name="t2">

<input type="submit" name="submit" value="add" onclick="add()">

</body>

</html>

**Objects and Events(CH-6)**

**Objects:-**

1. Window:- it consists of following methods and properties.

Methods:-

prompt(),confirm(),alert(),setInterval(),clearInterval()

Properties:-

closed,document,history,location,

1. Document:-it consists of following methods and properties.

Methods:-

write()

Properties:-

linkColor,alinkColor,vlinkColor,bgColor,fgColor,Form,Image[],lastModified

1. History:-it consists of following methods and properties.

Methods:-

back(),forward(),go()

Properties:-

length

1. Form:-it consists of following methods and properties.

Methods:-

submit(),reset()

Properties:-

elements[]

1. Image:-it consists of following methods and properties.

Methods:-

Image()

Properties:-

src,height,width,border,name

1. Array:-it consists of following methods and properties.

Methods:-

sort(),reverse,join,splice(),push(),pop(),shift(),unshift()

Properties:-

length

1. Date:-it consists of following methods and properties.

Methods:-

getDay(),getDate(),getMonth(),getYear(),getMinutes(),getSeconds(),getMilliseconds(),getTime()

1. Elements[]:-it consists of following methods and properties.

Properties:-

text,password,button,radio,checkbox,textarea,submit,reset

1. Options[]:-it consists of following methods and properties.

Properties:-

Index,text,value

1. Math:-it consists of following methods and properties.

Methods:-

pow(),sqrt(),ceil(),floor(),abs(),round(),sin(),cos(),tan(),log(),max(),min()

Properties:-

E,PI

1. String:-it consists of following methods and properties.

Methods:-

substring(),lastIndexOf(),indexOf(),charAt(),charCodeAt(),toLowerCase(),toUpperCase()

1. Global:-it consists of following methods and properties.

Methods:-

parseInt(),parseFloat()

**Events:-**

JavaScript's interaction with HTML is handled through events that occur when the user or the browser manipulates a page. When the page loads, it is called an event. When the user clicks a button, that click too is an event. Other examples include events like pressing any key, closing a window, resizing a window, etc.

**1. onabort:-**

**2. onblur:-**

<html>

<head>

<title>abc</title>

<script>

function abc()

{

var x=document.getElementById("txt");

x.value=x.value.toUpperCase();

}

</script>

<body>

Enter a character:<input type="text" id="txt" onblur="abc()">

</body>

</html>

**3. onchange:-**

<html>

<head>

<title>abc</title>

<script>

function xyz()

{

var x=document.getElementById("combo").value;

document.getElementById("dropdown").innerHTML = "You selected:"+x;

}

</script>

<body>

<select id="combo" onchange="xyz()">

<option value="January">January

<option value="February">February

<option value="March">March

</select>

<p id="dropdown"></p>

</body>

</html>

**4. onclick:-**

<html>

<head>

<title>abc</title>

<script>

function abc()

{

window.open(“abc.html”);

}

</script>

</head>

<body>

<input type=”button” value=”click me” onclick=”abc()”>

</body>

</html>

**5. ondblclcik:-**

<html>

<head>

<title>abc</title>

<script>

function abc()

{

Document.bgColor=”red”;

}

function def()

{

Document.bgColor=”green”;

}

function ghi()

{

Document.bgColor=”blue”;

}

</script>

</head>

<body>

<input type=”button” value=”btn1” ondblclick=”abc()”>

<input type=”button” value=”btn2” ondblclick=”def()”>

<input type=”button” value=”btn3” ondblclick=”ghi()”>

</body>

</html>

**6. onerror:-**

<html>

<head>

<title>abc</title>

<script>

function abc()

{

alert('The image could not be loaded.');

}

</script>

</head>

<body bgcolor=”red” text=”blue”>

<img src="image.gif" onerror="abc()">

</body>

</html>

**7. onfocus:-**

<html>

<head>

<title>abc</title>

<script>

function abc(x)

{

x.style.background=”red”;

}

</script>

<body>

Enter a character:<input type="text" id="txt" onfocus="abc(this)">

</body>

</html>

**8. onkeydown:-**

<html>

<head>

<title>abc</title>

<script>

function abc()

{

alert(“a key is pressed”);

}

</script>

<body>

Enter a character:<input type="text" id="txt" onkeydown="abc()">

</body>

</html>

**9. onkeypress:-**

<html>

<head>

<title>abc</title>

<script>

function abc()

{

alert(“a key is pressed”);

}

</script>

<body>

Enter a character:<input type="text" id="txt" onkeypress="abc()">

</body>

</html>

**10. onkeyup:-**

<html>

<head>

<title>abc</title>

<script>

function abc()

{

var x=document.getElementById("txt");

x.value=x.value.toUpperCase();

}

</script>

<body>

Enter a character:<input type="text" id="txt" onkeyup="abc()">

</body>

</html>

**11. onload:-**

Example done in timer function.

**12. onmousedown:-**

**13. onmousemove:-**

**14. onmouseout:-**

**15. onmouseover:-**

<html>

<head>

<title>abc</title>

<script>

function over()

{

document.bgColor=”red”;

}

function out()

{

document.bgColor=”green”;

}

</script>

</head>

<body>

<p onmouseover=”over()” onmouseout=”out()”>

Hello students !<br>

It is javascript class.<br>

We are doing event handling

</p>

</body>

</html>

**16. onmouseup:-**

**17. onselect:-**

<html>

<head>

<title>abc</title>

<script>

function abc(x)

{

alert(“you have selected some text”);

}

</script>

<body>

Enter a character:<input type="text" id="txt" onselect="abc()">

</body>

</html>

**18. onsubmit:-**

<html>

<head>

<title>abc</title>

<script>

function abc()

{

alert(“the form was submitted”);

}

</script>

<body>

Enter a character:<input type="text" id="txt" onfocus="abc(this)">

</body>

</html>

**19. onunload:-**

**20. ondrag:-**

**21. ondrop:-**

**22. oninvalid:-**

**23. onmessage:-**

**24. onprogress:-**

**25. onredo:-**

**26. onscroll:-**

**27. onundo:-**

**28. onwaiting:-**

**29. oninput:-**

<html>

<head>

<title>abc</title>

<script>

function myFunction()

{

var x=document.getElementById("txt1").value;

document.getElementById("demo").innerHTML = "You wrote: " + x;

}

</script>

</head>

<body>

<input type="text" id="txt1" oninput="myFunction()">

<p id="demo"></p>

</body>

</html>

**30. onplay:-**

**31. onpause:-**

**32. onplaying:-**

**33. onreset:-**

<html>

<head>

<title>abc</title>

<script>

function abc()

{

alert(“the form was reset”);

}

</script>

</head>

<body>

<form onreset=”abc()”>

Enter your name:<input type=”text”>

<input type=”reset”>

</form>

</body>

</html>

**DOM:-**

Stand for Document Object Model. It is a standard for how to get, change, add, or delete HTML elements. It defines HTML elements as objects. It defines properties of all HTML elements. It defines methods to access all HTML elements. It defines **events** for all HTML elements.

**DOM Methods:-**

HTML DOM methods are **actions** you can perform (on HTML Elements).HTML DOM properties are **values** (of HTML Elements) that you can set or change. The HTML DOM can be accessed with JavaScript (and with other programming languages).In the DOM, all HTML elements are defined as **objects**. The programming interface is the properties and methods of each object. A property is a value that you can get or set (like changing the content of an HTML element).A method is an action you can do (like add or deleting an HTML element).

**DOM Elements:-**

1. **elements by id:-**

var myElement = document.getElementById("txt1");

**2.Finding HTML elements by tag name:-**

var x = document.getElementsByTagName("p");

**3.Finding HTML elements by class name:-**

var x = document.getElementsByClassName("intro");

**4.Finding HTML elements by CSS selectors:-**

var x = document.querySelectorAll("p.intro");

1. **Finding HTML elements by HTML object collections:-**

var x = document.forms["frm1"];  
var text = "";  
var i;  
for (i = 0; i < x.length; i++)

{  
text += x.elements[i].value + "<br>";  
}  
document.getElementById("txt1").innerHTML = text;

**DOM Events:-**

* When a user clicks the mouse🡪onclick event
* When a web page has loaded🡪onload event
* When an image has been loaded🡪onload event
* When the mouse moves over an element🡪onmouseover event
* When an input field is changed🡪onchange event
* When an HTML form is submitted🡪onsubmit event
* When a user strokes a key🡪keytyped

Ex:-

<html>

<head>

<title>abc</title>

</head>

<body>

<h1 onclick="this.innerHTML='Ooops!'">Click on this text!</h1>

</body>

</html>

Ex:-

<html>

<head>

<title>abc</title>

<script>

function mOver(obj)

{

obj.innerHTML = "Thank You";

}

function mOut(obj)

{

obj.innerHTML = "Mouse Over Me";

}

</script>

</head>

<body>

<div onmouseover="mOver(this)" onmouseout="mOut(this)"

style="background:red;width:120px;height:20px;padding:40px;">

Mouse Over Me</div>

</body>

</html>

Ex:-

<html>

<head>

<title>abc</title>

<script>

function mDown(obj)

{

obj.style.backgroundColor = "blue";

obj.innerHTML = "Release Me";

}

function mUp(obj)

{

obj.style.backgroundColor="red";

obj.innerHTML="Thank You";

}

</script>

</head>

<body>

<div onmousedown="mDown(this)" onmouseup="mUp(this)"

style="background:green;width:90px;height:20px;padding:40px;">

Click Me</div>

</body>

</html>

**Interview Questions**

**1.What is JavaScript?**

JavaScript is a lightweight, interpreted programming language with object-oriented capabilities that allows you to build interactivity into otherwise static HTML pages.The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

**2.Name some of the JavaScript features.**

Following are the features of JavaScript −

* JavaScript is a lightweight, interpreted programming language.
* JavaScript is designed for creating network-centric applications.
* JavaScript is complementary to and integrated with Java.
* JavaScript is is complementary to and integrated with HTML.
* JavaScript is open and cross-platform.

**3.What are the advantages of using JavaScript?**

Following are the advantages of using JavaScript −

* Less server interaction − You can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
* Immediate feedback to the visitors − They don't have to wait for a page reload to see if they have forgotten to enter something.
* Increased interactivity − You can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
* Richer interfaces − You can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

**4.What are disadvantages of using JavaScript?**

We can not treat JavaScript as a full fledged programming language. It lacks the following important features −

* Client-side JavaScript does not allow the reading or writing of files. This has been kept for security reason.
* JavaScript can not be used for Networking applications because there is no such support available.
* JavaScript doesn't have any multithreading or multiprocess capabilities.

**5.Is JavaScript a case-sensitive language?**

Yes! JavaScript is a case-sensitive language. This means that language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters.

**6.How can you create an Object in JavaScript?**

JavaScript supports Object concept very well. You can create an object using the object literal as follows −

var emp = {

name: "Zara",

age: 10

};

**7.How can you read properties of an Object in JavaScript?**

You can write and read properties of an object using the dot notation as follows −

// Getting object properties

emp.name // ==> Zara

emp.age // ==> 10

// Setting object properties

emp.name = "Daisy" // <== Daisy

emp.age = 20 // <== 20

**8.How can you create an Array in JavaScript?**

You can define arrays using the array literal as follows −

var x = [];

var y = [1, 2, 3, 4, 5];

**9.How to read elements of an array in JavaScript?**

An array has a length property that is useful for iteration. We can read elements of an array as follows −

var x = [1, 2, 3, 4, 5];

for (var i = 0; i < x.length; i++) {

// Do something with x[i]

}

**10.What is a named function in JavaScript? How to define a named function?**

A named function has a name when it is defined. A named function can be defined using function keyword as follows −

function named(){

// do some stuff here

}

**11.How many types of functions JavaScript supports?**

A function in JavaScript can be either named or anonymous.

**12.How to define a anonymous function?**

An anonymous function can be defined in similar way as a normal function but it would not have any name.

**13.Can you assign a anonymous function to a variable?**

Yes! An anonymous function can be assigned to a variable.

**14.Can you pass a anonymous function as an argument to another function?**

Yes! An anonymous function can be passed as an argument to another function.

**15.What is arguments object in JavaScript?**

JavaScript variable arguments represents the arguments passed to a function.

**16.How can you get the type of arguments passed to a function?**

Using typeof operator, we can get the type of arguments passed to a function. For example −

function func(x){

console.log(typeof x, arguments.length);

}

func(); //==> "undefined", 0

func(1); //==> "number", 1

func("1", "2", "3"); //==> "string", 3

**17.How can you get the total number of arguments passed to a function?**

Using arguments.length property, we can get the total number of arguments passed to a function. For example −

function func(x){

console.log(typeof x, arguments.length);

}

func(); //==> "undefined", 0

func(1); //==> "number", 1

func("1", "2", "3"); //==> "string", 3

**18.How can you get the reference of a caller function inside a function?**

The arguments object has a callee property, which refers to the function you're inside of. For example −

function func() {

return arguments.callee;

}

func(); // ==> func

**19.What is the purpose of 'this' operator in JavaScript?**

JavaScript famous keyword this always refers to the current context.

**20.What are the valid scopes of a variable in JavaScript?**

The scope of a variable is the region of your program in which it is defined. JavaScript variable will have only two scopes.

* Global Variables − A global variable has global scope which means it is visible everywhere in your JavaScript code.
* Local Variables − A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

**21.Which type of variable among global and local, takes precedence over other if names are same?**

A local variable takes precedence over a global variable with the same name.

**22.What is callback?**

A callback is a plain JavaScript function passed to some method as an argument or option. Some callbacks are just events, called to give the user a chance to react when a certain state is triggered.

**23.What is closure?**

Closures are created whenever a variable that is defined outside the current scope is accessed from within some inner scope.

**24.Give an example of closure?**

Following example shows how the variable counter is visible within the create, increment, and print functions, but not outside of them −

function create() {

var counter = 0;

return {

increment: function() {

counter++;

},

print: function() {

console.log(counter);

}

}

}

var c = create();

c.increment();

c.print(); // ==> 1

**25.Which built-in method returns the character at the specified index?**

charAt() method returns the character at the specified index.

**26.Which built-in method combines the text of two strings and returns a new string?**

concat() method returns the character at the specified index.

**27.Which built-in method calls a function for each element in the array?**

forEach() method calls a function for each element in the array.

**28.Which built-in method returns the index within the calling String object of the first occurrence of the specified value?**

indexOf() method returns the index within the calling String object of the first occurrence of the specified value, or −1 if not found.

**29.Which built-in method returns the length of the string?**

length() method returns the length of the string.

**30.Which built-in method removes the last element from an array and returns that element?**

pop() method removes the last element from an array and returns that element.

**31.Which built-in method adds one or more elements to the end of an array and returns the new length of the array?**

push() method adds one or more elements to the end of an array and returns the new length of the array.

**32.Which built-in method reverses the order of the elements of an array?**

reverse() method reverses the order of the elements of an array −− the first becomes the last, and the last becomes the first.

**33.Which built-in method sorts the elements of an array?Which built-in method returns the characters in a string beginning at the specified location?**

substr() method returns the characters in a string beginning at the specified location through the specified number of characters.

**34.Which built-in method returns the calling string value converted to lower case?**

toLowerCase() method returns the calling string value converted to lower case.

**35.Which built-in method returns the calling string value converted to upper case?**

toUpperCase() method returns the calling string value converted to upper case.

**36.Which built-in method returns the string representation of the number's value?**

toString() method returns the string representation of the number's value.

**37.What are the variable naming conventions in JavaScript?**

While naming your variables in JavaScript keep following rules in mind.You should not use any of the JavaScript reserved keyword as variable name. These keywords are mentioned in the next section. For example, break or boolean variable names are not valid.JavaScript variable names should not start with a numeral (0-9). They must begin with a letter or the underscore character. For example, 123test is an invalid variable name but \_123test is a valid one.JavaScript variable names are case sensitive. For example, Name and name are two different variables.

**38.How typeof operator works?**

The typeof is a unary operator that is placed before its single operand, which can be of any type. Its value is a string indicating the data type of the operand.

The typeof operator evaluates to "number", "string", or "boolean" if its operand is a number, string, or boolean value and returns true or false based on the evaluation.

**39.What typeof returns for a null value?**

It returns "object".

**40.Can you access Cookie using javascript?**

JavaScript can also manipulate cookies using the cookie property of the Document object. JavaScript can read, create, modify, and delete the cookie or cookies that apply to the current web page.

**41.How to create a Cookie using JavaScript?**

The simplest way to create a cookie is to assign a string value to the document.cookie object, which looks like this −

Syntax −

document.cookie = "key1 = value1; key2 = value2; expires = date";

Here expires attribute is option. If you provide this attribute with a valid date or time then cookie will expire at the given date or time and after that cookies' value will not be accessible.

**42.How to read a Cookie using JavaScript?**

Reading a cookie is just as simple as writing one, because the value of the document.cookie object is the cookie. So you can use this string whenever you want to access the cookie.The document.cookie string will keep a list of name = value pairs separated by semicolons, where name is the name of a cookie and value is its string value.You can use strings' split() function to break the string into key and values.

**43.How to delete a Cookie using JavaScript?**

Sometimes you will want to delete a cookie so that subsequent attempts to read the cookie return nothing. To do this, you just need to set the expiration date to a time in the past.

**44.How to redirect a url using JavaScript?**

This is very simple to do a page redirect using JavaScript at client side. To redirect your site visitors to a new page, you just need to add a line in your head section as follows −

<head>

<script type="text/javascript">

<!--

window.location="http://www.newlocation.com";

//-->

</script>

</head>

**45.How to print a web page using javascript?**

JavaScript helps you to implement this functionality using print function of window object. The JavaScript print function window.print() will print the current web page when executed.

**46.What is Date object in JavaScript?**

The Date object is a datatype built into the JavaScript language. Date objects are created with the new Date( ).Once a Date object is created, a number of methods allow you to operate on it. Most methods simply allow you to get and set the year, month, day, hour, minute, second, and millisecond fields of the object, using either local time or UTC (universal, or GMT) time.

**47.What is Number object in JavaScript?**

he Number object represents numerical date, either integers or floating-point numbers. In general, you do not need to worry about Number objects because the browser automatically converts number literals to instances of the number class.

Syntax −

Creating a number object −

var val = new Number(number);

If the argument cannot be converted into a number, it returns NaN (Not-a-Number).

**48.How to handle exceptions in JavaScript?**

The latest versions of JavaScript added exception handling capabilities. JavaScript implements the try...catch...finally construct as well as the throw operator to handle exceptions.You can catch programmer-generated and runtime exceptions, but you cannot catch JavaScript syntax errors.

**49.What is purpose of onError event handler in JavaScript?**

The onerror event handler was the first feature to facilitate error handling for JavaScript. The error event is fired on the window object whenever an exception occurs on the page.The onerror event handler provides three pieces of information to identify the exact nature of the error −

* Error message − The same message that the browser would display for the given error.
* URL − The file in which the error occurred.
* Line number − The line number in the given URL that caused the error.

**50.What XML stands for?**

XML stands for Extensible Markup Language.

**51.What are the advantages of using XML?**

Following are the advantages that XML provides −

* Technology agnostic - Being plain text, XML is technology independent. It can be used by any technology for data storage and transmission purpose.
* Human readable- XML uses simple text format. It is human readable and understandable.
* Extensible - in XML, custom tags can be created and used very easily.
* Allow Validation - Using XSD, DTD and XML structure can be validated easily.

**52.What are the disadvantages of using XML?**

Following are the disadvantages of XML usage −

* Redundant Syntax - Normally XML file contains lot of repeatitive terms.
* Verbose-Being a verbose language, XML file size increases the transmission and storage costs.

**53.What is XML Parsing?**

Parsing XML refers to going through XML document to access data or to modify data in one or other way.

**54.What is XML Parser?**

XML Parser provides way how to access or modify data present in an XML document. Java provides multiple options to parse XML document.

**55.Name some of the parsers which are commonly used to parse XML documents.**

Following are various types of parsers which are commonly used to parse XML documents −

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

**56.What is DOM?**

DOM stands for Document Object Model and it is an official recommendation of the World Wide Web Consortium (W3C). It defines an interface that enables programs to access and update the style, structure,and contents of XML documents. XML parsers that support the DOM implement that interface.

**57.When to use a DOM Parser?**

You should use a DOM parser when −

* You need to know a lot about the structure of a document
* You need to move parts of the document around (you might want to sort certain elements, for example)
* You need to use the information in the document more than once

**58.What DOM Parser returns?**

When you parse an XML document with a DOM parser, you get back a tree structure that contains all of the elements of your document. The DOM provides a variety of functions you can use to examine the contents and structure of the document.

**59.What are the advantages of DOM Parsing?**

The DOM is a common interface for manipulating document structures. One of its design goals is that Java code written for one DOM-compliant parser should run on any other DOM-compliant parser without changes.

**60.What are the key components/interfaces of DOM Parsing?**

The DOM defines several Java interfaces. Here are the most common interfaces −

* Node - The base datatype of the DOM.
* Element - The vast majority of the objects you'll deal with are Elements.
* Attr Represents an attribute of an element.
* Text The actual content of an Element or Attr.
* Document Represents the entire XML document. A Document object is often referred to as a DOM tree.

**61.Name some of the important DOM parsing methods.**

When you are working with the DOM, there are several methods you'll use often −

* Document.getDocumentElement() - Returns the root element of the document.
* Node.getFirstChild() - Returns the first child of a given Node.
* Node.getLastChild() - Returns the last child of a given Node.
* Node.getNextSibling() - These methods return the next sibling of a given Node.
* Node.getPreviousSibling() - These methods return the previous sibling of a given Node.
* Node.getAttribute(attrName) - For a given Node, returns the attribute with the requested name.

**62.Can we create an XML document using DOM parser?**

Yes! Using DOM parser, we can parse, modify or create a XML document.

**63.What SAX stands for?**

SAX stands for Simple API for XML.

**64.What is a SAX Parser?**

SAX Parser is an event-based parser for xml documents.

**65.How a SAX Parser works?**

SAX (the Simple API for XML) is an event-based parser for xml documents.Unlike a DOM parser, a SAX parser creates no parse tree. SAX is a streaming interface for XML, which means that applications using SAX receive event notifications about the XML document being processed an element, and attribute, at a time in sequential order starting at the top of the document, and ending with the closing of the ROOT element.

**66.When to use a SAX Parser?**

You should use a SAX parser when −

* You can process the XML document in a linear fashion from the top down
* The document is not deeply nested
* You are processing a very large XML document whose DOM tree would consume too much memory.Typical DOM implementations use ten bytes of memory to represent one byte of XML
* The problem to be solved involves only part of the XML document
* Data is available as soon as it is seen by the parser, so SAX works well for an XML document that arrives over a stream

**67.What are the disadvantages of SAX Parsing?**

* We have no random access to an XML document since it is processed in a forward-only manner
* If you need to keep track of data the parser has seen or change the order of items, you must write the code and store the data on your own

**68.Name some of the important SAX parsing methods.**

ContentHandler Interface specifies the callback methods that the SAX parser uses to notify an application program of the components of the XML document that it has seen.

* void startDocument() - Called at the beginning of a document.
* void endDocument() - Called at the end of a document.
* void startElement(String uri, String localName, String qName, Attributes atts) - Called at the beginning of an element.
* void endElement(String uri, String localName,String qName) - Called at the end of an element.
* void characters(char[] ch, int start, int length) - Called when character data is encountered.
* void ignorableWhitespace( char[] ch, int start, int length) - Called when a DTD is present and ignorable whitespace is encountered.
* void processingInstruction(String target, String data) - Called when a processing instruction is recognized.
* void setDocumentLocator(Locator locator)) - Provides a Locator that can be used to identify positions in the document.
* void skippedEntity(String name) - Called when an unresolved entity is encountered.
* void startPrefixMapping(String prefix, String uri) - Called when a new namespace mapping is defined.
* void endPrefixMapping(String prefix) - Called when a namespace definition ends its scope.

**69.Name some methods for processing the attributes connected to an element in SAX parsing.**

Attributes Interface specifies methods for processing the attributes connected to an element.

* int getLength() - Returns number of attributes.
* String getQName(int index)
* String getValue(int index)
* String getValue(String qname)

**70.Can we create an XML document using SAX parser?**

No! Using SAX parser, we can only parse or modify a XML document.

**71.What is JDOM Parser?**

JDOM is an open source, java based library to parse XML document and it is typically java developer friendly API.

**72.What are the benefits of JDOM parser?**

It is java optimized, it uses java collection like List and Arrays. It works with DOM and SAX APIs and combines the best of the two. It is of low memory footprint and is nearly as fast as SAX.

**73.When to use a JDOM Parser?**

You should use a JDOM parser when −

* You need to know a lot about the structure of a document.
* You need to move parts of the document around (you might want to sort certain elements, for example).
* You need to use the information in the document more than once.
* You are a java developer and want to leverage java optimized parsing of XML.

**74.What are the advantages of JDOM parser?**

When you parse an XML document with a JDOM parser, you get the flexibility to get back a tree structure that contains all of the elements of your document without impacting the memory footprint of the application. The JDOM provides a variety of utility functions you can use to examine the contents and structure of the document in case document is well structured and its structure is known.JDOM gives java developers flexibility and easy maintainablity of xml parsing code. It is light weight and quick API.

**75.Name some of the important JDOM classes.**

The JDOM defines several Java classes. Here are the most common classes −

* Document - Represents the entire XML document. A Document object is often referred to as a DOM tree.
* Element - Represents an XML element. Element object has methods to manipulate its child elements,its text, attributes and namespaces.
* Attribute Represents an attribute of an element. Attribute has method to get and set the value of attribute. It has parent and attribute type.
* Text Represents the text of XML tag.
* Comment Represents the comments in a XML document.

**76.Name some of the important JDOM methods.**

When you are working with the JDOM, there are several methods you'll use often −

* SAXBuilder.build(xmlSource) - Build the JDOM document from the xml source.
* Document.getRootElement() - Get the root element of the XML.
* Element.getName() - Get the name of the XML node.
* Element.getChildren() - Get all the direct child nodes of an element.
* Node.getChildren(Name) - Get all the direct child nodes with a given name.
* Node.getChild(Name) - Get first child node with given name.

**77.Can we create an XML document using JDOM parser?**

Yes! Using JDOM parser, we can parse, modify and create a XML document.

**78.What is a StAX Parser?**

StAX is a JAVA based API to parse XML document in a similar way as SAX parser does but StAX is a PULL API where as SAX is a PUSH API. It means in case of StAX parser, client application need to ask StAX parser to get information from XML whenever it needs but in case of SAX parser, client application is required to get information when SAX parser notifies the client application that information is available.

**79.Can we create an XML document using StAX parser?**

Yes! Using StAX parser, we can parse, modify and create a XML document.

**80.Is StAX parser a PULL API?**

Yes! StAX is a PULL API.

**81.When you should use a StAX parser?**

You should use a StAX parser when −

* You can process the XML document in a linear fashion from the top down.
* The document is not deeply nested.
* You are processing a very large XML document whose DOM tree would consume too much memory. Typical DOM implementations use ten bytes of memory to represent one byte of XML.
* The problem to be solved involves only part of the XML document.
* Data is available as soon as it is seen by the parser, so StAX works well for an XML document that arrives over a stream.

**82.What are the disadvantages of StAX parser?**

* We have no random access to an XML document since it is processed in a forward-only manner
* If you need to keep track of data the parser has seen or change the order of items, you must write the code and store the data on your own

**83.Explain XMLEventReader Class of StAX parser.**

This class provide iterator of events which can be used to iterate over events as they occur while parsing the XML document.

* StartElement asStartElement() - used to retrieve value and attributes of element.
* EndElement asEndElement() - called at the end of a element.
* Characters asCharacters() - can be used to obtain characters such a CDATA, whitespace etc.

**84.Explain XMLEventWriter Class of StAX parser.**

This interface specifies methods for creating an event.

* add(Event event) - Add event containing elements to XML.

**85.Explain XMLStreamReader Class of StAX parser.**

This class provide iterator of events which can be used to iterate over events as they occur while parsing the XML document

* int next() - used to retrieve next event.
* boolean hasNext() - used to check further events exists or not
* String getText() - used to get text of an element
* String getLocalName() - used to get name of an element

**86.Explain XMLStreamWriter Class of StAX parser.**

This interface specifies methods for creating an event.

* writeStartElement(String localName) - Add start element of given name.
* writeEndElement(String localName) - Add end element of given name.
* writeAttribute(String localName, String value) - Write attribute to an element.

**87.What is XPath?**

The XPath is an official recommendation of the World Wide Web Consortium (W3C). It defines a language to find information in an XML file. It is used to traverse elements and attributes of an XML document. XPath provides various type of expressions which can be used to enquire relevant information from the XML document.

**88.What are the key components of XPath?**

Following are the key components of XPath −

* Structure Definitions - XPath defines the parts of an XML document like element, attribute, text, namespace, processing-instruction, comment, and document nodes.
* Path Expressions XPath provides powerful path expressions select nodes or list of nodes in XML documents.
* Standard FunctionsXPath provides a rich library of standard functions for manipulation of string values, numeric values, date and time comparison, node and QName manipulation, sequence manipulation, Boolean values etc.
* Major part of XSLTXPath is one of the major element in XSLT standard and is must have knowledge in order to work with XSLT documents.
* W3C recommendationXPath is official recommendation of World Wide Web Consortium (W3C).

**89.What is predicate in XPath?**

Predicate are used to find specific node or a node containing specific value and are defined using [...] .

|  |  |
| --- | --- |
| Expression | Result |
| /class/student[1] | Selects the first student element that is the child of the class element. |
| /class/student[last()] | Selects the last student element that is the child of the class element. |
| /class/student[last()-1] | Selects the last but one student element that is the child of the class element. |
| //student[@rollno='493'] | Selects all the student elements that have an attribute named rollno with a value of '493' |

**90.What is path expression in XPath?**

XPath uses a path expression to select node or list of nodes from an xml document. Following is the list of useful paths and expression to select any node/ list of nodes from an xml document.

|  |  |
| --- | --- |
| Expression | Description |
| node-name | Select all nodes with the given name "nodename" |
| / | Selection starts from the root node |
| // | Selection starts from the current node that match the selection |
| . | Selects the current node |
| .. | Selects the parent of the current node |
| @ | Selects attributes |
| student | Example − Selects all nodes with the name "student" |
| class/student | Example: Selects all student elements that are children of class |
| //student | Selects all student elements no matter where they are in the document |

**91.Can we create an XML document using XPath parser?**

No! XPath parser is used to to navigate XML Document only. It is better to use DOM parser for creating XML.

**92.What is DOM4J Parser?**

DOM4J is an open source, java based library to parse XML document and it is highly flexible, high-performance, and memory-efficient API. It is java optimized, it uses java collection like List and Arrays. It works with DOM, SAX, XPath and XSLT. It can parse large XML document with very low memory footprint.

**93.When to use a DOM4J Parser?**

You should use a DOM4J parser when −

* You need to know a lot about the structure of a document
* You need to move parts of the document around (you might want to sort certain elements, for example)
* You need to use the information in the document more than once
* You are a java developer and want to leverage java optimized parsing of XML.

**94.When are the benefits of using a DOM4J Parser?**

When you parse an XML document with a DOM4J parser, you get the flexibility to get back a tree structure that contains all of the elements of your document without impacting the memory footprint of the application. The DOM4J provides a variety of utility functions you can use to examine the contents and structure of the document in case document is well structured and its structure is known. DOM4J uses XPath expression to navigate through the XML document.

**95.When are the advantages of using a DOM4J Parser?**

DOM4J gives java developers flexibility and easy maintainablity of xml parsing code. It is light weight and quick API.

**96.Name some of the important DOM4J classes.**

The DOM4J defines several Java classes. Here are the most common classes −

* Document - Represents the entire XML document. A Document object is often referred to as a DOM tree.
* Element - Represents an XML element. Element object has methods to manipulate its child elements,its text, attributes and namespaces.
* Attribute Represents an attribute of an element. Attribute has method to get and set the value of attribute. It has parent and attribute type.
* Node Represents Element, Attribute or ProcessingInstruction

**97.Name some of the important DOM4J methods.**

When you are working with the DOM4J, there are several methods you'll use often −

* SAXReader.read(xmlSource)() - Build the DOM4J document from the xml source.
* Document.getRootElement() - Get the root element of the XML.
* Element.node(index) - Get the XML node at particular index in the element.
* Element.attributes() - Get all the attributes of an element.
* Node.valueOf(@Name) - Get the value of an attribute with given name of the element.

**98.Can we create an XML document using DOM4J parser?**

Yes! Using DOM4J parser, we can parse, modify and create a XML document.

**99.Which of the following is true about variable naming conventions in JavaScript?**

A-[You should not use any of the JavaScript reserved keyword as variable name.](javascript:void(0);)

[B - JavaScript variable names should not start with a numeral (0-9).](javascript:void(0);)

C - Both of the above.

[D - None of the above.](javascript:void(0);)

### Answer : C

### Explanation

Both of the above options are correct.

**100.Can you pass a anonymous function as an argument to anotherfunction?**

A - tru

[B - fale](javascript:void(0);)

### Answer : A

### Explanation

Yes! An anonymous function can be passed as an argument to another function.

**101.Which built-in method adds one or more elements to the end of an array and returns the new length of the array?**

[A - last()](javascript:void(0);)

[B - put()](javascript:void(0);)

C - push()

[D - None of the above.](javascript:void(0);)

### Answer : C

### Explanation

push() method adds one or more elements to the end of an array and returns the new length of the array.

**102.Which built-in method returns the string representation of the number's value?**

[A - toValue()](javascript:void(0);)

[B - toNumber()](javascript:void(0);)

C - toString()

[D - None of the above.](javascript:void(0);)

### Answer : C

### Explanation

toString() method returns the string representation of the number's value.

**103.Which of the following function of Boolean object returns the primitive value of the Boolean object?**

[A - toSource()](javascript:void(0);)

B - valueOf()

[C - toString()](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

### Answer : B

### Explanation

valueOf() − Returns the primitive value of the Boolean object.

**104.Which of the following function of String object returns a number indicating whether a reference string comes before or after or is the same as the given string in sort order?**

A - localeCompare()

[B - search()](javascript:void(0);)

[C - substr()](javascript:void(0);)

[D - concat()](javascript:void(0);)

### Answer : A

### Explanation

localeCompare() − Returns a number indicating whether a reference string comes before or after or is the same as the given string in sort order.

**105.Which of the following function of String object creates a string to be displayed as bold as if it were in a <b> tag?**

[A - anchor()](javascript:void(0);)

[B - big()](javascript:void(0);)

[C - blink()](javascript:void(0);)

D - bold()

### Answer : D

### Explanation

bold() − Creates a string to be displayed as bold as if it were in a <b> tag.

**106.Which of the following function of String object creates an HTML hypertext link that requests another URL?**

A - link()

[B - sub()](javascript:void(0);)

[C - sup()](javascript:void(0);)

[D - small()](javascript:void(0);)

### Answer : A

### Explanation

link() − Creates an HTML hypertext link that requests another URL.

**107.Which of the following function of Array object adds one or more elements to the end of an array and returns the new length of the array?**

[A - pop()](javascript:void(0);)

B - push()

[C - join()](javascript:void(0);)

[D - map()](javascript:void(0);)

### Answer : B

### Explanation

push() − Adds one or more elements to the end of an array and returns the new length of the array.

**108.Which of the following function of Array object adds one or more elements to the front of an array and returns the new length of the array?**

A - unshift()

[B - sort()](javascript:void(0);)

[C - splice()](javascript:void(0);)

[D - toString()](javascript:void(0);)

### Answer : A

### Explanation

unshift() − Adds one or more elements to the front of an array and returns the new length of the array.

**109.What XML stands for?**

[A - Extra Marked Language](javascript:void(0);)

B - Extensible Markup Language

[C - Extensible Marked Literal](javascript:void(0);)

[D - Extra Markup Language](javascript:void(0);)

### Answer : B

### Explaination

XML stands for Extensible Markup Language.

**110.Which of the following method returns the root element of the document in DOM Parsing?**

[A - Node.getRoot()](javascript:void(0);)

B - Document.getDocumentElement()

[C - Node.getFirstChild()](javascript:void(0);)

[D - Node.getLastChild()](javascript:void(0);)

### Answer : B

### Explaination

Document.getDocumentElement() returns the root element of the document in DOM Parsing.

**111.Which of the following method get called when element starts in SAX parsing?**

[A - startDocument()](javascript:void(0);)

[B - endDocument()](javascript:void(0);)

C - startElement()

[D - endElement()](javascript:void(0);)

**112.Which method of JDOM Parser gets the root element of the XML?**

[A - Element.getRootElement()](javascript:void(0);)

B - Document.getRootElement()

[C - Node.getRootElement()](javascript:void(0);)

[D - Node.getChild(Name)](javascript:void(0);)

**113.XPath provides a rich library of standard functions for manipulation of string values, numeric values, date and time comparison, node and QName manipulation, sequence manipulation, Boolean values etc.**

A - true

[B - false](javascript:void(0);)

### Answer : A

### Explaination

XPath provides a rich library of standard functions for manipulation of string values, numeric values, date and time comparison, node and QName manipulation, sequence manipulation, Boolean values etc.

**114.Which of the following XPath expression ensures that selectionstarts from the current node that match the selection?**

[A - .](javascript:void(0);)

[B - /](javascript:void(0);)

[C - ./](javascript:void(0);)

D - //

### Answer : D

### Explaination

// ensures that selection starts from the current node that match the selection.

**115.Which of the following method of DOM4J Parser gets all the attributes of an element?**

A - Element.attributes()

[B - Dom4j.attributes()](javascript:void(0);)

[C - Document.attributes()](javascript:void(0);)

[D - Node.attributes()](javascript:void(0);)

### Answer : A

### Explaination

Element.attributes() gets all the attributes of an element.

**116.XML is Technology agnostic.**

A - true

[B - false](javascript:void(0);)

### Answer : A

### Explaination

Being plain text, XML is technology independent. It can be used by any technology for data storage and transmission purpose.

**117.Is StAX parser a PULL API?**

A - true

[B - false](javascript:void(0);)

### Answer : A

### Explaination

StAX parser is a PULL API. It means in case of StAX parser, client application need to ask StAX parser to get information from XML whenever it needs.

**118.Which method of the following of StAX Parser can be used to obtain characters such a CDATA, whitespace etc?**

[A - StartElement asStartElement()](javascript:void(0);)

[B - EndElement asEndElement()](javascript:void(0);)

C - Characters asCharacters()

[D - None of the above.](javascript:void(0);)

### Answer : C

### Explaination

Characters asCharacters() of XMLEventReader class can be used to obtain characters such a CDATA, whitespace etc.

**119.Which of the following is correct about JavaScript?**

[A - JavaScript is a lightweight, interpreted programming language.](javascript:void(0);)

[B - JavaScript has object-oriented capabilities that allows you to build interactivity into otherwise static HTML pages.](javascript:void(0);)

[C - The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.](javascript:void(0);)

[D - All of the above.](javascript:void(0);)

**120.Which of the following is correct about features of JavaScript?**

[A - JavaScript is a lightweight, interpreted programming language.](javascript:void(0);)

[B - JavaScript is designed for creating network-centric applications.](javascript:void(0);)

[C - JavaScript is complementary to and integrated with Java.](javascript:void(0);)

[D - All of the above.](javascript:void(0);)

**121.Which of the following is correct about features of JavaScript?**

[A - JavaScript is is complementary to and integrated with HTML.](javascript:void(0);)

[B - JavaScript is open and cross-platform.](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - All of the above.](javascript:void(0);)

**122.Which of the following is an advantage of using JavaScript?**

[A - Less server interaction](javascript:void(0);)

[B - Immediate feedback to the visitors](javascript:void(0);)

[C - Increased interactivity](javascript:void(0);)

[D - All of the above.](javascript:void(0);)

**123.Which of the following is a disadvantage of using JavaScript?**

[A - Client-side JavaScript does not allow the reading or writing of files.](javascript:void(0);)

[B - JavaScript can not be used for Networking applications because there is no such support available.](javascript:void(0);)

[C - JavaScript doesn't have any multithreading or multiprocess capabilities.](javascript:void(0);)

[D - All of the above.](javascript:void(0);)

**124.Is JavaScript a case-sensitive language?**

[A - true](javascript:void(0);)

[B - false](javascript:void(0);)

**125.Which of the following is true about variable naming conventions in JavaScript?**

[A - You should not use any of the JavaScript reserved keyword as variable name.](javascript:void(0);)

[B - JavaScript variable names should not start with a numeral (0-9).](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**126.Which of the following is true about variable naming conventions in JavaScript?**

[A - JavaScript variable names must begin with a letter or the underscore character.](javascript:void(0);)

[B - JavaScript variable names are case sensitive.](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**127.Which of the following is true about typeof operator in JavaScript?**

[A - The typeof is a unary operator that is placed before its single operand, which can be of any type.](javascript:void(0);)

[B - Its value is a string indicating the data type of the operand.](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**128.Can you access Cookie using javascript?**

[A - true](javascript:void(0);)

[B - false](javascript:void(0);)

**129.Which of the following is true about cookie handling in JavaScript?**

[A - JavaScript can manipulate cookies using the cookie property of the Document object.](javascript:void(0);)

[B - JavaScript can read, create, modify, and delete the cookie or cookies that apply to the current web page.](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**130.Which of the following is the correct syntax to create a cookie using JavaScript?**

[A - document.cookie = 'key1 = value1; key2 = value2; expires = date';](javascript:void(0);)

[B - browser.cookie = 'key1 = value1; key2 = value2; expires = date';](javascript:void(0);)

[C - window.cookie = 'key1 = value1; key2 = value2; expires = date';](javascript:void(0);)

[D - navigator.cookie = 'key1 = value1; key2 = value2; expires = date';](javascript:void(0);)

**131.Which of the following is the correct syntax to redirect a url using JavaScript?**

[A - document.location='http://www.newlocation.com';](javascript:void(0);)

[B - browser.location='http://www.newlocation.com';](javascript:void(0);)

[C - navigator.location='http://www.newlocation.com';](javascript:void(0);)

[D - window.location='http://www.newlocation.com';](javascript:void(0);)

**132.Which of the following is the correct syntax to print a page using JavaScript?**

[A - window.print();](javascript:void(0);)

[B - browser.print();](javascript:void(0);)

[C - navigator.print();](javascript:void(0);)

[D - document.print();](javascript:void(0);)

**133.Which of the following is a valid type of function javascript supports?**

[A - named function](javascript:void(0);)

[B - anonymous function](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**134.Can you assign a anonymous function to a variable?**

[A - true](javascript:void(0);)

[B - false](javascript:void(0);)

**135.Can you pass a anonymous function as an argument to another function?**

[A - true](javascript:void(0);)

[B - false](javascript:void(0);)

**136.How can you get the type of arguments passed to a function?**

[A - using typeof operator](javascript:void(0);)

[B - using getType function](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**137.How can you get the total number of arguments passed to a function?**

[A - Using args.length property](javascript:void(0);)

[B - Using arguments.length property](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**138.Which of the following type of variable is visible everywhere in your JavaScript code?**

[A - global variable](javascript:void(0);)

[B - local variable](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**139.Which of the following type of variable is visible only within a function where it is defined?**

[A - global variable](javascript:void(0);)

[B - local variable](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**140.Which of the following type of variable takes precedence over other if names are same?**

[A - global variable](javascript:void(0);)

[B - local variable](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**141.Which of the following is correct about callbacks?**

[A - A callback is a plain JavaScript function passed to some method asan argument or option.](javascript:void(0);)

[B- Some callbacks are just events, called to give the user a chance toreact when a certain state is triggered.](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**142.Which built-in method returns the character at the specified index?**

[A - characterAt()](javascript:void(0);)

[B - getCharAt()](javascript:void(0);)

[C - charAt()](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**143.Which built-in method combines the text of two strings and returns a new string?**

[A - append()](javascript:void(0);)

[B - concat()](javascript:void(0);)

[C - attach()](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**144.What XML stands for?**

[A - Extra Marked Language](javascript:void(0);)

[B - Extensible Markup Language](javascript:void(0);)

[C - Extensible Marked Literal](javascript:void(0);)

[D - Extra Markup Language](javascript:void(0);)

**145.Which of the following is true about XML?**

[A - XML is technology independent. It can be used by any technology for data storage and transmission purpose.](javascript:void(0);)

[B - XML uses simple text format. It is human readable and understandable.](javascript:void(0);)

[C - Using XSD, DTD and XML structure can be validated easily.](javascript:void(0);)

[D - All of the above.](javascript:void(0);)

**146.Which of the following is true about XML?**

[A - XML is a markup language.](javascript:void(0);)

[B - XML is a tag based language like HTML.](javascript:void(0);)

[C - XML tags are not predefined like HTML.](javascript:void(0);)

[D - All of the above.](javascript:void(0);)

**147.Which of the following parses the document by loading the complete contents of the document and creating its complete hierarchical tree in memory?**

[A - Dom Parser](javascript:void(0);)

[B - SAX Parser](javascript:void(0);)

[C - JDOM Parser](javascript:void(0);)

[D - StAX Parser](javascript:void(0);)

**148.Which of the following parses the document on event based triggers and does not load the complete document into the memory?**

[A - Dom Parser](javascript:void(0);)

[B - SAX Parser](javascript:void(0);)

[C - JDOM Parser](javascript:void(0);)

[D - StAX Parser](javascript:void(0);)

**149.Which of the following parses the document in similar fashion to DOM parser but in more easier way?**

[A - Dom Parser](javascript:void(0);)

[B - SAX Parser](javascript:void(0);)

[C - JDOM Parser](javascript:void(0);)

[D - StAX Parser](javascript:void(0);)

**150.Which of the following parses the document in similar fashion toSAX parser but in more efficient way?**

[A - Dom Parser](javascript:void(0);)

[B - SAX Parser](javascript:void(0);)

[C - JDOM Parser](javascript:void(0);)

[D - StAX Parser](javascript:void(0);)

**151.Which of the following parses the XML based on expression and is used extensively in conjuction with XSLT?**

[A - XPath Parser](javascript:void(0);)

[B - SAX Parser](javascript:void(0);)

[C - JDOM Parser](javascript:void(0);)

[D - StAX Parser](javascript:void(0);)

**152.Which of the following provides options to parse XML, XPath and XSLT using Java Collections Framework and provides support for DOM, SAX and JAXP?**

[A - XPath Parser](javascript:void(0);)

[B - DOM4J Parser](javascript:void(0);)

[C - JDOM Parser](javascript:void(0);)

[D - StAX Parser](javascript:void(0);)

**153.Which of the following component represent base datatype of the DOM in DOM Parsing?**

[A - Node](javascript:void(0);)

[B - Element](javascript:void(0);)

[C - Attr](javascript:void(0);)

[D - Document](javascript:void(0);)

**154.Which of the following component represent actual content of an element or attribute in DOM Parsing?**

[A - Text](javascript:void(0);)

[B - Element](javascript:void(0);)

[C - Attr](javascript:void(0);)

[D - Document](javascript:void(0);)

**155.Which of the following method returns the root element of the document in DOM Parsing?**

[A - Node.getRoot()](javascript:void(0);)

[B - Document.getDocumentElement()](javascript:void(0);)

[C - Node.getFirstChild()](javascript:void(0);)

[D - Node.getLastChild()](javascript:void(0);)

**156.Which of the following method returns the first child of a given Node in DOM Parsing?**

[A - Node.getChild()](javascript:void(0);)

[B - Document.getFirstChild()](javascript:void(0);)

[C - Node.getFirstChild()](javascript:void(0);)

[D - Node.getLastChild()](javascript:void(0);)

**157.What DOM stands for?**

[A - Direct Object Model](javascript:void(0);)

[B - Document Object Modeling](javascript:void(0);)

[C - Document Object Model](javascript:void(0);)

[D - Document Output Model](javascript:void(0);)

**158.Which of the following is true about DOM?**

[A - It is an official recommendation of the World Wide Web Consortium (W3C).](javascript:void(0);)

[B - It defines an interface that enables programs to access and update the style, structure,and contents of XML documents.](javascript:void(0);)

[C - XML parsers that support the DOM implement DOM interface.](javascript:void(0);)

[D - All of the above.](javascript:void(0);)

**159.Which of the following is true about DOM Parser?**

[A - When you parse an XML document with a DOM parser, you get back a tree structure that contains all of the elements of your document.](javascript:void(0);)

[B - The DOM Parser provides a variety of functions you can use to examine the contents and structure of the document.](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**160.Can we create an XML document using DOM parser?**

[A - true](javascript:void(0);)

[B - false](javascript:void(0);)

**161.What SAX stands for?**

[A - Solve Analyze XML](javascript:void(0);)

[B - Simple API for XML](javascript:void(0);)

[C - Streaming API for XML](javascript:void(0);)

[D - Simple Analyze XML](javascript:void(0);)

**162.Which of the following is true about SAX parsing?**

[A - SAX is an event-based parser for xml documents.](javascript:void(0);)

[B - Unlike a DOM parser, a SAX parser creates no parse tree.](javascript:void(0);)

[C - Both of the above.](javascript:void(0);)

[D - None of the above.](javascript:void(0);)

**163.Which of the following is true about SAX parsing?**

[A - SAX parser reads an XML document from top to bottom,recognizing the tokens that make up a well-formed XML document.](javascript:void(0);)

[B - SAX parser reports the application program the nature of tokens that the parser has encountered as they occur](javascript:void(0);)

[C- The application program provides an 'event' handler that must beregistered with the parser.](javascript:void(0);)

[D - All of the above.](javascript:void(0);)

**164.Q 21 - Can we create an XML document using SAX parser?**

[A - true](javascript:void(0);)

[B - false](javascript:void(0);)

**165.Which of the following method get called when document beginsin SAX parsing?**

[A - startDocument()](javascript:void(0);)

[B - endDocument()](javascript:void(0);)

[C - startElement()](javascript:void(0);)

[D - endElement()](javascript:void(0);)

**166.Which of the following method get called when document ends in SAX parsing?**

[A - startDocument()](javascript:void(0);)

[B - endDocument()](javascript:void(0);)

[C - startElement()](javascript:void(0);)

[D - endElement()](javascript:void(0);)

**167.Which of the following method get called when element starts inSAX parsing?**

[A - startDocument()](javascript:void(0);)

[B - endDocument()](javascript:void(0);)

[C - startElement()](javascript:void(0);)

[D - endElement()](javascript:void(0);)

**168.Which of the following method get called when element ends inSAX parsing?**

[A - startDocument()](javascript:void(0);)

[B - endDocument()](javascript:void(0);)

[C - startElement()](javascript:void(0);)

[D - endElement()](javascript:void(0);)