|  |  |  |  |
| --- | --- | --- | --- |
| Sr.No | Date | Program | Remark |
| 1 |  | Write a program to create singly linear linked list for insert delete search print operation using menu driven program |  |
| 2 |  | Write a program to create doubly circular linked list for insert delete search print operation using menu driven program |  |
| 3 |  | Write a program to check the parenthesis of expression is well formedness or not Write a program to reverse the given string. |  |
| 4 |  | Write a program to create stack using linked list and manipulate it using menu driven code |  |
| 5 |  | Write a program to create CQ using array and manipulate it using menu driven code |  |
| 6 |  | Write a program to reverse stack using queue |  |
| 7 |  | Write a program to create Binary search tree and traverse it using recursive preorder, inorder, postorder methods. |  |
| 8 |  | Write a program to implement graph using adjacency matrix and traverse it using BFT and DFT |  |
| 9 |  | Write a program to implement linear probing hashing for insert and search operation |  |
| 10 |  | Write a program which implement the rain terraces to measure trapping rain water |  |
| 11 |  | Write a program to implements to implement Kruskals Algorithm MST |  |
| 12 |  | Write a program to implements to implement Prims Algorithm MST |  |
| 13 |  | Write a program to implement binary search algorithm using divide and conquer technique |  |
| 14 |  | Write a program to implement TOWER OF HANNOI problem |  |
| 15 |  | Write a program to find longest common sub sequence |  |
| 16 |  | Write a program to implement regular expression to check whether given pattern match with input string or not? |  |
| 17 |  | Write a program to implement N queen Problem using backtracking approach. |  |

**Q.1 ) Write a program to create singly linear linked list for insert delete search print operation using menu driven program.**

HTML Code:

<html>

<head>

<script type="text/javascript" src="SLL.js"></script>

</head>

<body>

Click here for the result

<input type="button" onclick="SLL()" value="Singly Linear Linked List" />

</body>

</html>

JS Code:

class Node

{ constructor(data,next=null)

{

this.data=data;

this.next=next;

}

}

class LinkedList

{ constructor()

{ this.head=null;//i m creating null singly linear linked list

this.size=0;

}

Create()

{ var i=1,item=0;

let oldnode=null;

let newnode=null;

do

{ item=parseInt(prompt("Enter Item "));

newnode=new Node(item);

if(newnode==null)

{ document.write("overflow");

alert("overflow");

}else

{

if(this.head==null)

{

this.head=newnode;

}else

{

oldnode.next=newnode;

}

oldnode=newnode;

this.size++;

i++;

}

}while(i<=n);

}

insertFirst(data)

{

alert(" in insert.first...");

this.head=new Node(data,this.head);

this.size++;

}

printAllNodes()

{

alert("in print");

document.write("<br> Linked List elements are :- <br>");

let ptr=this.head;

alert(" in print Size=..."+this.size);

while(ptr)

{

alert("in print"+ptr.data);

console.log(ptr.data);

document.writeln(ptr.data+":");

ptr=ptr.next;

}

document.write("<br>");

}

insertLast(data)

{ alert(" in insert.Last...");

let node=new Node(data);

let ptr;

if(this.head==null){this.head=node;}

else

{

ptr=this.head;

while(ptr.next)

{

ptr=ptr.next;

}

ptr.next=node;

}

this.size++;

}

insert(item,pos)

{ var i=1;

let newnode=null,ptr=null;

newnode= new Node(item);

if(newnode==null)

{ document.write("Linked list Overflow");

}

else

{ document.write("<br>insert Operation <br>")

if(pos==1)

{ newnode.next=this.head;

this.head=newnode;

}else

{ ptr=this.head;i=1;

while((i<pos-1)&&(ptr!=null))

{ i++;

ptr=ptr.next;

}

if(ptr!=null)

{ newnode.next=ptr.next;

ptr.next=newnode;

}

}

}

this.size++;

}

search(key)

{

var ptr = this.head;

var i= 1;

document.write("<br>");

do

{

if(ptr.data == key)

{

document.write(key + " is found at "+ i+"location ");

return;

}

ptr =ptr.next;

i=i+1;

}while(ptr != null);

document.write(sn+" is not found");

}

del(pos)

{ var i=1,item=0;

alert("within del...");

let delnode=null,ptr=null;

if(this.head==null)

{ document.write("linked list underflow");

}else

{

document.write("<br> Delete Operation <br>");

if (pos == 1)

{ alert("within del pos=1...");

delnode=this.head;

this.head=this.head.next;

item=delnode.data;

document.write("deleted item="+item);

}else

{ ptr=this.head;i=1;

while((i<pos-1)&&(ptr!=null))

{

i++;ptr=ptr.next;

}

if(ptr==null)

{ document.write("Invalid position");

}else

{ delnode=ptr.next;

item=delnode.data;

ptr.next=delnode.next;

}

}

}

this.size--;

}

}

function SLL()

{

const l1=new LinkedList();//creating sll

var ch=0,item=0,pos=0,n=0;

do

{

ch=parseInt(prompt("1:create 2:insert 3:delete 4:Print 5:search 7:exit ...Enter your choice"));

switch(ch)

{ case 1: n=parseInt(prompt("Enter total nodes to create"));

l1.create(n);break;

case 2: item=parseInt(prompt("Enter Item to insert"));

pos=parseInt(prompt("Enter position to insert"));

l1.insert(item,pos);break;

case 3: pos=parseInt(prompt("Enter position to delete"));

l1.del(pos);break;

case 4:l1.printAllNodes();break;

case 5:var key=parseInt(prompt("Enter element to search in linked list "));

l1.search(key); break;

case 7:break;

default:alert("Wrong choice");

}

document.write(" --->");

}while(ch!=7);

}

OUTPUT :-

--->

Linked List elements are :-

11: 12: 13: 14: 15:

--->

insert Operation

--->

Linked List elements are :-

11: 12: 3: 13: 14: 15:

--->

Delete Operation

---> --->

Linked List elements are :-

11: 12: 3: 13: 15:

--->

3 is found at 3location ---> ---> --->

**Q2) Write a program to create doubly circular linked list for insert delete search print operation using menu driven program**.

HTML Code:

<html>

<head>

<script type="text/javascript" src="DCL.js"></script>

</head>

<body>

Click here for the result

<input type="button" onclick="dcl()" value="Doubly Circular Linked List" />

</body>

</html>

JS Code:

class DENode

{ constructor(ec=0,nm=" ",s=0.0,prev=null,next=null)

{ this.prev=prev;

this.name=nm;

this.code=ec;

this.sal=s;

this.next=next;

}

}

class DC\_LinkedList

{ constructor()

{ this.start=null;//i m creating null linked list

this.size=0;

}

Create()

{ var i=1,ec=0,nm=" ",s=0.0;

let oldnode=null;

let newnode=null;

do

{ ec=parseInt(prompt("Enter Emp Code "));

nm=prompt("Enter Name ");

s=parseFloat(prompt("Enter Salary "));

newnode=new DENode(ec,nm,s);

if(newnode==null)

{ document.write("overflow<br />");

alert("overflow");

}else

{

if(this.start==null)

{

this.start=newnode;

}else

{

oldnode.next=newnode;

newnode.prev=oldnode;

}

oldnode=newnode;

this.size++;

i++;

}

newnode.next=this.start;

this.start.prev=newnode;

}while(i<=n);

}

insert(ec,nm,s,pos)

{ var i=1;

let newnode=null,ptr=null;

newnode=new DENode(ec,nm,s);

if(pos==1)

{

newnode.next=this.start;

this.start.prev.next=newnode;

newnode.prev=this.start.prev;

this.start.prev=newnode;

this.start=newnode;

this.size++;

document.write(":Inserted Node at "+pos+"th position<br />");

}else

{ ptr=this.start;i=1;

while((i<pos-1)&&(ptr.next!=this.start))

{ ptr=ptr.next;i++;

}

if(i==pos-1)

{ newnode.next=ptr.next;newnode.prev=ptr;

ptr.next.prev=newnode;

ptr.next=newnode;this.size++;

document.write(":Inserted Node at "+pos+"th position<br />");

}else

{ alert("wrong position"+pos);

document.write("invalid position<br />");

}

}

}

del(pos)

{

var i=1;

let delnode=null,ptr=null;

if(this.start==null)

{

alert("Linklist Underflow");

document.write("linkedlist underflow<br />");

}else

{ if(pos==1)

{

delnode=this.start;

this.start=this.start.next;this.start.prev=delnode.prev;

delnode.prev.next=this.start;

document.write("Deleted item="+delnode.code+":"+delnode.name+":"+delnode.sal);

document.write(":Deleted "+pos+" th item <br />");

this.size--;delnode=null;

}else

{ ptr=this.start;i=1;

while((i<pos-1)&&(ptr.next!=this.start))

{ i++;ptr=ptr.next;

}

if(i==pos-1)

{ delnode=ptr.next;

ptr.next=delnode.next;delnode.next.prev=ptr;

document.write("Deleted item="+delnode.code+":"+delnode.name+":"+delnode.sal);

document.write(":Deleted "+pos+" th item <br />");

this.size--;delnode=null;

}else

{ document.write("Invalid Position<br />");

}

}

}

}

search(key)

{

var ptr =this.start;

var i= 1;

document.write("<br>");

do

{

if(ptr.name ==key)

{

document.write(key + " is found at "+ i+"location ");

return;

}

ptr = ptr.next;

i=i+1;

}while(ptr != this.start);

document.write(key +" not found");

}

print()

{

let ptr=this.start;

document.write("Linked List:::");

do

{ document.write(ptr.code+":"+ptr.name+":"+ptr.sal+"::");

ptr=ptr.next;

}while(ptr!=this.start);document.write("<br/>");

}

}

function dcl()

{

var i=1,ec=0,nm=" ",s=0.0;

const l1=new DC\_LinkedList();//creating Dll

var ch=0,pos=0,n=0;

do

{

ch=parseInt(prompt("1:create 2:insert 3:delete 4:Print 5:search 7:exit ...Enter your choice"));

switch(ch)

{ case 1: n=parseInt(prompt("Enter total nodes to create"));

l1.create(n);break;

case 2: ec=parseInt(prompt("Enter Emp code "));

nm=prompt("Enter Name ");

s=parseFloat(prompt("Enter salary "));

pos=parseInt(prompt("Enter position to insert"));

l1.insert(ec,nm,s,pos);break;

case 3: pos=parseInt(prompt("Enter position to delete"));

l1.del(pos);break;

case 4:l1.print();break;

case 5:var key=prompt("Enter name to search");

l1.search(key); break;

case 7:break;

default:alert("Wrong choice");

}

document.write(" ---><br />");

}while(ch!=7);

}

#\_\_\_ OUTPUT :-

--->

Linked List:::1:Abhay:999999999::2:Virat:998989::3:Jonathan:1000000::4:Shreeman:5400000::5:Scout:650000::

--->

:Inserted Node at 5th position

--->

Linked List:::1:Abhay:999999999::2:Virat:998989::3:Jonathan:1000000::4:Shreeman:5400000::6:Danny:687421::5:Scout:650000::

--->

Deleted item=6:Danny:687421:Deleted 5 th item

--->

Linked List:::1:Abhay:999999999::2:Virat:998989::3:Jonathan:1000000::4:Shreeman:5400000::5:Scout:650000::

--->

Shreeman is found at 4location --->

--->

**Q3. Write a program to check the parenthesis of expression is well formedness or not Write a program to reverse the given string**.

HTML Code:

<html>

<head>

<script type="text/javascript" src="expr.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="Expr()" value="WelFormedness Expression" />

</body>

</html>

JS Code:

/\*Write a program which check whether the given expression is welformedness expression or not?\*/

class Stack

{ constructor(s)

{ this.size=s;

this.stk=new Array(this.size);

this.top=-1;//i m creating empty stact

}

isStackFull()

{ if(this.top==this.size-1)

return(true);

else

return(false);

}

isStackEmpty()

{ if(this.top==-1)

return(true);

else

return(false);

}

push(item)

{

if(this.top==this.size-1)

{ alert("Stack overflow");

document.write("Stack Overflow <br/>");

}else

{ this.top++;

this.stk[this.top]=item;

}

}

pop()

{ var item;

if(this.top==-1)

{ alert("Stack underflow");

document.write("Stack underflow <br/>");

}else

{ item=this.stk[this.top];

this.top=this.top-1;

}

return(item);

}

}

function Expr()

{ var i,flag=0;

var str;

str=prompt("Enter Expression : ");

alert("Given Expression :"+str);

document.write("Given String="+str+"<br/>");

const l1=new Stack(str.length);

for(i=0;i<str.length;i++)

{ if(str.charAt(i)=="(")

{ l1.push(str.charAt(i));

//document.write(str.charAt(i)+" pushed<br />");

}else if(str.charAt(i)==")")

{ if(l1.isStackEmpty()==false)

l1.pop();

else

{ document.write("Expression is not wellFormedness...Extra closing<br/>");

alert("Expression is not wellFormedness...Extra closing");

flag=1;break;

}

}

}

if(flag==0)

{

if((l1.isStackEmpty()==true)&&(i==str.length))

{ document.write("Expression is wellFormedness<br/>");

alert("Expression is wellFormedness");

}else

{

document.write("Expression is not wellFormedness...Extra Openings<br/>");

alert("Expression is not wellFormedness...Extra Openings");

}

}

}

Output:

Given String=((X^2)+(Y^2))

Expression is wellFormedness

Output:

Given String=((((a+b)\*c)/(a+b)

Expression is not wellFormedness...Extra Openings

Output:

Given String=(a+b)\*c)

Expression is not wellFormedness...Extra closing

**Q4. Write a program to create stack using linked list and manipulate it using menu driven code**

HTML Code:

<html>

<head>

<script type="text/javascript" src="Stack.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="Stack()" value="Stack" />

</body>

</html>

JS Code:

class SLNode

{ constructor(data,next=null)

{

this.data=data;

this.next=next;

}

}

class Stack\_list

{ constructor()

{ this.top=null;//i m creating empty stact

}

push(item)

{ let newnode=null;

newnode=new SLNode(item);

if(newnode==null)

{ alert("Stack overflow");

document.write("Stack Overflow <br/>");

}else

{

if(this.top==null)

{

this.top=newnode;

}else

{ newnode.next=this.top;

this.top=newnode;

}

document.write(item+"is inserted in the stack");

}

}

pop()

{ let delnode=null;var item;

if(this.top==null)

{ alert("Stack underflow");

document.write("Stack underflow <br/>");

}else

{ item=this.top.data;delnode=this.top;

this.top=this.top.next;

delnode=null;

alert("Deleted item: "+item);

document.write("deleted item: "+item+"<br/>");

}

}

printStack()

{ var stk="";

let ptr=this.top;

document.write("Stack ::");

while(ptr)

{ stk=stk+ptr.data+":";

document.write(ptr.data+":");

ptr=ptr.next;

}document.write(":<br />");

alert("Stack ::"+stk);

}

}

function Stack()

{

const l1=new Stack\_list();//creating stack

var ch=0,item=0;

do

{

ch=parseInt(prompt("1:Push 2:Pop 3:Print stack 4:exit ...Enter your choice"));

switch(ch)

{ case 1: item=parseInt(prompt("Enter item to push"));

l1.push(item);break;

case 2: l1.pop();break;

case 3: l1.printStack();break;

case 4:break;

default:alert("Wrong choice");

}

document.write(" ---><br/>");

}while(ch!=4);

}

Output:

Stack :::

--->

11is inserted in the stack --->

22is inserted in the stack --->

33is inserted in the stack --->

Stack ::33:22:11::

--->

deleted item 33

--->

Stack ::22:11::

--->

deleted item 22

--->

Stack ::11::

--->

66is inserted in the stack --->

Stack ::66:11::

--->

deleted item 66

--->

Stack ::11::

--->

--->

**Q5. Write a program to create CQ using array and manipulate it using menu driven code**

HTML Code:

<html>

<head>

<script src="Static\_CQ.js"> </script>

</head>

<body>

click here to continue <input type="button" onclick="Q()" value="CQ" />

</body>

</html>

JS Code:

class CQ

{ constructor(s)

{ this.size=s;

this.Q=new Array(this.size);

this.rear=-1;this.front=-1//i m creating empty Q

}

insertQ(item)

{

if(((this.front==0)&&(this.rear==this.size-1))||(this.rear==this.front-1))

{ alert("Q overflow");

document.write("Q Overflow <br/>");

}else

{ if((this.rear==-1)&&(this.front==-1)){this.rear++;this.front++;}

else {this.rear=(this.rear+1)%this.size;}

this.Q[this.rear]=item;

document.write(item+" is inserted in the Q");

}

}

delQ()

{ var item;

if((this.rear==-1)&&(this.front==-1))

{ alert("Q underflow");

document.write("Q underflow <br/>");

}else

{ item=this.Q[this.front];

if(this.front==this.rear)

{ this.front=-1;this.rear=-1;

}else

this.front=(this.front+1)%this.size;

alert("Deleted item : "+item);

document.write("deleted item : "+item+"<br/>");

}

}

printQ()

{

var qt="";

let ptr=this.front;

document.write("Q ::");

if(this.front<=this.rear)

{

while(ptr<=this.rear)

{ qt=qt+this.Q[ptr]+":";

document.write(this.Q[ptr]+":");

ptr=ptr+1;

}document.write(":<br />");

}else

{ for(ptr=this.front;ptr<this.size;ptr++)

{ qt=qt+this.Q[ptr]+":";

document.write(this.Q[ptr]+":");

}

for(ptr=0;ptr<=this.rear;ptr++)

{ qt=qt+this.Q[ptr]+":";

document.write(this.Q[ptr]+":");

}

}alert("Q :: "+qt);

}

}

function Q()

{

const l1=new CQ(5);//creating Q

var ch=0,item=0;

do

{

ch=parseInt(prompt("1:Insert 2:Delete 3:Print Q 4:exit ...Enter your choice"));

switch(ch)

{ case 1: item=parseInt(prompt("Enter item to insert"));

l1.insertQ(item);break;

case 2: l1.delQ();break;

case 3: l1.printQ();break;

case 4:break;

default:alert("Wrong choice");

}

document.write(" ---><br/>");

}while(ch!=4);

}

Output:

11is inserted in the Q --->

22is inserted in the Q --->

33is inserted in the Q --->

44is inserted in the Q --->

Q ::11:22:33:44::

--->

55is inserted in the Q --->

Q Overflow

--->

deleted item 11

--->

deleted item 22

--->

Q ::33:44:55::

--->

77is inserted in the Q --->

Q ::33:44:55:77: --->

98is inserted in the Q --->

Q ::33:44:55:77:98: --->

Q Overflow

--->

deleted item 33

--->

Q ::44:55:77:98: --->

deleted item 44

--->

deleted item 55

--->

deleted item 77

--->

deleted item 98

--->

Q underflow

--->

--->

**Q6. Write a program to reverse stack using queue**

HTML Code:

<html>

<head>

<script type="text/javascript" src="ReverseQ.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="ReverseQ()" value="Reverse Queue" />

</body>

</html>

JS Code:

class SLNode

{ constructor(data,next=null)

{

this.data=data;

this.next=next;

}

}

class DQ

{ constructor()

{ this.front=null;this.rear=null//i m creating empty Q

}

insertQ(item)

{ let newnode=null;

newnode=new SLNode(item);

if(newnode==null)

{ alert("Q overflow");

document.write("Q Overflow <br/>");

}else

{

if((this.rear==null)&&(this.front==null))

{

this.rear=newnode;this.front=newnode;

}else

{ this.rear.next=newnode;

this.rear=newnode;

}

document.write(item+" is inserted in the Q<br/>");

}

}

delQ()

{

let delnode=null;var item;

if((this.front==null)&&(this.rear==null))

{ alert("Q underflow");

document.write("Q underflow <br/>");

}else

{

item=this.front.data;delnode=this.front;

if(this.front==this.rear)

{ this.front=null;this.rear=null;

}else this.front=this.front.next;

delnode=null;

alert("Deleted item : "+item);

document.write("deleted item : "+item+"<br/>");

}

return(item);

}

printQ()

{

var qt="";

let ptr=this.front;

document.write("Q ::");

while(ptr)

{ qt=qt+ptr.data+":";

document.write(ptr.data+":");

ptr=ptr.next;

}

document.write(":<br />");alert("Q::"+qt);

}

}

function Q()

{

const l1=new DQ();//creating dynamic Q

var ch = 0, item = 0;

do

{

ch=parseInt(prompt("1:Insert 2:Delete 3:Print Q 4:exit ...Enter your choice"));

switch(ch)

{ case 1: item=parseInt(prompt("Enter item to push"));

l1.insertQ(item);break;

case 2: l1.delQ();break;

case 3: l1.printQ();break;

case 4:break;

default:alert("Wrong choice");

}

document.write(" ---><br/>");

}while(ch!=4);

}

class Stack\_list

{ constructor()

{ this.top=null;//i m creating empty stact

}

push(item)

{ let newnode=null;

newnode=new SLNode(item);

if(newnode==null)

{ alert("Stack overflow");

document.write("Stack Overflow <br/>");

}else

{

if(this.top==null)

{

this.top=newnode;

}else

{ newnode.next=this.top;

this.top=newnode;

}

document.write(item+"is inserted in the stack<br/>");

}

}

pop()

{ let delnode=null;var item;

if(this.top==null)

{ alert("Stack underflow");

document.write("Stack underflow <br/>");

}else

{ item=this.top.data;delnode=this.top;

this.top=this.top.next;

delnode=null;

alert("Deleted item: "+item);

document.write("deleted item: "+item+"<br/>");

}

return(item);

}

printStack()

{ var stk="";

let ptr=this.top;

document.write("Stack ::");

while(ptr)

{ stk=stk+ptr.data+":";

document.write(ptr.data+":");

ptr=ptr.next;

}document.write(":<br />");

alert("Stack ::"+stk);

}

}

function Stack()

{

const l1=new Stack\_list();//creating stack

var ch=0,item=0;

do

{

ch=parseInt(prompt("1:Push 2:Pop 3:Print stack 4:exit ...Enter your choice"));

switch(ch)

{ case 1: item=parseInt(prompt("Enter item to push"));

l1.push(item);break;

case 2: l1.pop();break;

case 3: l1.printStack();break;

case 4:break;

default:alert("Wrong choice");

}

document.write(" ---><br/>");

}while(ch!=4);

}

function ReverseQ()

{ const q=new DQ();

const stk=new Stack\_list();var n=0,i=0,item=0;

n=parseInt(prompt("Enter Total elements of the Queue"));

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

{ item=parseInt(prompt("Enter item to insert in Q"));

q.insertQ(item);

}

q.printQ();

while(!((q.rear==null)&&(q.front==null)))

{ item=q.delQ();

stk.push(item);

}

while(stk.top!=null)

{ item=stk.pop();

q.insertQ(item);

}

q.printQ();

}

OutPut:

12 is inserted in the Q  
13 is inserted in the Q  
14 is inserted in the Q  
15 is inserted in the Q  
16 is inserted in the Q  
17 is inserted in the Q  
18 is inserted in the Q  
19 is inserted in the Q  
20 is inserted in the Q  
21 is inserted in the Q  
22 is inserted in the Q  
Q ::12:13:14:15:16:17:18:19:20:21:22::  
deleted item : 12  
12is inserted in the stack  
deleted item : 13  
13is inserted in the stack  
deleted item : 14  
14is inserted in the stack  
deleted item : 15  
15is inserted in the stack  
deleted item : 16  
16is inserted in the stack  
deleted item : 17  
17is inserted in the stack  
deleted item : 18  
18is inserted in the stack  
deleted item : 19  
19is inserted in the stack  
deleted item : 20  
20is inserted in the stack  
deleted item : 21  
21is inserted in the stack  
deleted item : 22  
22is inserted in the stack  
deleted item: 22  
22 is inserted in the Q  
deleted item: 21  
21 is inserted in the Q  
deleted item: 20  
20 is inserted in the Q  
deleted item: 19  
19 is inserted in the Q  
deleted item: 18  
18 is inserted in the Q  
deleted item: 17  
17 is inserted in the Q  
deleted item: 16  
16 is inserted in the Q  
deleted item: 15  
15 is inserted in the Q  
deleted item: 14  
14 is inserted in the Q  
deleted item: 13  
13 is inserted in the Q  
deleted item: 12  
12 is inserted in the Q  
Q ::22:21:20:19:18:17:16:15:14:13:12::

**Q7. Write a program to create Binary search tree and traverse it using recursive preorder, inorder, postorder methods.**

HTML Code:

<html>

<head>

<script type="text/javascript" src="BST.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="Tree()" value="BST" />

</body>

</html>

JS Code:

class treeNode

{ constructor(data,left=null,right=null)

{ this.left=left

this.data=data;

this.right=right;

}

}

class BST

{ constructor()

{ this.root=null;//i m creating empty Tree

}

create(n)

{ var i=1;let newnode=null,temp=null;var item=0;

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

{ item=parseInt(prompt("Enter item:"));

newnode=new treeNode(item);

if(this.root==null)

{

this.root=newnode;

}else

{ temp=this.root;

do

{ if(newnode.data<=temp.data)

{ if(temp.left==null)

{ temp.left=newnode;break;

}else

temp=temp.left;

}else

{ if(temp.right==null)

{ temp.right=newnode;break;

}else

temp=temp.right;

}

}while(temp!=null);

}

}

}

preorder(temp)

{ if(this.root==temp)document.write("<br/>Preorder::");

if(temp!=null)

{ document.write(temp.data+":");

this.preorder(temp.left);

this.preorder(temp.right);

}

}

inorder(temp)

{ if(this.root==temp)document.write("<br/>Inorder::");

if(temp!=null)

{ this.inorder(temp.left);

document.write(temp.data+":");

this.inorder(temp.right);

}

}

postorder(temp)

{ if(this.root==temp)document.write("<br/>Postorder::");

if(temp!=null)

{ this.postorder(temp.left);

this.postorder(temp.right);

document.write(temp.data+":");

}

}

}

function Tree()

{

const t1=new BST();

t1.create(7);

t1.preorder(t1.root);

t1.inorder(t1.root);

t1.postorder(t1.root);

}

Output:

Preorder::34:23:67:56:89:

Inorder::23:34:56:67:89:

Postorder::23:56:89:67:34:

**Q8. Write a program to implement graph using adjacency matrix and traverse it using BFT and DFT**

HTML Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>graph</title>

<script src="./graph\_bft\_dft.js"></script>

</head>

<body>

</body>

</html>

JS Code:

class Graph {

constructor(numVertices) {

this.numVertices = numVertices;

this.adjMatrix = new Array(numVertices);

for (let i = 0; i < numVertices; i++) {

this.adjMatrix[i] = new Array(numVertices).fill(0);

}

}

addEdge(v1, v2) {

this.adjMatrix[v1][v2] = 1;

this.adjMatrix[v2][v1] = 1;

}

removeEdge(v1, v2) {

this.adjMatrix[v1][v2] = 0;

this.adjMatrix[v2][v1] = 0;

}

bfs(start) {

const visited = new Array(this.numVertices).fill(false);

const queue = [start];

visited[start] = true;

while (queue.length > 0) {

const v = queue.shift();

document.writeln(v);

for (let i = 0; i < this.numVertices; i++) {

if (this.adjMatrix[v][i] === 1 && !visited[i]) {

visited[i] = true;

queue.push(i);

}

}

}

}

dfs(start, visited = []) {

visited[start] = true;

document.writeln(start);

for (let i = 0; i < this.numVertices; i++) {

if (this.adjMatrix[start][i] === 1 && !visited[i]) {

this.dfs(i, visited);

}

}

}

}

// Example usage

const graph = new Graph(5);

graph.addEdge(0, 1);

graph.addEdge(0, 2);

graph.addEdge(1, 2);

graph.addEdge(2, 3);

graph.addEdge(3, 4);

document.write("BFS:");

graph.bfs(0);

document.write("DFS:");

graph.dfs(0);

OutPut:

BFS:0 1 2 3 4 DFS:0 1 2 3 4

**Q9. Write a program to implement linear probing hashing for insert and search operation**

HTML Code:

<html>

<head>

<script type="text/javascript" src="Hash\_LB.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="Hashing()" value="Hash Table with Linear Probing" />

</body>

</html>

JS Code:

class HashTable

{ constructor(s)

{ this.hashTable=new Array(s);//i m creating HashTable

}

init\_Hash()

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

this.hashTable[i]=" ";

}

print\_Hash()

{ var altr="";

document.write("<Br/>Hash Table<Br/>");

for(var i=0;i<this.hashTable.length;i++)

{

if(this.hashTable[i]!=" ")

{

document.write("<Br/>"+i+":"+this.hashTable[i]);

altr=altr+this.hashTable[i]+":";

}

}alert(altr);

}

hashFunction(item)

{ return(item.charCodeAt(0));

}

insert(item)

{ var i=this.hashFunction(item);var j=i;

if(this.hashTable[i]==" ")

{

this.hashTable[i]=item;

document.write("<Br/> "+item+"is succesfully inserted in Hash table");

alert("This "+item+"is succesfully inserted in Hash table");

}else

{ while(this.hashTable[i]!=" ")

{ i=(i+1)%this.hashTable.length;if(i==j)break;

}

if(this.hashTable[i]==" ")

{

this.hashTable[i]=item;

document.write("<Br/> "+item+"is succesfully inserted in Hash table");

alert("This "+item+"is succesfully inserted in Hash table");

}else if(i==j)

{

document.write("<Br/> Hash Overflow");

alert("Hash Overflow");

}

}

}

search(item)

{ var i=this.hashFunction(item);var j=i;

do

{

if(this.hashTable[i]==item)

{ document.write("<Br/>"+item+" is found at "+i+" th location in Hash Table");

alert(item+" is found at "+i+" th location in Hash Table");return;

}i=(i+1)%this.hashTable.length;if(this.hashTable[i]==" ")break;

}while(i!=j)

if((this.hashTable[i]==" ")||(i==j))

{ document.write("<Br/>"+item+" is not found in Hash Table");

alert(item+" is not found in Hash Table");

}

}

}

function Hashing()

{ const h=new HashTable(128);h.init\_Hash();

var ch=0,item;

do

{ ch=parseInt(prompt("1:Insert 2:Print 3:Search 4:exit ...Enter your choice"));

switch(ch)

{ case 1: item=prompt("Enter item to insert");

h.insert(item);break;

case 2: h.print\_Hash();break;

case 3: item=prompt("Enter item to search");

h.search(item);break;

case 4:break;

default:alert("Wrong choice");

}

}while(ch!=4);

}

Output:

111is succesfully inserted in Hash table

222is succesfully inserted in Hash table

333is succesfully inserted in Hash table

aaais succesfully inserted in Hash table

AAAis succesfully inserted in Hash table

ABCis succesfully inserted in Hash table

BBBBis succesfully inserted in Hash table

112is succesfully inserted in Hash table

Hash Table

49:111

50:222

51:333

52:112

65:AAA

66:ABC

67:BBBB

97:aaa

AA2is succesfully inserted in Hash table

Hash Table

49:111

50:222

51:333

52:112

65:AAA

66:ABC

67:BBBB

68:AA2

97:aaa

**Q10. Write a program which implement the rain terraces to measure trapping rain water**

HTML Code:

<html>

<head>

<script type="text/javascript" src="TW.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="TW()" value="Trapped Water on Rain terraces" />

</body>

</html>

JS Code:

function maxWater(e, n) {

var stw = 0, i = 0, left = 0, right = 0, j = 0, min;

for (i = 1; i < n - 1; i++) {

left = e[i];

for (j = 0; j < i; j++) {

if (left < e[j]) left = e[j];

}

right = e[i];

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

if (right < e[j]) right = e[j];

}

if (left <= right) min = left; else min = right;

stw = stw + (min - e[i]);

}

return (stw);

}

function TW() {

alert("Hi");

var el\_map, n, i;

n = parseInt(prompt("Enter Array size :"));

el\_map = new Array(n);

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

el\_map[i] = parseInt(prompt("Enter Array element :"));

document.write("<Br/>Elivation map length=" + n + "<Br/>Elivation Map =");

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

document.write(el\_map[i] + ":");

var tw = maxWater(el\_map, n);

document.write("<Br/> Total Trapped water in above elivation map=" + tw);

}

Output:

Elivation map length=9

Elivation Map =3:1:2:5:2:3:6:0:7:

Total Trapped water in above elivation map=14

**Q11. Write a program to implements to implement Kruskals Algorithm MST**

HTML Code:

<html>

<head>

<script type="text/javascript" src="K\_MST.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="graph()" value="Kruskal's Algorithm for minimum spanning tree" />

</body>

</html>

JS Code:

class Edge

{ constructor(s=0,d=0,w=0)

{ //alert("in edge");

this.src=s;

this.dest=d;

this.weight=w;

}

}

class Graph

{ constructor(v,e)

{ //alert("in graph");

this.V=v;

this.E=e;

//alert("in graph1");

this.Edges=new Array(this.E);

//alert("in graph2");

for(var i=0;i<this.E;i++)

{ this.Edges[i]=new Edge();

}//alert("in graph3");

}

}

class subset

{ constructor()

{ this.parent;

this.rank;

}

}

function find(subsets,i)

{ if(subsets[i].parent!=i)

subsets[i].parent=find(subsets,subsets[i].parent);

return subsets[i].parent;

}

function Union(subsets,x,y)

{ var xroot=find(subsets,x);

var yroot=find(subsets,y);

if(subsets[xroot].rank<subsets[yroot].rank)

subsets[xroot].parent=yroot;

else if(subsets[xroot].rank>subsets[yroot].rank)

subsets[yroot].parent=xroot;

else

{ subsets[yroot].parent=xroot;

subsets[xroot].rank++;

}

}

function sort(g)

{ var i,j,temp;

for(i=0;i<g.E;i++)

{ for(j=i+1;j<g.E;j++)

{ if(g.Edges[i].weight>g.Edges[j].weight)

{

temp=g.Edges[i].src;g.Edges[i].src=g.Edges[j].src;g.Edges[j].src=temp;

temp=g.Edges[i].dest;g.Edges[i].dest=g.Edges[j].dest;g.Edges[j].dest=temp;

temp=g.Edges[i].weight;g.Edges[i].weight=g.Edges[j].weight;g.Edges[j].weight=temp;

}

}

}

}

function KruskalMST(g)

{ alert("In Kruskal");

var V=g.V;

var i,e;

var result=new Array(V);alert("In Kruskal1"+V);

for(i=0;i<V;i++){ result[i]=new Edge(); }

alert("In Kruskal1");

sort(g);

var subsets=new Array(V);

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

{ subsets[i]=new subset();

subsets[i].parent=i;

subsets[i].rank=0;

}e=0;i=0;

alert("In Kruskal2");

while(e<V-1 && i<g.E)

{ var next\_edge=new Edge();

next\_edge=g.Edges[i++];

var x=find(subsets,next\_edge.src);

var y=find(subsets,next\_edge.dest);

if(x!=y)

{ result[e++]=next\_edge;

Union(subsets,x,y);

}

}

var minCost=0;

document.write("<Br/>Following are the edges in the constructed MST<Br/>");

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

{ document.write(result[i].src+"--"+result[i].dest+"=="+result[i].weight+"<Br/>");

minCost=minCost+result[i].weight;

}

document.write("Minimum Cost Spanning Tree :"+minCost+"<Br/>");

}

function graph()

{ alert("Hi");var vt,ed;

vt=parseInt(prompt("Enter Total Vertex"));

ed=parseInt(prompt("Enter Total Edges"));

var i, g=new Graph(vt,ed);

for(i=0;i<g.E;i++)

{ g.Edges[i].src=parseInt(prompt("Enter Source"));

g.Edges[i].dest=parseInt(prompt("Enter Destinition"));

g.Edges[i].weight=parseInt(prompt("Enter Weight"));

}

document.write("<Br/>Graph("+g.V+","+g.E+")");

for(i=0;i<g.E;i++)

{ document.write("<Br/>"+g.Edges[i].src+" to "+g.Edges[i].dest+"weight="+g.Edges[i].weight);

}

KruskalMST(g);

/\*sort(g);

document.write("<Br/>Graph("+g.V+","+g.E+")");

for(i=0;i<g.E;i++)

{ document.write("<Br/>"+g.Edges[i].src+" to "+g.Edges[i].dest+"weight="+g.Edges[i].weight);

}\*/

}

Output:

Graph(4,5)

0 to 1weight=10

0 to 2weight=6

0 to 3weight=5

1 to 3weight=15

2 to 3weight=6

Following are the edges in the constructed MST

0--3==5

0--2==6

0--1==10

Minimum Cost Spanning Tree :21

Graph(9,14)

0 to 1weight=4

1 to 2weight=8

2 to 3weight=7

3 to 4weight=9

4 to 5weight=10

5 to 6weight=2

6 to 7weight=1

7 to 0weight=8

1 to 7weight=11

7 to 8weight=7

8 to 2weight=2

8 to 6weight=6

2 to 5weight=4

3 to 5weight=14

Following are the edges in the constructed MST

6--7==1

5--6==2

8--2==2

0--1==4

2--5==4

2--3==7

1--2==8

3--4==9

Minimum Cost Spanning Tree :37

**Q12. Write a program to implements to implement Prims Algorithm MST**

HTML Code:

<html>

<head>

<script type="text/javascript" src="P\_MST.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="graph()" value="Prim's Minimum Spanning Tree Algorithm" />

</body>

</html>

JS Code:

function minKey(key,mstSet,V)

{ var min=66000,min\_index,v;

for(v=0;v<V;v++)

{ if(mstSet[v]==false && key[v]<=min)

{ min=key[v];min\_index=v;

}

}

return min\_index;

}

function primMST(graph,src,v)

{ alert(" in primMST");

var key=new Array(v);

var parent=new Array(v);

var mstSet=new Array(v),i;alert(" in primMST1");

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

{

key[i]=66000;mstSet[i]=false;

}

key[0]=src;

parent[src]=-1;

var count=0,n=0,u=0;alert(" in primMST2");

for(count=0;count<v-1;count++)

{

u=minKey(key,mstSet,v);

mstSet[u]=true;

for(n=0;n<v;n++)

{

if( !mstSet[n] && graph[u][n] && graph[u][n]<key[n])

{ parent[n]=u;key[n]=graph[u][n];

}

}

}

document.write("<Br/>Vertex:::Distance From Source Node 0<BR/>");

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

{ document.write(parent[i]+"--"+i+":::"+graph[i][parent[i]]+"<Br/>");

}

}

function graph()

{ alert("Hi");

var v,g,i,j;

v=parseInt(prompt("Enter total number of vertices :"));

g=new Array(v);

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

g[i]=new Array(v);

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

{ for(j=0;j<v;j++)

{ g[i][j]=-1;

}

}

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

{ for(j=0;j<v;j++)

{

if(i==j)

{

g[i][j]=0;

}else

{ if(g[i][j]==-1)

{

g[i][j]=parseInt(prompt("Enter distance of node "+i+" to node "+j+" :"));

g[j][i]=g[i][j];

}

}

}

}

document.write("<Br/>Total Vertyex="+v+"<Br/>Graph =<Br/>");

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

{ for(j=0;j<v;j++)

{

document.write(" "+g[i][j]);

}

document.write("<Br/>");

}

primMST(g,0,v);

}

Output:

Total Vertyex=9

Graph =

0 4 0 0 0 0 0 8 0

4 0 8 0 0 0 0 11 0

0 8 0 7 0 4 0 0 2

0 0 7 0 9 14 0 0 0

0 0 0 9 0 10 0 0 0

0 0 4 14 10 0 2 0 0

0 0 0 0 0 2 0 1 6

8 11 0 0 0 0 1 0 7

0 0 2 0 0 0 6 7 0

Vertex:::Distance From Source Node 0

0--1:::4

5--2:::4

2--3:::7

3--4:::9

6--5:::2

7--6:::1

0--7:::8

2--8:::2

Total Vertyex=5

Graph =

0 2 0 6 0

2 0 3 8 5

0 3 0 0 7

6 8 0 0 9

0 5 7 9 0

Vertex:::Distance From Source Node 0

0--1:::2

1--2:::3

0--3:::6

1--4:::5

sss

**Q13. Write a program to implement binary search algorithm using divide and conquer technique**

HTML Code:

<html>

<head>

<script type="text/javascript" src="BS.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="BS()" value="Binary Search" />

</body>

</html>

JS Code:

function binarySearch(A,LB,UB,it)

{ var midPoint;

alert("LB="+LB+" UB="+UB);

if(UB>=LB)

{ midPoint=parseInt(LB+(UB-LB)/2);

if(A[midPoint]==it)

return(midPoint);

else if(A[midPoint]<it)

{ return(binarySearch(A,midPoint+1,UB,it));

}else

{ return(binarySearch(A,LB,midPoint-1,it));

}

}

return(-1);

}

function BS()

{ alert("hi");

var i,item,loc=-1,n=parseInt(prompt("Enter total elements"));

var num=new Array(n);

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

{ num[i]=parseInt(prompt("Enter element"));

}

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

{ document.write("<Br/>"+num[i]);

}

item=parseInt(prompt("Enter element which you want to search"));

document.write("<Br/>item="+item);

loc=binarySearch(num,0,n-1,item);

if(loc!=-1)

{ document.write("<Br/>"+item+" is found at location in num "+loc);

}else

{ document.write("<Br/>"+item+" is not found in num");

}

}

OUTPUT:

11

22

33

44

55

66

77

88

99

122

item=88

88 is found at location in num 7

**Q14. Write a program to implement TOWER OF HANNOI problem**

HTML Code:

<html>

<head>

<script type="text/javascript" src="TOH.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="mainTower()" value="Tower of Hannoi" />

</body>

</html>

JS Code:

function TOH(disk,source,dest,aux)

{ alert("in TOH disk="+disk);

if(disk==1)

{ document.write("<Br/> Move Disk from Tower "+source+" to Tower "+dest);

}else

{ TOH(disk-1,source,aux,dest);

document.write("<Br/> Move Disk from Tower "+source+" to Tower "+dest);

TOH(disk-1,aux,dest,source);

}

}

function mainTower()

{ alert("hi");

var n;

n=parseInt(prompt("Enter total number of the disk"));

document.write("<Br/>Total disk="+n);

TOH(n,"A","C","B");

}

Output:

Total disk=4

Move Disk from Tower A to Tower B

Move Disk from Tower A to Tower C

Move Disk from Tower B to Tower C

Move Disk from Tower A to Tower B

Move Disk from Tower C to Tower A

Move Disk from Tower C to Tower B

Move Disk from Tower A to Tower B

Move Disk from Tower A to Tower C

Move Disk from Tower B to Tower C

Move Disk from Tower B to Tower A

Move Disk from Tower C to Tower A

Move Disk from Tower B to Tower C

Move Disk from Tower A to Tower B

Move Disk from Tower A to Tower C

Move Disk from Tower B to Tower C

**Q15. Write a program to find longest common sub sequence**

HTML Code:

<html>

<head>

<script type="text/javascript" src="LCSubSq.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="LCS()" value="Longest Common Sub Sequence" />

</body>

</html>

JS Code:

function Print\_LCS(B,X,i,j)

{

if((i==0)&&(j==0))

{ //document.write(X.charAt(i));

return;

}

if(B[i][j]=="D")

{ Print\_LCS(B,X,i-1,j-1);

document.write(X.charAt(i-1));//alert(x.charAt(i));

}else if(B[i][j]=="U")

Print\_LCS(B,X,i-1,j);

else

Print\_LCS(B,X,i,j-1);

}

function longestCommonSubSequence(x,m,y,n)

{ var c=new Array(m+1);

var b=new Array(m+1);

var i,j,L=0;

for(i=0;i<m+1;i++)

{ c[i]=new Array(n+1);

b[i]=new Array(n+1);

}

for(i=0;i<m+1;i++){c[i][0]=0;b[i][0]="x";}

for(j=0;j<n+1;j++){c[0][j]=0;b[0][j]="x";}

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

{ for(j=1;j<=n;j++)

{ b[i][j]="x";

if(x.charAt(i-1)==y.charAt(j-1))

{ c[i][j]=c[i-1][j-1]+1;

if(L<c[i][j])L=c[i][j];

b[i][j]="D";

}else if(c[i-1][j]>=c[i][j-1])

{

c[i][j]=c[i-1][j];

b[i][j]="U";

}else

{ c[i][j]=c[i][j-1];

b[i][j]="L";

}

}

}

document.write("<BR/>Matrix C<BR/>");

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

{ //alert("this is "+i);

for(j=0;j<=n;j++)

{ //alert("this is "+j+"::"+c[i][j]+":"+b[i][j]);

document.write(c[i][j]+":"+b[i][j]+" ");

}document.write("<Br/>");

}

document.write("<Br/>Longest Common subsequence = ");

Print\_LCS(b,x,m,n);

return(c[m][n]);

}

function LCS()

{ alert("Hi");

var x,y,m,n;

x=prompt("Enter First sub sequence");

y=prompt("Enter second sub sequence");

m=x.length;

n=y.length;

document.write("<Br/>Sequence 1 = "+x+" length = "+m+"<Br/>Sequence 2 = "+y+" length="+n);

len=longestCommonSubSequence(x,m,y,n);

document.write("<Br/> Length of LCSubsequence="+len);

}

Output:

Sequence 1 = GXTXYAB length = 7

Sequence 2 = AGGTAB length=6

Matrix C

0:x 0:x 0:x 0:x 0:x 0:x 0:x

0:x 0:U 1:D 1:D 1:L 1:L 1:L

0:x 0:U 1:U 1:U 1:U 1:U 1:U

0:x 0:U 1:U 1:U 2:D 2:L 2:L

0:x 0:U 1:U 1:U 2:U 2:U 2:U

0:x 0:U 1:U 1:U 2:U 2:U 2:U

0:x 1:D 1:U 1:U 2:U 3:D 3:L

0:x 1:U 1:U 1:U 2:U 3:U 4:D

Longest Common subsequence = GTAB

Length of LCSubsequence=4

**Q16. Write a program to implement regular expression to check whether given pattern match with input string or not?**

HTML Code:

<html>

<head>

<script src="Stacknode.js"></script>

</head>

<body>

<input type="Button" onclick="pcheck()" value="Stack\_Expr\_check"/>

</body>

</html>

JS Code:

class Stacknode

{

constructor(d='10',n=null)

{

this.data=d;

this.next=n;

}

}

class Stack

{

constructor()

{

this.top=null;

}

push(ptr)

{

var newnode=new Stacknode(ptr);

if(newnode==null)

{

document.write("can't check stack overflow");

return;

}

if(this.top!=null)

{

this.top.next=newnode;

}

this.top=newnode;

}

pop()

{

var ptr=' ',delnode=null;

if(this.top==null)

{

document.write("Wrong Expression,stack underflow");

return;

}

delnode=top;

ptr=delnode.data;

this.top=this.top.next;

}

}

function pcheck()

{

var expr=prompt("Enter Expression");

var stk=new Stack();

var i=0;

for(i=0;i<expr.length;i=i+1)

{

if(expr[i]=='(')

{

stk.push(expr[i]);

}

else if(expr[i]==')')

{

stk.pop();

}

}

if(stk.top==null)

{

document.write("Expr is well formatted");

}

else

{

document.write("Expr is not well formatted");

}

}

OutPut:

Expr is well formatted

**Q17. Write a program to create doubly circular linked list for insert delete search print operation using menu driven program**

HTML Code:

<html>

<head>

<script type="text/javascript" src="NQueen.js" ></script>

</head>

<body>

Click here for the result

<input type="button" onclick="Queen()" value="N Queen Problem" />

</body>

</html>

JS Code:

function printBoard(b)

{ document.write("<Br/>Queens Placed at following position on the board<Br/>");

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

{ document.write("<Br/>::::");

for(var j=0;j<b.length;j++)

document.write(b[i][j]+"::::");

}

}

function printBoard1(b)

{ document.write("<Br/><Br/>Queens Placed at following position on the board<Br/>");

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

{ document.write("<Br/>");

for(var j=0;j<b.length;j++)

document.write(b[i][j]);

}

}

function isSafe(b,r,c,N)

{ var i,j;

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

if(b[r][i]==1)

return false;

for(i=r,j=c;j>=0 && i>=0; i--,j--)

if(b[i][j]==1)

return false;

for(i=r,j=c;j>=0 && i<N; i++,j--)

if(b[i][j]==1)

return false;

return true;

}

function placeNQueen(b,c,N)

{ if(c>=N)return true;

for(var i=0;i<N;i++)

{ if(isSafe(b,i,c,N))

{

b[i][c]=1;

if(placeNQueen(b,c+1,N))

return true;

b[i][c]=0;

}

}

return false;

}

function Queen()

{ //alert("Hi");

var n=parseInt(prompt("Enter total number of queens"));

var board=new Array(n);

var i,j;

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

{ board[i]=new Array(n);

}

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

{ for(j=0;j<n;j++)

{ board[i][j]=0;

}

}

if(placeNQueen(board,0,n)==false)

{ document.write("<Br/> Solution does not exist");

return;

}

printBoard(board);

printBoard1(board);

}

Output:

Queens Placed at following position on the board

::::0::::0::::0::::1::::0::::0::::

::::1::::0::::0::::0::::0::::0::::

::::0::::0::::0::::0::::1::::0::::

::::0::::1::::0::::0::::0::::0::::

::::0::::0::::0::::0::::0::::1::::

::::0::::0::::1::::0::::0::::0::::

Output 2:

Queens Placed at following position on the board

::::1::::0::::0::::0::::0::::0::::0::::0::::

::::0::::0::::0::::0::::0::::0::::1::::0::::

::::0::::0::::0::::0::::1::::0::::0::::0::::

::::0::::0::::0::::0::::0::::0::::0::::1::::

::::0::::1::::0::::0::::0::::0::::0::::0::::

::::0::::0::::0::::1::::0::::0::::0::::0::::

::::0::::0::::0::::0::::0::::1::::0::::0::::

::::0::::0::::1::::0::::0::::0::::0::::0::::

Queens Placed at following position on the board

10000000

00000010

00001000

00000001

01000000

00010000

00000100

00100000