***Discrete mathematics practical list Questions***

***Q1.Write a program to create a Set A and determine the cardinality of Set for an input array of elements (repetition allowed) and perform the following operations on the Set:***

1. ***Ismember(a,A):check whether an element belongs to set or not and return value as true/false.***
2. ***Powerset(A):list all the elements of power set of A.***

**Solution:**

**#include<iostream>**

**#include<cmath>**

**using namespace std;**

**void set(int n,int Set\_A[])**

**{**

**int count=0;**

**cout<<"enter the elements in the Set\_A"<<endl;**

**int i,j,k;**

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

**cin>>Set\_A[i];**

**}**

**cout<<"Set\_A ";**

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

**cout<< Set\_A[i]<<" ";**

**count++;**

**}**

**cout<<"\n cardinality of the entered set is"<<count;**

**}**

**bool ismember( int Set\_A[],int n,int x1){**

**int low=0;**

**int mid=0;**

**int high=n-1;**

**while(low<=high){**

**mid=(low+high)/2;**

**if(Set\_A[mid]==x1){**

**return true;**

**}**

**else if(Set\_A[mid]>x1){**

**high=mid-1;**

**}**

**else{**

**low=mid+1;**

**}**

**}**

**if(low>high){**

**return false;**

**}**

**}**

**void powerset(int Set\_A[],int n)**

**{**

**int N=pow(2,n);**

**int i,j;**

**cout<<"Number of elements in the powerset of Set\_A including phi element\n"<<N;**

**cout<<"{}";//null set as it is a subset of each and every set;**

**for(i=0;i<N;i++){**

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

**if(i&(1<<j))**

**cout<<Set\_A[j]<<",";**

**}**

**cout<<endl;**

**}**

**}**

**int main()**

**{**

**int Set\_A[10],n,x1;**

**cout<<"how many elements you want to enter"<<endl;**

**cin>>n;**

**int ch1;**

**char ch='y';**

**while(ch=='y'){**

**cout<<"1.input and display:"<<endl;**

**cout<<"2.binary search of the element that whether it belongs to the set or not:"<<endl;**

**cout<<"3.Powerset:"<<endl;**

**cout<<"enter your choice:"<<endl;**

**cin>>ch1;**

**switch(ch1){**

**case 1: set(n,Set\_A);**

**break;**

**case 2: cout<<"enter the element to be search";**

**cin>>x1;**

**cout<<"The element belongs to these set"<<" "<<ismember(Set\_A,n,x1);**

**cout<<"\n Note: 1is representing the true and 0 is representing false\n";**

**break;**

**case 3: cout<<"powerset of the Set\_A is:\n";**

**powerset(Set\_A,n);**

**break;**

**}**

**cout<<"\n do u want to continue";**

**cin>>ch;**

**}**

**return 0;**

**}**

**Output:**

**how many elements you want to enter**

**6**

**1.input and display:**

**2.binary search of the element that whether it belongs to the set or not:**

**3.Powerset:**

**enter your choice:**

**1**

**enter the elements in the Set\_A**

**1**

**2**

**3**

**4**

**5**

**8**

**Set\_A 1 2 3 4 5 8**

**cardinality of the entered set is6**

**do u want to continuey**

**1.input and display:**

**2.binary search of the element that whether it belongs to the set or not:**

**3.Powerset:**

**enter your choice:**

**2**

**enter the element to be search8**

**The element belongs to these set 1**

**Note: 1is representing the true and 0 is representing false**

**do u want to continue y**

**1.input and display:**

**2.binary search of the element that whether it belongs to the set or not:**

**3.Powerset:**

**enter your choice:**

**3**

**powerset of the Set\_A is:**

**Number of elements in the powerset of Set\_A including phi element**

**64{}**

**1,**

**2,**

**1,2,**

**3,**

**1,3,**

**2,3,**

**1,2,3,**

**4,**

**1,4,**

**2,4,**

**1,2,4,**

**3,4,**

**1,3,4,**

**2,3,4,**

**1,2,3,4,**

**5,**

**1,5,**

**2,5,**

**1,2,5,**

**3,5,**

**1,3,5,**

**2,3,5,**

**1,2,3,5,**

**4,5,**

**1,4,5,**

**2,4,5,**

**1,2,4,5,**

**3,4,5,**

**1,3,4,5,**

**2,3,4,5,**

**1,2,3,4,5,**

**8,**

**1,8,**

**2,8,**

**1,2,8,**

**3,8,**

**1,3,8,**

**2,3,8,**

**1,2,3,8,**

**4,8,**

**1,4,8,**

**2,4,8,**

**1,2,4,8,**

**3,4,8,**

**1,3,4,8,**

**2,3,4,8,**

**1,2,3,4,8,**

**5,8,**

**1,5,8,**

**2,5,8,**

**1,2,5,8,**

**3,5,8,**

**1,3,5,8,**

**2,3,5,8,**

**1,2,3,5,8,**

**4,5,8,**

**1,4,5,8,**

**2,4,5,8,**

**1,2,4,5,8,**

**3,4,5,8,**

**1,3,4,5,8,**

**2,3,4,5,8,**

**1,2,3,4,5,8,**

**do u want to continue n**

***Q2.Create a class SET and take two sets as input from user to perform following SET Operations***

1. ***Subset: check whether one set is a subset of other or not.***
2. ***Union and intersection of two Sets.***
3. ***Complement: Assume Universal Set as per the input elements from the user.***
4. ***Set Difference and Symmetric Difference between two SETS.***
5. ***Cartesian Product of Sets.***

**Solution:**

**#include<iostream>**

**using namespace std;**

**class SET**

**{**

**private:**

**int a[20],size;**

**public:**

**int getinput()**

**{**

**cout<<"enter the cardinality of set\n";**

**cin>>size;**

**cout<<"enter the elements in array\n";**

**for(int i=0;i<size;i++)**

**cin>>a[i];**

**return size;**

**}**

**void display()**

**{**

**int i,j,k;**

**for(i=0;i<size-1;i++)**

**{**

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

**{**

**if(a[i]==a[j])**

**{**

**for(k=j;k<=size-1;k++)**

**a[k]=a[k+1];**

**size--;**

**}**

**else**

**j++;**

**}**

**}**

**for(i=0;i<size;i++)**

**{**

**cout<<a[i]<<" ";**

**}**

**cout<<endl;**

**}**

**bool subset(SET setA )**

**{**

**int i,j;**

**for( i=0;i<size;i++)**

**{**

**for(j=0;j<setA.size;j++)**

**{**

**if(a[i]==setA.a[j])**

**break;**

**}**

**if(j==setA.size)**

**{**

**return false;**

**}**

**}**

**return true;**

**}**

**SET Union(SET setA)**

**{**

**SET setB;**

**int k=size+setA.size;**

**setB.size=k;**

**int i,j;**

**for(i=0,k=0;i<size;i++,k++)**

**{**

**setB.a[k]=a[i];**

**}**

**for(j=0;j<setA.size;k++,j++)**

**{**

**setB.a[k]=setA.a[j];**

**}**

**return setB;**

**}**

**SET Intersection(SET setA)**

**{**

**SET setB;**

**int i,j,k=0;**

**for(i=0;i<size;i++)**

**{**

**for(j=0;j<setA.size;j++)**

**{**

**if(a[i]==setA.a[j])**

**{**

**setB.a[k]=a[i];**

**k++;**

**}**

**}**

**}**

**if(k==0)**

**{**

**cout<<"intersection of two sets is an empty set\n";**

**}**

**setB.size=k;**

**return setB;**

**}**

**SET complement()**

**{**

**SET setA;**

**int i,j,k=0,counter=0;**

**SET setU;**

**cout<<"Universal set......\n";**

**setU.size=setU.getinput();**

**cout<<"elements in universal set are-> ";**

**setU.display();**

**for(i=0;i<setU.size;i++)**

**{**

**for(j=0;j<size;j++)**

**{**

**if(setU.a[i]==a[j])**

**{**

**counter++;**

**}**

**}**

**if(counter==0)**

**{**

**setA.a[k]=setU.a[i];**

**k++;**

**}**

**counter=0;**

**}**

**setA.size=k;**

**return setA;**

**}**

**SET operator-(SET setA)**

**{**

**SET setB;**

**int i,j,k=0,counter=0;**

**for(i=0;i<size;i++)**

**{**

**for(j=0;j<setA.size;j++)**

**{**

**if(a[i]==setA.a[j])**

**{**

**counter++;**

**}**

**}**

**if(counter==0)**

**{**

**setB.a[k]=a[i];**

**k++;**

**}**

**counter=0;**

**}**

**setB.size=k;**

**return setB;**

**}**

**void operator\*(SET setA)**

**{**

**SET setB;**

**int i,j,k=0;**

**for(i=0;i<size;i++)**

**{**

**for(j=0;j<setA.size;j++,k++)**

**{**

**cout<<"{"<<a[i]<<" , "<<setA.a[j]<<"}"<<endl;**

**}**

**cout<<endl;**

**}**

**}**

**};**

**int main()**

**{**

**SET setA;**

**SET setB;**

**SET setC;**

**SET setD;**

**int ch1,x,y,z;**

**bool p;**

**char ch='y';**

**while(ch=='y')**

**{**

**cout<<"1.INPUT and DISPLAY\n";**

**cout<<"2.SUBSET\n";**

**cout<<"3.Union and intersection\n";**

**cout<<"4.complement\n";**

**cout<<"5.set difference and symmetric difference\n";**

**cout<<"6.cartesian product\n";**

**cout<<"enter the choice\n";**

**cin>>ch1;**

**switch(ch1)**

**{**

**case 1:cout<<"SetA......\n";**

**x=setA.getinput();**

**cout<<"SetB......\n";**

**y=setB.getinput();**

**cout<<"elements in setA are\n";**

**setA.display();**

**cout<<"elements in setB are\n";**

**setB.display();**

**break;**

**case 2:if(x<=y)**

**{**

**p=setA.subset(setB);**

**if(p)**

**{**

**cout<<"setA is the subset of setB\n";**

**}**

**else**

**{**

**cout<<"set A is not the subset of setB\n";**

**}**

**}**

**else if(y<x)**

**{**

**p=setB.subset(setA);**

**if(p)**

**{**

**cout<<"setB is the subset of setA\n";**

**}**

**else**

**{**

**cout<<"set B is not the subset of setA\n";**

**}**

**}**

**break;**

**case 3:cout<<"union of 2 sets ->";**

**setC=setA.Union(setB);**

**setC.display();**

**cout<<"intersection of 2 sets -> ";**

**setD=setA.Intersection(setB);**

**setD.display();**

**break;**

**case 4:setC=setA.complement();**

**cout<<"complement of setA\n";**

**setC.display();**

**setD=setB.complement();**

**cout<<"complement of setB\n";**

**setD.display();**

**break;**

**case 5:setC=setB-setA;**

**cout<<"set difference B and A\n";**

**setC.display();**

**cout<<"symmetric difference of B and A \n";**

**setD=(setB.Union(setA))-(setB.Intersection(setA));**

**setD.display();**

**break;**

**case 6:cout<<"cartesian product of setA and setB \n";**

**setA\*setB;**

**break;**

**}**

**cout<<"Do you want to continue\n";**

**cin>>ch;**

**}**

**}**

**Output:**

**1.INPUT and DISPLAY**

**2.SUBSET**

**3.Union and intersection**

**4.complement**

**5.set difference and symmetric difference**

**6.cartesian product**

**enter the choice**

**1**

**SetA......**

**enter the cardinality of set**

**6**

**enter the elements in array**

**1 2 6 7 3 8**

**SetB......**

**enter the cardinality of set**

**8**

**enter the elements in array**

**1 2 3 4 5 6 7 8**

**elements in setA are**

**1 2 6 7 3 8**

**elements in setB are**

**1 2 3 4 5 6 7 8**

**Do you want to continue**

**y**

**1.INPUT and DISPLAY**

**2.SUBSET**

**3.Union and intersection**

**4.complement**

**5.set difference and symmetric difference**

**6.cartesian product**

**enter the choice**

**2**

**setA is the subset of setB**

**Do you want to continue**

**y**

**1.INPUT and DISPLAY**

**2.SUBSET**

**3.Union and intersection**

**4.complement**

**5.set difference and symmetric difference**

**6.cartesian product**

**enter the choice**

**3**

**union of 2 sets ->1 2 6 7 3 8 4 5**

**intersection of 2 sets -> 1 2 6 7 3 8**

**Do you want to continue**

**y**

**1.INPUT and DISPLAY**

**2.SUBSET**

**3.Union and intersection**

**4.complement**

**5.set difference and symmetric difference**

**6.cartesian product**

**enter the choice**

**4**

**Universal set......**

**enter the cardinality of set**

**12**

**enter the elements in array**

**1 2 3 4 5 6 7 8 9 10 11 12**

**elements in universal set are-> 1 2 3 4 5 6 7 8 9 10**

**complement of setA**

**4 5 9 10 11 12**

**Universal set......**

**enter the cardinality of set**

**12**

**enter the elements in array**

**1 2 3 4 5 6 7 8 9 10 11 12**

**elements in universal set are-> 1 2 3 4 5 6 7 8 9 10**

**complement of setB**

**9 10 11 12**

**Do you want to continue**

**y**

**1.INPUT and DISPLAY**

**2.SUBSET**

**3.Union and intersection**

**4.complement**

**5.set difference and symmetric difference**

**6.cartesian product**

**enter the choice**

**5**

**set difference B and A**

**4 5**

**symmetric difference of B and A**

**4 5**

**Do you want to continue**

**y**

**1.INPUT and DISPLAY**

**2.SUBSET**

**3.Union and intersection**

**4.complement**

**5.set difference and symmetric difference**

**6.cartesian product**

**enter the choice**

**6**

**cartesian product of setA and setB**

**{1 , 1}**

**{1 , 2}**

**{1 , 3}**

**{1 , 4}**

**{1 , 5}**

**{1 , 6}**

**{1 , 7}**

**{1 , 8}**

**{2 , 1}**

**{2 , 2}**

**{2 , 3}**

**{2 , 4}**

**{2 , 5}**

**{2 , 6}**

**{2 , 7}**

**{2 , 8}**

**{6 , 1}**

**{6 , 2}**

**{6 , 3}**

**{6 , 4}**

**{6 , 5}**

**{6 , 6}**

**{6 , 7}**

**{6 , 8}**

**{7 , 1}**

**{7 , 2}**

**{7 , 3}**

**{7 , 4}**

**{7 , 5}**

**{7 , 6}**

**{7 , 7}**

**{7 , 8}**

**{3 , 1}**

**{3 , 2}**

**{3 , 3}**

**{3 , 4}**

**{3 , 5}**

**{3 , 6}**

**{3 , 7}**

**{3 , 8}**

**{8 , 1}**

**{8 , 2}**

**{8 , 3}**

**{8 , 4}**

**{8 , 5}**

**{8 , 6}**

**{8 , 7}**

**{8 , 8}**

**Do you want to continue**

**n**

***Q3.Create a class RELATION, use Matrix notation to represent a relation. Include functions to check if a relation is Reflexive,Symmertic,Anti-Symmetric and Transitive. Write a program to use this class.***

**Solution:**

**#include <iostream>**

**using namespace std;**

**class relation{**

**private:**

**int arr[20][20];**

**int x;**

**public:**

**void getinput()**

**{**

**cout<<"enter the order of matrix of the relation";**

**cin>>x;**

**cout<<"enter the elements in matrix\n";**

**for(int i=0;i<x;i++){**

**for(int j=0;j<x;j++)**

**cin>>arr[i][j];**

**}**

**}**

**void display()**

**{**

**for(int i=0;i<x;i++){**

**for(int j=0;j<x;j++)**

**cout<<arr[i][j]<<" ";**

**cout<<endl;**

**}**

**}**

**bool reflexive()**

**{**

**for(int i=0;i<x;i++)**

**{**

**if (arr[i][i]!=1)**

**return false;**

**}**

**return true;**

**}**

**bool symmetric()**

**{**

**for(int i=0;i<x;i++)**

**{**

**for(int j=0;j<x;j++)**

**{**

**if(arr[i][j]==1 && arr[j][i]!=1)**

**return false;**

**}**

**}**

**return true;**

**}**

**bool antisymmetric()**

**{**

**for(int i=0;i<x;i++)**

**{**

**for(int j=0;j<x;j++)**

**{**

**if(arr[i][j]==1 && arr[j][i]==1 && i!=j)**

**{**

**return false;**

**}**

**}**

**}**

**return true;**

**}**

**bool transitive()**

**{**

**for(int i=0;i<x;i++)**

**{**

**for(int j=0;j<x;j++)**

**{**

**for(int k=0;k<x;k++)**

**{**

**if(arr[i][j]==1 && arr[j][k]==1 && arr[i][k]==1)**

**return true;**

**}**

**}**

**}**

**return false;**

**}**

**};**

**int main()**

**{**

**relation r1;**

**int ch1;**

**char ch='y';**

**while(ch=='y')**

**{**

**cout<<"1.Input and display\n";**

**cout<<"2.reflexive relation\n";**

**cout<<"3.symmetric relation\n";**

**cout<<"4.anti-symmetric relation\n";**

**cout<<"5.transitive relation\n";**

**cout<<"enter the choice\n";**

**cin>>ch1;**

**switch(ch1)**

**{**

**case 1:cout<<"r1 matrix.......\n";**

**r1.getinput();**

**cout<<"matrix r1 is......\n";**

**r1.display();**

**break;**

**case 2:if(r1.reflexive())**

**cout<<"r1 relation is reflexive\n";**

**else**

**cout<<"r1 relation is not reflexive\n";**

**break;**

**case 3:if(r1.symmetric())**

**cout<<"r1 relation is symmetric\n";**

**else**

**cout<<"r1 relation is not symmetric\n";**

**break;**

**case 4:if(r1.antisymmetric())**

**cout<<"r1 relation is anti-symmetric\n";**

**else**

**cout<<"r1 relation is not anti-symmetric\n";**

**break;**

**case 5:if(r1.transitive())**

**cout<<"r1 relation is transitive\n";**

**else**

**cout<<"r1 relation is not transitive\n";**

**break;**

**}**

**cout<<"Do you want to continue\n";**

**cin>>ch;**

**}**

**return 0;**

**}**

**Output:**

**1.Input and display**

**2.reflexive relation**

**3.symmetric relation**

**4.anti-symmetric relation**

**5.transitive relation**

**enter the choice**

**1**

**r1 matrix.......**

**enter the order of matrix of the relation 3**

**enter the elements in matrix**

**1 0 0**

**1 1 1**

**1 0 1**

**matrix r1 is......**

**1 0 0**

**1 1 1**

**1 0 1**

**Do you want to continue**

**y**

**1.Input and display**

**2.reflexive relation**

**3.symmetric relation**

**4.anti-symmetric relation**

**5.transitive relation**

**enter the choice**

**2**

**r1 relation is reflexive**

**Do you want to continue**

**y**

**1.Input and display**

**2.reflexive relation**

**3.symmetric relation**

**4.anti-symmetric relation**

**5.transitive relation**

**enter the choice**

**3**

**r1 relation is not symmetric**

**Do you want to continue**

**y**

**1.Input and display**

**2.reflexive relation**

**3.symmetric relation**

**4.anti-symmetric relation**

**5.transitive relation**

**enter the choice**

**4**

**r1 relation is anti-symmetric**

**Do you want to continue**

**y**

**1.Input and display**

**2.reflexive relation**

**3.symmetric relation**

**4.anti-symmetric relation**

**5.transitive relation**

**enter the choice**

**5**

**r1 relation is transitive**

**Do you want to continue**

**n**

***Q4.Use the functions defined in Question to check whether the given relation is:***

1. ***Equivalent,or***
2. ***Partial order relation,or***
3. ***None.***

**Solution:**

**#include <iostream>**

**using namespace std;**

**class relation{**

**private:**

**int arr[20][20];**

**int x;**

**public:**

**void getinput()**

**{**

**cout<<"enter the order of matrix of the relation";**

**cin>>x;**

**cout<<"enter the elements in matrix\n";**

**for(int i=0;i<x;i++){**

**for(int j=0;j<x;j++)**

**cin>>arr[i][j];**

**}**

**}**

**void display()**

**{**

**for(int i=0;i<x;i++){**

**for(int j=0;j<x;j++)**

**cout<<arr[i][j]<<" ";**

**cout<<endl;**

**}**

**}**

**bool reflexive()**

**{**

**for(int i=0;i<x;i++)**

**{**

**if (arr[i][i]!=1)**

**return false;**

**}**

**return true;**

**}**

**bool symmetric()**

**{**

**for(int i=0;i<x;i++)**

**{**

**for(int j=0;j<x;j++)**

**{**

**if(arr[i][j]==1 && arr[j][i]!=1)**

**return false;**

**}**

**}**

**return true;**

**}**

**bool antisymmetric()**

**{**

**for(int i=0;i<x;i++)**

**{**

**for(int j=0;j<x;j++)**

**{**

**if(arr[i][j]==1 && arr[j][i]==1 && i!=j)**

**{**

**return false;**

**}**

**}**

**}**

**return true;**

**}**

**bool transitive()**

**{**

**for(int i=0;i<x;i++)**

**{**

**for(int j=0;j<x;j++)**

**{**

**for(int k=0;k<x;k++)**

**{**

**if(arr[i][j]==1 && arr[j][k]==1 && arr[i][k]==1)**

**return true;**

**}**

**}**

**}**

**return false;**

**}**

**};**

**int main()**

**{**

**relation r1;**

**int ch1;**

**bool a,b;**

**char ch='y';**

**while(ch=='y')**

**{**

**cout<<"1.Input and display\n";**

**cout<<"2.Equivalence relation\n";**

**cout<<"3.Prtial-order relation\n";**

**cout<<"enter the choice\n";**

**cin>>ch1;**

**switch(ch1)**

**{**

**case 1:cout<<"r1 matrix.......\n";**

**r1.getinput();**

**cout<<"matrix r1 is......\n";**

**r1.display();**

**break;**

**case 2:a=(r1.reflexive() && r1.symmetric() && r1.transitive());**

**if(a)**

**cout<<"r1 relation is equivalence relation\n";**

**else**

**cout<<"r1 relation is not equivalence relation\n";**

**break;**

**case 3:b=(r1.reflexive() && r1.antisymmetric() && r1.transitive());**

**if(b)**

**cout<<"r1 relation is partial-ordered relation\n";**

**else**

**cout<<"r1 relation is not partial-ordered relation\n";**

**break;**

**}**

**cout<<"Do you want to continue\n";**

**cin>>ch;**

**}**

**if(!a && !b)**

**cout<<"None\n";**

**return 0;**

**}**

**Output:**

**1.Input and display**

**2.Equivalence relation**

**3.Prtial-order relation**

**enter the choice**

**1**

**r1 matrix.......**

**enter the order of matrix of the relation 3**

**enter the elements in matrix**

**1 0 0**

**1 1 1**

**0 1 1**

**matrix r1 is......**

**1 0 0**

**1 1 1**

**0 1 1**

**Do you want to continue**

**y**

**1.Input and display**

**2.Equivalence relation**

**3.Prtial-order relation**

**enter the choice**

**2**

**r1 relation is not equivalence relation**

**Do you want to continue**

**y**

**1.Input and display**

**2.Equivalence relation**

**3.Prtial-order relation**

**enter the choice**

**3**

**r1 relation is not partial-ordered relation**

**Do you want to continue**

**n**

**None**

***Q5. Write a program to generate the Fibonacci series using recursion.***

**Solution:**

**#include<iostream>**

**using namespace std;**

**int fibonacci(int n)**

**{**

**if((n==1)||(n==0))**

**{**

**return n;**

**}**

**else**

**{**

**return(fibonacci(n-1)+fibonacci(n-2));**

**}**

**}**

**int main()**

**{**

**int n,a=0,i;**

**cout<<"enter the number of terms for fibonacci series:";**

**cin>>n;**

**cout<<"\n Fibonacci series is as follows\n";**

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

**{**

**cout<<" "<<fibonacci(a);**

**a++;**

**}**

**return 0;**

**}**

**Output:**

**enter the number of terms for fibonacci series: 5**

**Fibonacci series is as follows**

**0 1 1 2 3 5**

***Q6.Write a program to implement Tower of Hanoi.***

**Solution:**

**#include <iostream>**

**using namespace std;**

**void tower(int n, char source, char des, char via){**

**if(n == 1){**

**cout << "Moved the ring "<< n << " from " << source << " to " << des << ".\n";**

**}**

**else{**

**tower(n-1,source, via, des);**

**cout << "Moved the ring " << n << " from " << source << " to " << des << ".\n";**

**tower(n-1,via,des,source);**

**}**

**}**

**int main()**

**{**

**int rings;**

**char source='A';**

**char des = 'B';**

**char via = 'C';**

**cout << "Enter the number of rings : ";**

**cin >> rings;**

**tower(rings,source,des,via);**

**return 0;**

**}**

**Output:**

**Enter the number of rings : 4**

**Moved the ring 1 from A to C.**

**Moved the ring 2 from A to B.**

**Moved the ring 1 from C to B.**

**Moved the ring 3 from A to C.**

**Moved the ring 1 from B to A.**

**Moved the ring 2 from B to C.**

**Moved the ring 1 from A to C.**

**Moved the ring 4 from A to B.**

**Moved the ring 1 from C to B.**

**Moved the ring 2 from C to A.**

**Moved the ring 1 from B to A.**

**Moved the ring 3 from C to B.**

**Moved the ring 1 from A to C.**

**Moved the ring 2 from A to B.**

**Moved the ring 1 from C to B.**

***Q7.Write a program to implement binary search using recursion.***

**Solution:**

**#include<iostream>**

**using namespace std;**

**void input(int x[],int n)**

**{**

**cout<<"enter the elements of array\n";**

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

**{**

**cin>>x[i];**

**}**

**}**

**void binarysearch(int x[],int n,int x1,int low,int high)**

**{**

**int mid=0;**

**if(low>high)**

**{**

**cout<<"element not found"<<endl;**

**return;**

**}**

**else**

**{**

**mid=(low+high)/2;**

**if(x[mid]==x1)**

**{**

**cout<<"element is found at position"<<(mid+1)<<endl;**

**return;**

**}**

**else if(x[mid]>x1)**

**binarysearch(x,n,x1,low,mid-1);**

**else**

**binarysearch(x,n,x1,mid+1,high);**

**}**

**}**

**int main()**

**{**

**int a[10];**

**int n;**

**cout<<"how many elements do you want to enter\n";**

**cin>>n;**

**input(a,n);**

**char ch='y';**

**int x1;**

**while(ch=='y')**

**{**

**cout<<"enter the element you want to search ";**

**cin>>x1;**

**binarysearch(a,n,x1,0,n-1);**

**cout<<"do you want to continue";**

**cin>>ch;**

**}**

**return 0;**

**}**

**Output:**

**how many elements do you want to enter**

**6**

**enter the elements of array**

**24**

**67**

**89**

**98**

**99**

**200**

**enter the element you want to search 200**

**element is found at position6**

**do you want to continue y**

**enter the element you want to search 45**

**element not found**

**do you want to continue n**

***Q8.Write a program to implement Bubble Sort.***

**Solution:**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**int a[50],n,i,j,temp;**

**cout<<"Enter the size of array: ";**

**cin>>n;**

**cout<<"Enter the array elements: ";**

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

**cin>>a[i];**

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

**{**

**for(j=0;j<(n-i);j++)**

**if(a[j]>a[j+1])**

**{**

**temp=a[j];**

**a[j]=a[j+1];**

**a[j+1]=temp;**

**}**

**}**

**cout<<"Array after bubble sort:";**

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

**cout<<" "<<a[i];**

**return 0;**

**}**

**Output:**

**Enter the size of array: 6**

**Enter the array elements: 5 1 8 4 3 9**

**Array after bubble sort: 1 3 4 5 8 9**

***Q9.Write a program to implement Insertion Sort.***

**Solution:**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**int i,j,n,temp,a[30];**

**cout<<"Enter the size of array:";**

**cin>>n;**

**cout<<"\nEnter the elements\n";**

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

**{**

**cin>>a[i];**

**}**

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

**{**

**temp=a[i];**

**j=i-1;**

**while((temp<a[j])&&(j>=0))**

**{**

**a[j+1]=a[j]; //moves element forward**

**j=j-1;**

**}**

**a[j+1]=temp; //insert element in proper place**

**}**

**cout<<"\n After insertion sort:\n";**

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

**{**

**cout<<a[i]<<" ";**

**}**

**return 0;**

**}**

**Output:**

**Enter the size of array:8**

**Enter the elements**

**6 8 5 3 2 9 12 4**

**After insertion sort:**

**2 3 4 5 6 8 9 12**

***Q10.Write a program that generates all the permutations of a given set of digits, with or without repetition.***

**Solution:**

**#include<iostream>**

**using namespace std;**

**void swap(int \*a,int \*b)**

**{**

**int temp;**

**temp=\*a;**

**\*a=\*b;**

**temp=\*b;**

**}**

**void perm(int A[],int b,int n)**

**{**

**if(b==n-1)**

**{**

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

**{**

**cout<<A[i];**

**}**

**cout<<endl;**

**}**

**else**

**{**

**for(int i=b;i<n;i++)**

**{**

**swap(A[i],A[b]);**

**perm(A,b+1,n);**

**swap(A[i],A[b]);**

**}**

**}**

**}**

**void permrep(int A[],int B[],int b,int n)**

**{**

**if(b==n)**

**{**

**for(int i=0;i<b;i++)**

**{**

**cout<<B[i];**

**}**

**cout<<endl;**

**}**

**else**

**{**

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

**{**

**B[b]=A[i];**

**permrep(A,B,b+1,n);**

**}**

**}**

**}**

**int main ()**

**{**

**int ch;**

**int A[50],B[50];**

**int n,b=0;**

**cout<<"Enter the size of set ";**

**cin>>n;**

**cout<<"Enter the elements of set"<<endl;**

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

**{**

**cin>>A[i];**

**}**

**cout<<"The set entered is {";**

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

**{**

**if(i==0)**

**cout<<A[i];**

**else**

**cout<<","<<A[i];**

**}**

**cout<<"}";**

**cout<<endl;**

**cout<<"Enter your choice "<<endl<<"1.permutation with repetition"<<endl<<"2.permutation without repetition"<<endl<<"3.exit"<<endl;**

**cin>>ch;**

**char ch1='y';**

**switch(ch)**

**{**

**case 1: permrep(A,B,b,n);**

**break;**

**case 2: perm(A,b,n);**

**break;**

**case 3: exit(0);**

**default:cout<<"wrong choice";**

**break;**

**}**

**return 0;**

**}**

**Output:**

**Enter the size of set 4**

**Enter the elements of set**

**1**

**3**

**7**

**9**

**The set entered is {1,3,7,9}**

**Enter your choice**

**1.permutation with repetition**

**2.permutation without repetition**

**3.exit**

**2**

**1379**

**1397**

**1739**

**1793**

**1973**

**1937**

**3179**

**3197**

**3719**

**3791**

**3971**

**3917**

**7319**

**7391**

**7139**

**7193**

**7913**

**7931**

**9371**

**9317**

**9731**

**9713**

**9173**

**9137**

***Q11.Write a to calculate Permutation and Combination for an input value of n and r Uing recursive formula of nCr and nPr.***

**Solution:**

**#include<iostream>**

**using namespace std;**

**int fact(int num)**

**{**

**if(num==1)**

**{**

**return 1;**

**}**

**else**

**{**

**return num \* fact(num-1);**

**}**

**}**

**int main()**

**{**

**int n,r,nPr,nCr;**

**cout<<"Enter n: ";**

**cin>>n;**

**cout<<"Enter r: ";**

**cin>>r;**

**nPr=fact(n)/fact(n-r);**

**nCr=nPr/fact(r);**

**cout<<"nPr is: "<<nPr<<endl;**

**cout<<"nCr is: "<<nCr;**

**return 0;**

**}**

**Output:**

**Enter n: 6**

**Enter r: 3**

**nPr is: 120**

**nCr is: 20**

***Q12.For any number n,Write a program to list all the solutions of the equation x1+x2+x3+…+xn=C where C is a constant (C<<10) and x1,x2,x3,…xn are nonnegative integers using brute force strategy.***

**Solution:**

**#include<iostream>**

**using namespace std ;**

**int nCr(int n, int r)**

**{**

**if(r==0 || r==n)**

**return 1;**

**else**

**return ( nCr(n-1,r-1) +nCr(n-1,r)) ;**

**}**

**int main()**

**{**

**int n ,k;**

**cout<<"\nx1+x2+x3+…+xn=c"<<endl;**

**cout<<"enter the number of terms"<<endl;**

**cin>>n;**

**cout<<"enter the value of the constant"<<endl;**

**cin>>k;**

**cout<<"\nNumber of possible solutions of the given equation is:";**

**cout<<nCr(n+k-1,k);**

**return 0 ;**

**}**

**Output:**

**x1+x2+x3+…+xn=c**

**enter the number of terms**

**6**

**enter the value of the constant**

**8**

**Number of possible solutions of the given equation is:1287**

***Q13.Write a program to accept the truth values of variable x and y,and print the truth table of the following logical operations:***

1. ***Conjunction***
2. ***Disjunction***
3. ***Exclusive OR***
4. ***Conditional***
5. ***Bi-conditional***
6. ***Exclusive NOR***
7. ***Negation***
8. ***NAND***
9. ***NOR***

**Solution:**

**#include <iostream>**

**using namespace std;**

**class Truthtable**

**{**

**private:**

**int x,y;**

**public:**

**void conjunction()**

**{**

**cout<<"x y x&&y"<<endl;**

**for(int i=0;i<2;i++)**

**{**

**for(int j=0;j<2;j++)**

**{**

**x=i;y=j;**

**cout<<x<<" "<<y<<" "<<x\*y<<endl;;**

**}**

**}**

**}**

**void disjunction()**

**{**

**int z;**

**cout<<"x y x||y"<<endl;**

**for(int i=0;i<2;i++)**

**{**

**for(int j=0;j<2;j++)**

**{**

**x=i;y=j,z=x||y;**

**cout<<x<<" "<<y<<" "<<z<<endl;;**

**}**

**}**

**}**

**void exclusiveOR()**

**{**

**int z;**

**cout<<"x y x XOR y"<<endl;**

**for(int i=0;i<2;i++)**

**{**

**for(int j=0;j<2;j++)**

**{**

**x=i;y=j,z=((!x&&y)||(x&&!y));**

**cout<<x<<" "<<y<<" "<<z<<endl;;**

**}**

**}**

**}**

**void Conditional()**

**{**

**int z;**

**cout<<"x y x=>y"<<endl;**

**for(int i=0;i<2;i++)**

**{**

**for(int j=0;j<2;j++)**

**{**

**x=i;y=j,z=!(x&&!y);**

**cout<<x<<" "<<y<<" "<<z<<endl;;**

**}**

**}**

**}**

**void Biconditional()**

**{**

**int z;**

**cout<<"x y x<=>y"<<endl;**

**for(int i=0;i<2;i++)**

**{**

**for(int j=0;j<2;j++)**

**{**

**x=i;y=j,z=(x&&y)||(!x&&!y);**

**cout<<x<<" "<<y<<" "<<z<<endl;;**

**}**

**}**

**}**

**void exclusiveNOR()**

**{**

**int z;**

**cout<<"x y x XNOR y"<<endl;**

**for(int i=0;i<2;i++)**

**{**

**for(int j=0;j<2;j++)**

**{**

**x=i;y=j,z=(x&&y)||(!x&&!y);**

**cout<<x<<" "<<y<<" "<<z<<endl;;**

**}**

**}**

**}**

**void Negation()**

**{**

**int z;**

**cout<<"x ~x"<<endl;**

**for(int i=0;i<2;i++)**

**{**

**x=i;**

**z=!x;**

**cout<<x<<" "<<z<<endl;**

**}**

**}**

**void NAND()**

**{**

**int z;**

**cout<<"x y x NAND y\n";**

**for(int i=0;i<2;i++)**

**{**

**for(int j=0;j<2;j++)**

**{**

**x=i;y=j;z=!(x&&y);**

**cout<<x<<" "<<y<<" "<<z<<endl;**

**}**

**}**

**}**

**void NOR()**

**{**

**int z;**

**cout<<"x y x NOR y\n";**

**for(int i=0;i<2;i++)**

**{**

**for(int j=0;j<2;j++)**

**{**

**x=i;y=j;z=!(x||y);**

**cout<<x<<" "<<y<<" "<<z<<endl;**

**}**

**}**

**}**

**};**

**int main()**

**{**

**Truthtable t1;**

**char ch='y';**

**int ch1;**

**while(ch=='y')**

**{**

**cout<<"1.Conjunction\n";**

**cout<<"2.Disjunction\n";**

**cout<<"3.Exclusive-OR\n";**

**cout<<"4.Conditional\n";**

**cout<<"5.Biconditional\n";**

**cout<<"6.Exclusive-NOR\n";**

**cout<<"7.Negation\n";**

**cout<<"8.NAND\n";**

**cout<<"9.NOR\n";**

**cout<<"Enter the choice\n";**

**cin>>ch1;**

**switch(ch1)**

**{**

**case 1:t1.conjunction();**

**break;**

**case 2:t1.disjunction();**

**break;**

**case 3:t1.exclusiveOR();**

**break;**

**case 4:t1.Conditional();**

**break;**

**case 5:t1.Biconditional();**

**break;**

**case 6:t1.exclusiveNOR();**

**break;**

**case 7:t1.Negation();**

**break;**

**case 8:t1.NAND();**

**break;**

**case 9:t1.NOR();**

**break;**

**}**

**cout<<"Do you want to continue\n";**

**cin>>ch;**

**}**

**return 0;**

**}**

**Output:**

**1.Conjunction**

**2.Disjunction**

**3.Exclusive-OR**

**4.Conditional**

**5.Biconditional**

**6.Exclusive-NOR**

**7.Negation**

**8.NAND**

**9.NOR**

**Enter the choice**

**1**

**x y x&&y**

**0 0 0**

**0 1 0**

**1 0 0**

**1 1 1**

**Do you want to continue**

**y**

**1.Conjunction**

**2.Disjunction**

**3.Exclusive-OR**

**4.Conditional**

**5.Biconditional**

**6.Exclusive-NOR**

**7.Negation**

**8.NAND**

**9.NOR**

**Enter the choice**

**2**

**x y x||y**

**0 0 0**

**0 1 1**

**1 0 1**

**1 1 1**

**Do you want to continue**

**y**

**1.Conjunction**

**2.Disjunction**

**3.Exclusive-OR**

**4.Conditional**

**5.Biconditional**

**6.Exclusive-NOR**

**7.Negation**

**8.NAND**

**9.NOR**

**Enter the choice**

**3**

**x y x XOR y**

**0 0 0**

**0 1 1**

**1 0 1**

**1 1 0**

**Do you want to continue**

**y**

**1.Conjunction**

**2.Disjunction**

**3.Exclusive-OR**

**4.Conditional**

**5.Biconditional**

**6.Exclusive-NOR**

**7.Negation**

**8.NAND**

**9.NOR**

**Enter the choice**

**4**

**x y x=>y**

**0 0 1**

**0 1 1**

**1 0 0**

**1 1 1**

**Do you want to continue**

**y**

**1.Conjunction**

**2.Disjunction**

**3.Exclusive-OR**

**4.Conditional**

**5.Biconditional**

**6.Exclusive-NOR**

**7.Negation**

**8.NAND**

**9.NOR**

**Enter the choice**

**5**

**x y x<=>y**

**0 0 1**

**0 1 0**

**1 0 0**

**1 1 1**

**Do you want to continue**

**y**

**1.Conjunction**

**2.Disjunction**

**3.Exclusive-OR**

**4.Conditional**

**5.Biconditional**

**6.Exclusive-NOR**

**7.Negation**

**8.NAND**

**9.NOR**

**Enter the choice**

**6**

**x y x XNOR y**

**0 0 1**

**0 1 0**

**1 0 0**

**1 1 1**

**Do you want to continue**

**y**

**1.Conjunction**

**2.Disjunction**

**3.Exclusive-OR**

**4.Conditional**

**5.Biconditional**

**6.Exclusive-NOR**

**7.Negation**

**8.NAND**

**9.NOR**

**Enter the choice**

**7**

**x ~x**

**0 1**

**1 0**

**Do you want to continue**

**y**

**1.Conjunction**

**2.Disjunction**

**3.Exclusive-OR**

**4.Conditional**

**5.Biconditional**

**6.Exclusive-NOR**

**7.Negation**

**8.NAND**

**9.NOR**

**Enter the choice**

**8**

**x y x NAND y**

**0 0 1**

**0 1 1**

**1 0 1**

**1 1 0**

**Do you want to continue**

**y**

**1.Conjunction**

**2.Disjunction**

**3.Exclusive-OR**

**4.Conditional**

**5.Biconditional**

**6.Exclusive-NOR**

**7.Negation**

**8.NAND**

**9.NOR**

**Enter the choice**

**9**

**x y x NOR y**

**0 0 1**

**0 1 0**

**1 0 0**

**1 1 0**

**Do you want to continue**

**N**

***Q14.)Write a program to accept an input n from the user and graphically represent the values of T(n) where n varies from 0 to n for the recurrence relation.***

***Solution:***

***#include <iostream>***

***using namespace std;***

***int t1(int n)***

***{***

***if(n==0)***

***{***

***return 1;***

***}***

***else{***

***return(t1(n-1)+n);***

***}***

***}***

***int t2(int n1)***

***{***

***if(n1==0)***

***{***

***return 1;***

***}***

***else{***

***return(t2(n1-1)+n1\*n1);***

***}***

***}***

***int t3(int n2)***

***{***

***if(n2==0)***

***{***

***return 0;***

***}***

***else if(n2==1)***

***{***

***return 1;***

***}***

***else{***

***return(2\*t3(n2/2)+n2);***

***}***

***}***

***int main()***

***{***

***int k,ch ;***

***int result;***

***char ch1='y';***

***while(ch1=='y'){***

***cout<<"1.input"<<endl;***

***cout<<"2.first recurrence relation"<<endl;***

***cout<<"3.second recurrence relation"<<endl;***

***cout<<"4.third recurrence relation"<<endl;***

***cout<<"enter your choice"<<endl;***

***cin>>ch;***

***switch(ch){***

***case 1: cout<<"enter the value of number of terms"<<endl;***

***cin>>k;***

***break;***

***case 2: cout<<"first recurrence relation"<<endl;***

***result=t1(k);***

***cout<<"result is"<<result<<endl;***

***break;***

***case 3: cout<<"second recurrence relation"<<endl;***

***result=t2(k);***

***cout<<"result is"<<result<<endl;***

***break;***

***case 4: cout<<"third recurrence relation"<<endl;***

***result=t3(k);***

***cout<<"result is"<<result<<endl;***

***break;***

***}***

***cout<<"do u want to continue"<<endl;***

***cin>>ch1;***

***}***

***return 0;***

***}***

***Output:***

***enter your choice***

***1***

***enter the value of number of terms***

***3***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***2***

***first recurrence relation***

***result is7***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***4***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***2***

***first recurrence relation***

***result is11***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***5***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***2***

***first recurrence relation***

***result is16***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***0***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***3***

***second recurrence relation***

***result is1***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***1***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***3***

***second recurrence relation***

***result is2***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***2***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***3***

***second recurrence relation***

***result is6***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***3***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***3***

***second recurrence relation***

***result is15***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***4***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***3***

***second recurrence relation***

***result is31***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***5***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***3***

***second recurrence relation***

***result is56***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***0***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***4***

***third recurrence relation***

***result is0***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***1***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***4***

***third recurrence relation***

***result is1***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***2***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***4***

***third recurrence relation***

***result is4***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***3***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***4***

***third recurrence relation***

***result is5***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***4***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***4***

***third recurrence relation***

***result is12***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***1***

***enter the value of number of terms***

***5***

***do u want to continue***

***y***

***1.input***

***2.first recurrence relation***

***3.second recurrence relation***

***4.third recurrence relation***

***enter your choice***

***4***

***third recurrence relation***

***result is13***

***do u want to continue***

***n***

***Graphical Representation:***

|  |  |  |
| --- | --- | --- |
| S.NO | Value of n | T1 |
| 1 | n=0 | 1 |
| 2 | n=1 | 2 |
| 3 | n=2 | 4 |
| 4 | n=3 | 7 |
| 5 | n=4 | 11 |
| 6 | n=5 | 16 |

Table 1 and figure 1:Recurrence relation of T1:T(n)=T(n-1)+n,T(0)=1

|  |  |  |
| --- | --- | --- |
| S.NO | Value of n | T2 |
| 1 | n=0 | 1 |
| 2 | n=1 | 2 |
| 3 | n=2 | 6 |
| 4 | n=3 | 15 |
| 5 | n=4 | 31 |
| 6 | n=5 | 56 |

Table 2 and figure 2:Recurrence relation of T2:T(n)=T(n-1)+n2,T(0)=1

|  |  |  |
| --- | --- | --- |
| S.NO | Value of n | T3 |
| 1 | n=0 | 0 |
| 2 | n=1 | 1 |
| 3 | n=2 | 4 |
| 4 | n=3 | 5 |
| 5 | n=4 | 12 |
| 6 | n=5 | 13 |

Table 3 and figure 3:Recurrence relation of T3:T(n)=2\*T(n/2)+n,T(1)=1

***Q15.Write a Program to store a function (polynomial/exponential), and then evaluate the polynomial.(For example store f(n)=4n2+2n+9 in an array and for a given value of n,say n=5.evaluate(i.e. compare the value of f(5)).***

**Solution:**

**#include <iostream>**

**#include <math.h>**

**using namespace std;**

**int main()**

**{**

**int arr[20],deg,x,sum=0;**

**char ch;**

**do**

**{**

**cout<<"Enter the degree of the polynomial :";**

**cin>>deg;**

**for(int i=deg; i>=0;i--)**

**{**

**cout<<"Enter the coefficient of degree "<<i<<" :";**

**cin>>arr[i];**

**}**

**cout<<"Our required polynomial is :";**

**cout<<arr[deg]<<"x^"<<deg;**

**for(int i=deg-1;i>0;i--)**

**{**

**if(arr[i]>0)**

**cout<<"+"<<arr[i]<<"x^"<<i;**

**else**

**cout<<"-"<<arr[i]<<"x^"<<i;**

**}**

**cout<<"+"<<arr[0]<<"x^"<<0;**

**cout<<"\n Enter the value of x : ";**

**cin>>x;**

**for(int i=deg;i>=0;i--)**

**{**

**sum+=(arr[i]\*pow(x,i));**

**}**

**cout<<"\nThe solution of this polynomial is :"<<sum;**

**char ch;**

**cout<<"\n Do you want to continue?(y/n):";**

**cin>>ch;**

**}while(ch=='y'||ch=='Y');**

**}**

**Output:**

**Enter the degree of the polynomial :3**

**Enter the coefficient of degree 3 :4**

**Enter the coefficient of degree 2 :0**

**Enter the coefficient of degree 1 :2**

**Enter the coefficient of degree 0 :9**

**Our required polynomial is :4x^3-0x^2+2x^1+9x^0**

**Enter the value of x : 5**

**The solution of this polynomial is :519**

**Do you want to continue?(y/n):n**

***Q16. Write a program to represent Graphs using the Adjacency Matrices and check if it is a complete graph.***

***Solution:***

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**char choice;**

**int v,flag=0,q;**

**cout<<"enter the number of vertices";**

**cin>>v;**

**int ar[20][20];**

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

**{**

**for(int j=0;j<v;j++)**

**{**

**cout<<"\n How many edge from "<<(char)(97+i)<<" to "<<(char)(97+j)<<" - ";**

**cin>>ar[i][j];**

**}**

**}**

**cout<<"TheAdjacenc*y* matrix of graph is \n";**

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

**{**

**cout<<endl;**

**for(int j=0;j<v;j++)**

**cout<<ar[i][j]<<" ";**

**}**

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

**{**

**cout<<endl;**

**for(int j=0;j<v;j++)**

**{**

**if((i!=j) && (i<j))**

**{**

**if(ar[i][j]!=1)**

**{**

**flag=1;**

**break;**

**}**

**}**

**}**

**}**

**if(flag==1)**

**{**

**cout<<"This is a not complete graph\n";**

**}**

**else**

**{**

**cout<<"This is a complete graph\n";**

**}**

**return 0;**

**}**

**Output:**

**enter the number of vertices 4**

**How many edge from a to a - 0**

**How many edge from a to b - 1**

**How many edge from a to c - 1**

**How many edge from a to d - 1**

**How many edge from b to a - 1**

**How many edge from b to b - 0**

**How many edge from b to c - 1**

**How many edge from b to d - 1**

**How many edge from c to a - 1**

**How many edge from c to b - 1**

**How many edge from c to c - 0**

**How many edge from c to d - 1**

**How many edge from d to a - 1**

**How many edge from d to b - 1**

**How many edge from d to c - 1**

**How many edge from d to d - 0**

**The Adjacency matrix of graph is**

**0 1 1 1**

**1 0 1 1**

**1 1 0 1**

**1 1 1 0**

**This is a complete graph**

***Q17. Write a program to accept a directed graph G and compute the in-degree and out-degree of each vertex.***

***Solution:***

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**int arr[20][20];**

**int count=0;**

**int out=0;**

**int in=0;**

**int i,j;**

**int v,e;**

**cout<<" Enter the number of vertex :";**

**cin>>v;**

**cout<<"Enter the edges for the graph : "<<endl;**

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

**{**

**for(int j=0;j<v;j++)**

**{**

**cout<<"\n Enter the no. of edges ";**

**cout<<"from"<<" "<<(char)(i+97)<<" "<<"to vertex"<<" "<<(char)(j+97)<<" : ";**

**cin>>e;**

**if(e>0)**

**{**

**arr[i][j]=e;**

**}**

**else**

**arr[i][j]=0;**

**}**

**}**

**cout<<"\n The matrix you entered is : "<<endl;**

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

**{**

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

**{**

**cout<<arr[i][j]<<" ";**

**}**

**cout<<endl;**

**}**

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

**{**

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

**{**

**if(arr[i][j]>0)**

**{**

**out=out+arr[i][j];**

**}**

**}**

**cout<<"\n The out degree of the vertex is"<<(char)(i+97)<<out;**

**out=0;**

**}**

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

**{**

**for(int j=0;j<v;j++)**

**{**

**if(arr[j][i]>0)**

**{**

**in=in+arr[j][i];**

**}**

**}**

**cout<<"\n The in degree of the vertex is"<<(char)(i+97)<<in;**

**in=0;**

**}**

**return 0;**

**}**

**Output:**

**Enter the number of vertex :**

**4**

**Enter the edges for the graph :**

**Enter the no. of edges from a to vertex a : 1**

**Enter the no. of edges from a to vertex b : 1**

**Enter the no. of edges from a to vertex c : 1**

**Enter the no. of edges from a to vertex d : 1**

**Enter the no. of edges from b to vertex a : 0**

**Enter the no. of edges from b to vertex b : 1**

**Enter the no. of edges from b to vertex c : 1**

**Enter the no. of edges from b to vertex d : 1**

**Enter the no. of edges from c to vertex a : 1**

**Enter the no. of edges from c to vertex b : 0**

**Enter the no. of edges from c to vertex c : 0**

**Enter the no. of edges from c to vertex d : 1**

**Enter the no. of edges from d to vertex a : 1**

**Enter the no. of edges from d to vertex b : 1**

**Enter the no. of edges from d to vertex c : 0**

**Enter the no. of edges from d to vertex d : 1**

**The matrix you entered is :**

**1 1 1 1**

**0 1 1 1**

**1 0 0 1**

**1 1 0 1**

**The out degree of the vertex isa4**

**The out degree of the vertex isb3**

**The out degree of the vertex isc2**

**The out degree of the vertex isd3**

**The in degree of the vertex isa3**

**The in degree of the vertex isb3**

**The in degree of the vertex isc2**

**The in degree of the vertex isd4**

***Q18.Given a graph G,write a program to find the number of paths of length n between the source and destination entered by user.***

***Solution:***

**#include<iostream>**

**using namespace std;**

**void multiplication(int a1[50][50],int v,int pl,int source,int dest)**

**{**

**int a3[50][50],a2[50][50];**

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

**{**

**for(int j=0;j<v;j++)**

**{**

**a2[i][j]=a1[i][j];**

**}**

**}**

**if(pl==1)**

**{**

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

**{**

**for(int j=0;j<v;j++)**

**{**

**cout<< a1[i][j]<<" ";**

**}**

**cout<<endl;**

**}**

**}**

**else**

**{**

**for(int l=2;l<=pl;l++)**

**{**

**cout<<"\n The Matrix after multiplication is : ";**

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

**{**

**cout<<endl;**

**for(int j=0;j<v;j++)**

**{**

**a3[i][j]=0;**

**for(int k=0;k<v;k++)**

**{**

**a3[i][j]+=a1[i][k]\*a2[k][j];**

**}**

**cout<< a3[i][j]<<" ";**

**}**

**}**

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

**{**

**for(int j=0;j<v;j++)**

**{**

**a2[i][j]=a3[i][j];**

**}**

**}**

**cout<<endl<<endl;**

**}**

**cout<<"\n Enter the path between "<<char(source)<<" and "<<char(dest)<<" ";**

**source=source-97;**

**dest=dest-97;**

**cout<<a3[source][dest];**

**}**

**}**

**int main()**

**{**

**int pl;**

**int a[50][50];**

**int i,j;**

**int ch;**

**int v;**

**int length;**

**char source,dest;**

**cout<<"\n Enter the vertices : ";**

**cin>>v;**

**cout<<endl;**

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

**{**

**for(int j=0;j<v;j++)**

**{**

**cout<<"\n Enter the elements ";**

**cout<<(char)(i+97)<<" "<<"to vertex"<<" "<<(char)(j+97)<<" : ";**

**cin>>a[i][j];**

**}**

**}**

**cout<<"\n The matrix you entered is : "<<endl;**

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

**{**

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

**{**

**cout<<a[i][j]<<" ";**

**}**

**cout<<endl;**

**}**

**cout<<"\n Enter the path length: ";**

**cin>>pl;**

**cout<<endl;**

**cout<<"\n Please Enter the source : ";**

**cin>>source;**

**cout<<"\n Please Enter the destination : ";**

**cin>>dest;**

**multiplication(a,v,pl,source,dest);**

**return 0;**

**}**

**Output:**

**Enter the vertices : 4**

**Enter the elements a to vertex a : 0**

**Enter the elements a to vertex b : 1**

**Enter the elements a to vertex c : 1**

**Enter the elements a to vertex d : 1**

**Enter the elements b to vertex a : 1**

**Enter the elements b to vertex b : 0**

**Enter the elements b to vertex c : 1**

**Enter the elements b to vertex d : 1**

**Enter the elements c to vertex a : 1**

**Enter the elements c to vertex b : 1**

**Enter the elements c to vertex c : 0**

**Enter the elements c to vertex d : 1**

**Enter the elements d to vertex a : 1**

**Enter the elements d to vertex b : 1**

**Enter the elements d to vertex c : 1**

**Enter the elements d to vertex d : 0**

**The matrix you entered is :**

**0 1 1 1**

**1 0 1 1**

**1 1 0 1**

**1 1 1 0**

**Enter the path length: 6**

**Please Enter the source : a**

**Please Enter the destination : d**

**The Matrix after multiplication is :**

**3 2 2 2**

**2 3 2 2**

**2 2 3 2**

**2 2 2 3**

**The Matrix after multiplication is :**

**6 7 7 7**

**7 6 7 7**

**7 7 6 7**

**7 7 7 6**

**The Matrix after multiplication is :**

**21 20 20 20**

**20 21 20 20**

**20 20 21 20**

**20 20 20 21**

**The Matrix after multiplication is :**

**60 61 61 61**

**61 60 61 61**

**61 61 60 61**

**61 61 61 60**

**The Matrix after multiplication is :**

**183 182 182 182**

**182 183 182 182**

**182 182 183 182**

**182 182 182 183**

**Enter the path between a and d 182**

***Q19.Given an adjacency matrix of a graph,write a program to check whether a given set of vertices{v1,v2,v3………..vi}forms an Eucler path/Eucler Circuit.***

***Solution:***

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**char charr[50],choice;**

**int v,i,j,p=0,sum=0,flag=0,c=0;**

**cout<<"Enter number of vertices for a adjancency matrix \n";**

**cin>>v;**

**int arr[v][v],arr1[v];**

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

**{**

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

**{**

**cout<<"\n How many edge from "<<(char)(97+i)<<" to "<<(char)(97+j)<<" - ";**

**cin>>arr[i][j];**

**}**

**}**

**cout<<"\n THE ADJANCY MATRIX : \n ";**

**for(int m=0;m<v;m++)**

**{**

**cout<<endl;**

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

**cout<<arr[m][n]<<" ";**

**}**

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

**{**

**sum=0;**

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

**{**

**sum+=arr[i][j];**

**}**

**arr1[i]=sum;**

**}**

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

**{**

**cout<<"\n THE DEGREE OF " <<(char)(97+i) <<" -- "<<arr1[i]<<endl;**

**}**

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

**{**

**if( (arr1[i]%2) !=0)**

**{**

**cout<<"\n There is no euler circuit exist \n";**

**flag =1;**

**c++;**

**}**

**}**

**if(flag ==0)**

**cout<<"\n There is euler circuit \n ";**

**if(c==2)**

**cout<<"\n There is a euler path \n ";**

**else**

**cout<<"\n There is no euler path \n";**

**return 0;**

**}**

**Ouput:**

**Enter number of vertices for a adjancency matrix**

**4**

**How many edge from a to a - 0**

**How many edge from a to b - 1**

**How many edge from a to c - 1**

**How many edge from a to d - 0**

**How many edge from b to a - 1**

**How many edge from b to b - 0**

**How many edge from b to c - 1**

**How many edge from b to d - 0**

**How many edge from c to a - 1**

**How many edge from c to b - 1**

**How many edge from c to c - 0**

**How many edge from c to d - 0**

**How many edge from d to a - 1**

**How many edge from d to b - 1**

**How many edge from d to c - 0**

**How many edge from d to d - 0**

**THE ADJANCY MATRIX :**

**0 1 1 0**

**1 0 1 0**

**1 1 0 0**

**1 1 0 0**

**THE DEGREE OF a -- 2**

**THE DEGREE OF b -- 2**

**THE DEGREE OF c -- 2**

**THE DEGREE OF d -- 2**

**There is euler circuit**

**There is no euler path**

***Q20.Given a full m-ary with I internal vertices, write a program to find the number of leaf nodes.***

**Solution:**

**#include<iostream>**

**using namespace std;**

**int main()**

**{**

**int m,l,i;**

**cout<<"enter the degree of tree:";**

**cin>>m;**

**cout<<"enter the value of internal vertices:";**

**cin>>i;**

**l=i\*(m-1)+1;**

**cout<<"the number of leaves:"<<l<<endl;**

**return 0;**

**}**

**Output:**

**enter the degree of tree :4**

**enter the value of internal vertices:5**

**the number of leaves:16**

**Name: Neha kumari**

**Roll no: 2020331**