MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY & MANAGEMENT
(Approved by AICTE-New Delhi, Accredited by NAAC with 'A' & Affliated to JNTU, Hyderabad)
Recognised Under Section 2(f) & 12(B) of The UGC act, 1956
Dundigal, Quthbullapur (M), Hyderabad-500043.

Design Analysis of Algorithms Lab using Java LabManual

SubjectCode:2040576

Regulation: MLRS R20
Class:B.Tech II-II Semester

Prepared

by

SivaRamakrishna(Asst.Prof,CSE)
K.Suresh (Asst.Prof,CSE)
T.S.Srinivas (Asst.Prof,CSE)

DEPARTMENT
OF
COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that this manual is a bonafide record of practical work in the **Design Analysis and Algorithms** in **First Semester of II year B.Tech (CSE) programme** during the academic year **2021-22**. This book is prepared by **Mr.Sivaramakrishna (Asst.Professor), Mr.K.Suresh (Asst.Professor), Mr.T.S.Srinivas (Asst.Professor)** Department of Computer Science and Engineering.

INDEX

PREFACE

This book “Design analysis and Algorithms” lab manual is intended to teach the design and analysis of basic data structures and their implementation in an object-oriented language. Readers of this book need only be familiar with the basic syntax of Java and similar languages. The “DAA Concepts” is increasingly becoming the default choice of the IT industry especially industries involved in software development at system level. Therefore, for proper development of “Object Oriented Programming ” skills among the students this practical manual has been prepared. The manual contains the exercise programs and their solution for easy & quick understanding of the students. We hope that this practical manual will be helpful for students of Computer Science & Engineering for understanding the subject from the point of view of applied aspects. There is always scope for improvement in the manual. We would appreciate to receive valuable suggestions from readers and users for future use.

By

Sivaramakrishna
K.Suresh,
T.S.Srinivas.

ACKNOWLEDGEMENT

It was really a good experience, working with “**Design Analysis and Algorithms lab”**. First we would like to thank Mr.K.Abdul Basith, Assoc.Professor, HOD of Department of Computer Science and Engineering, Marri Laxman Reddy Institute of technology & Management for his concern and giving the technical support in preparing the document.

We are deeply indebted and gratefully acknowledge the constant support and valuable patronage of Dr.R.Kotaih, Director, Marri Laxman Reddy Institute of technology & Management for giving us this wonderful opportunity for preparing the ***Design Analysis and Algorithms*** laboratory manual.

We express our hearty thanks to Dr.K.Venkateswara Reddy, Principal, Marri Laxman Reddy Institute of technology & Management, for timely corrections and scholarly guidance.

At last, but not the least I would like to thanks the entire CSE Department faculties those who had inspired and helped us to achieve our goal.

By

Sivaramakrishna
K.Suresh,
T.S.Srinivas.

GENERAL INSTRUCTIONS

1. Students are instructed to come to Design Analysis and Algorithms laboratory on time. Late comers are not entertained in the lab.

2. Students should be punctual to the lab. If not, the conducted experiments will not be repeated.

3. Students are expected to come prepared at home with the experiments which are going to be performed.

4. Students are instructed to display their identity cards before entering into the lab.

5. Students are instructed not to bring mobile phones to the lab.

6. Any damage/loss of system parts like keyboard, mouse during the lab session, it is student's responsibility and penalty or fine will be collected from the student.

7. Students should update the records and lab observation books session wise. Before leaving the lab the student should get his lab observation book signed by the faculty.

8. Students should submit the lab records by the next lab to the concerned faculty members in the staffroom for their correction and return.

9. Students should not move around the lab during the lab session.

10. If any emergency arises, the student should take the permission from faculty member concerned in written format.

11. The faculty members may suspend any student from the lab session on disciplinary grounds.

12. Never copy the output from other students. Write down your own outputs.

INSTITUTION VISION AND MISSION

VISION

To establish as an ideal academic institutions in the service of the nation, the world and the humanity by graduating talented engineers to be ethically strong, globally competent by conducting high quality research, developing breakthrough technologies, and disseminating and preserving technical knowledge.

MISSION

To fulfill the promised vision through the following strategic characteristics and aspirations:

- Contemporary and rigorous educational experiences that develop the engineers and managers.
- An atmosphere that facilitates personal commitment to the educational success of students in an environment that values diversity and community.
- Undergraduate programs that integrate global awareness, communication skills and team building.
- Education and Training that prepares students for interdisciplinary engineering research and advanced problem solving abilities.

DEPARTMENT VISION, MISSION , PROGRAMME EDUCATIONAL OBJECTIVES AND SPECIFIC OUTCOMES

VISION

The Computer science Engineering department strives to impart quality education by extracting the innovative skills of students and to face the challenges in latest technological advancements and to serve the society.

MISSION

Provide quality education and to motivate students towards professionalism

Address the advanced technologies in research and industrial issues

PROGRAMME EDUCATIONAL OBJECTIVES

The Programme Educational Objectives (PEOs) that are formulated for the computer science engineering programme are listed below;

PEO-I solving computer science engineering problems in different circumstances PEO-II Pursue higher education and research for professional development.

PEO-III Inculcate qualities of leadership for technology innovation and entrepreneurship.

PROGRAM SPECIFIC OUUTCOMES

PSO1. UNDERSTANDING: Graduates will have an ability to describe, analyze, and solve problems using mathematics and systematic problem-solving techniques.

PSO2. ANALYTICAL SKILLS: Graduates will have an ability to design a system, component, or process to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability.

PSO3. BROADNESS: Graduates will have a broad education necessary to understand the impact of engineering solutions in a global, economic, and societal context.

PROGRAMME OUT COMES

a) An ability to apply knowledge of mathematics, science, and engineering

Graduates should transform knowledge of mathematics, Physics, chemistry, Engineering Mechanics, probability and statistics, and engineering drawing in solving a wide range of computer science engineering problems.

b) An ability to design, implement, evaluate a system and conduct experiments, as well as to analyze and interpret data

Graduates should show that they can make decisions regarding type, and number of data points to be collected, duration of the experiment to obtain intended results, and demonstrate an understanding of accuracy and precision of data

(c) **An ability to design, implement and evaluate a system, or process to meet desired needs** Graduates should be able to: identify the project goal; define the project; search for alternative possibilities; choose the best of the possible solutions; create a design drawing, design plan, or computer simulation; evaluate the design; and justify the final design in written and oral forms.

d) An ability to function effectively on multi-disciplinary teams

Graduates should show that they can participate effectively as team members with people who bring different skills, expertise, and perspectives to a project; and with people from different sub-disciplines of computer science engineering and interdisciplinary groups.

e) An ability to identify, formulate, analyse and solve engineering problems

Graduates should be able to describe the important components of a given problem, apply mathematical, engineering principles and to find the unknowns and arrive at appropriate and effective solutions.

f) **An understanding of professional, ethical, legal, security and social responsibilities** Graduates should be familiar with the applicable professional code of conduct for engineers. They should be able to apply the codes, where appropriate, to particular cases in which ethical issues arise. Graduates should also understand the importance of professionalism.

g) An ability to communicate effectively both in writing and orally

Computer science engineering graduates should have the ability to speak and write effectively in various domains like laboratory reports, technical reports, technical presentations, project reports etc.

h) The broad education necessary to analyse the impact of engineering solutions on a global and societal context

Graduates should get exposed to the interactions among science, technology, and social values, understand the influence of science and technology on computer scienceizations and how science and technology have been addressed for the betterment of humankind.

i) Recognition of the need for, and an ability to engage in continuing professional development and life-long learning

Graduates should show that they appreciate the need for further education and self improvement, understand the value of professional licensure the necessity of continuing professional developments, and the value of membership in appropriate professional organizations.

j) Knowledge of contemporary issues

Graduates should have knowledge and understand selected contemporary technical and social issues relevant to their field of study.

k) An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice

Graduates should have ability to use practical methods readily and effectively in the performance of engineering analysis and design. Graduates should be able to select and use modern engineering tools used by practicing engineers, including computer software such as computer aided drawing (CAD)

l) An ability to apply design and development principles in the construction of software and hardware systems of varying complexity

Computer science Graduates should have ability to design and develop principles involved in construction of different structures like buildings, shopping complexes, roads, water structures and to analyse the stability of structures using different softwares like stadpro. Studs etc.

COURSE STRUCTURE, OBJECTIVES & OUTCOMES

COURSE STRUCTURE

Environmental engineering lab will have a continuous evaluation during 7th semester for 25 sessional marks and 50 end semester examination marks.

Out of the 25 marks for internal evaluation, day-to-day work in the laboratory shall be evaluated for 15 marks and internal practical examination shall be evaluated for 10 marks conducted by the laboratory teacher concerned.

The end semester examination shall be conducted with an external examiner and internal examiner. The external examiner shall be appointed by the principal / Chief Controller of examinations

COURSE OBJECTIVE

- To write programs in java to solve problems using divide and conquer strategy.
- To write programs in java to solve problems using backtracking strategy.
- To write programs in java to solve problems using greedy and dynamic programming techniques.

COURSE OUTCOME

Ability to write programs in java to solve problems using algorithm design techniques such as Divide and Conquer, Greedy, Dynamic programming, and Backtracking.

III. Course files – Soft copies

HODs and faculty members are directed to submit the soft copies of the course files in the director’s office/L.M.S at the earliest, other wise it will be treated as not submitted

Exp:1. a

AIM. Use Eclipse or Net bean platform and acquaint with the various menus. Create a test project, add a test class, and run it. See how you can use auto suggestions, auto fill. Try code formatter and code refactoring like renaming variables, methods, and classes. Try debug step by step with a small program of about 10 to 15 lines which contains at least one if else condition and a for loop.

Algorithm:

Step 1: Start.

Step 2: Read the no from the keyboard.

Step 3: If the no is divisible by 2 Then even no .

Step 4: If the no is not divisible by 2 Then not even no

Step 5: Stop.

Program:

```
import java.util.Scanner;
class Sample
{
public static void main(String[] args)
{
int n;
Scanner s=new Scanner(System.in);
System.out.println("Enter n value:");
n=s.nextInt();
for(int i=0;i<=n;i++)
{
if(i%2==0)
{
System.out.println(i+" is an even number");
}
else
{
System.out.println(i+" is an odd number");
}
}
}
}
```

Out put:

```
Enter a number 2
Even no
```

Exp:1. b

AIM. Write a java program that prints all real solutions to the quadratic equation ax2 +bx+c=0. Read in a, b, c and use the quadratic formula.

Algorithm:

```
Step 1: Start.
Step 2: Read the a,b,c from the keyboard.
Step 3: b is greater than 0 then roots are real
Step 4:b is lessr than 0 then roots are complex
Step 5: Stop.
```

Program:

```
import java.util.*;
class QuadraticRoots
{
        public static void main(String args[])
        {
                int a,b,c,d;
                Scanner s=new Scanner(System.in);
                System.out.println("Enter the values of a ,b ,c : ");
                a=s.nextInt();
                b=s.nextInt();
                c=s.nextInt();
                d=(b*b)-(4*a*c);
                if(d==0)
                {
                        System.out.println("Roots are real and Equal");
                        float r1=(float)(-b+Math.sqrt(d))/(2*a);
                        float r2=(float)(-b-Math.sqrt(d))/(2*a);
                        System.out.println("Roots are :"+r1+" ,"+r2);
                }
                else if(d>0)
                {
                        System.out.println("Roots are real and UnEqual");
                        float r1=(float)(-b+Math.sqrt(d))/(2*a);
                        float r2=(float)(-b-Math.sqrt(d))/(2*a);
                        System.out.println("Roots are :"+r1+" ,"+r2);
                }
                else
                     System.out.println("Roots are imaginary");

                }
```

Output:

```
Enter a,b,c values
```

```
4
5
2
Roots are =0.5,1
```

Exp:1. c

AIM. Write a java program to implement Fibonacci series.

Algorithm:

```
Step 1: Start.
Step 2: Read the n value  from the keyboard.
Step 3:print the fibanaci series
Step 4: Stop.
```

Program:

```
import java.util.*;
class Fibonacci
{
	public static void main(String args[])
	{
		int n,n1=0,n2=1,n3,i;
		Scanner s=new Scanner(System.in);
		System.out.print("Enter number of terms:");
		n=s.nextInt();
		System.out.print("Fibonacii series:");
		System.out.print(n1+" "+n2);
		for(i=2;i<n;++i)
		{
			n3=n1+n2;
			System.out.print(" "+n3);
			n1=n2;
			n2=n3;
		}
	}
}
```

Output:

```
Enter n
3
0,1,2
```

VIVA QUESTIONS:

Explain public static void main(String args[]) in Java

Why Java is platform independent?

Why Java is not 100% Object-oriented?

What are wrapper classes in Java?

What are constructors in Java?

EXP 2a.

Algorithm:

```
Step 1: Start.
Step 2: Read the a,b,c from the keyboard.
Step 5: Stop.
```

AIM:Write a java program to implement method overloading and constructors overloading.

```
class Sample
{
Sample()
{
System.out.println("No Parameter Constructor");
}
Sample( int a)
{
System.out.println("One Parameter Constructor");
}
Sample( int a,int b)
{
System.out.println("Two Parameter Constructor");
}
public static void main(String[] args)
{
Sample x = new Sample();
Sample y = new Sample(10);
Sample z = new Sample(10,20);
}
}
```

Output:

```
10,20
```

C.Aim: Write a java program to implement method overriding.

Algorithm:

```
Step 1: Start.
Step 2: Read the bank interest
Step 3:print the interest
Step 5: Stop.
```

Program:

```
class Bank
{
int getRateOfInterest()
{
return 0;
}
}
class Bank1 extends Bank
```

```
{
int getRateOfInterest()
{
return 10;
}
}
class Bank2 extends Bank
{
int getRateOfInterest()
{
return 9;
}
}
class Bank3 extends Bank
{
int getRateOfInterest()
{
return 8;
}
}
class Sample
{
public static void main(String args[])
{
Bank1 b1=new Bank1();
Bank2 b2=new Bank2();
Bank3 b3=new Bank3();
System.out.println("Bank1 Rate of Interest: "+b1.getRateOfInterest());
System.out.println("Bank2 Rate of Interest: "+b2.getRateOfInterest());
System.out.println("Bank3 Rate of Interest: "+b3.getRateOfInterest());
}
}
```

Output:

```
Rate of interest 15
```

c) Aim: Write a Java program to create an abstract class named Shape that contains two integers and an empty method named print Area (). Provide three classes named Rectangle, Triangle, and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape.

Algorithm:

Step 1: Start.

Step 2: Read the values from the keyboard.

Step 3:print area

Step 4: Stop.

Program:

```
abstract class Shape
```

```
{
int height;
int width;
int radius;
abstract int printArea();
}
class Rectangle extends Shape
{
Rectangle(int width,int height)
{
this.width=width;
this.height=height;
}
int printArea()
{
System.out.println("Inside Area for Rectangle.");
return height * width;
}
}
class Triangle extends Shape
{
Triangle(int width, int height)
{
this.width=width;
this.height=height;
}
int printArea()
{
System.out.println("Inside Area for Triangle.");
return height * width / 2;
}
}
class Circle extends Shape
{
Circle(int radius)
{
this.radius=radius;
}int printArea()
{
System.out.println("Inside Area for Circle.");
int area=(int)(3.14*radius*radius);
return(area);
}
}
class AbstractDemo
{
public static void main(String args[])
{
Rectangle r = new Rectangle(10, 5);
Triangle t = new Triangle(10, 8);
Circle c=new Circle(10);
System.out.println("Area is " +r.printArea());
System.out.println("Area is " + t.printArea());
System.out.println("Area is " +c.printArea());
}
}
```

Output:

```
Area:30
Area:40
Area:2270
```

Viva questions

What do you understand by an instance variable and a local variable?

What do you mean by data encapsulation?

How is an infinite loop declared in Java?

When can you use super keyword?

Can the static methods be overloaded?

3.a. Aim:Write a java program to check whether a given string is palindrome.

Algorithm:

```
Step 1: Start.
Step 2: Read the values from the keyboard.
Step 3:print area
Step 4: Stop.
```

Program:

```
import java.util.*;
class Palindrome
{
        public static void main(String args[])
        {
                Scanner s=new Scanner(System.in);
                System.out.print("Enter the String:");
                String str=s.nextLine();
                int size=str.length();
                char ch[]=new char[size];
                for(int i=size-1,j=0;i>=0;i--,j++)
                {
                        ch[j]=str.charAt(i);
                }
                String rev=new String(ch);
                if(str.equals(rev))
                {
                    System.out.println(str+" "+"is a Palindrome");
                }
                else
                {
                    System.out.println(str+" "+"is not a Palindrome");
                }
        }
}
```

Output: enter string
Aba
palindrome

3b.

Aim:Write a Java program to create an abstract class named Shape that contains two integers and an empty method named print Area (). Provide three classes namedRectangle, Triangle, and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape

Algorithm:

```
Step 1: Start.
Step 2: Read the string from the keyboard.
Step 3: string and reverse of the string is same then write palindrome
Step 4:string and reverse of the string is  not same then write not  palindrome
Step 5: Stop
```

Program:

```
abstract class Shape
{
        int height;
        int width;
        int radius;
        abstract int printArea();
}
class Rectangle extends Shape
{
        Rectangle(int width,int height)
        {
                this.width=width;
                this.height=height;
        }
        int printArea()
        {
                System.out.println("Inside Area for Rectangle.");
                return height * width;
        }
}
class Triangle extends Shape
{
        Triangle(int width, int height)
        {
                this.width=width;
                this.height=height;
        }
        int printArea()
        {
                System.out.println("Inside Area for Triangle.");
                return height * width / 2;
        }
}
class Circle extends Shape
{
        Circle(int radius)
        {
                this.radius=radius;
        }
        int printArea()
        {
                System.out.println("Inside Area for Circle.");
                int area=(int)(3.14*radius*radius);
                return(area);
        }
}
class AbstractDemo
{
        public static void main(String args[])
        {
                Rectangle r = new Rectangle(10, 5);
                Triangle t = new Triangle(10, 8);
                Circle c=new Circle(10);
                System.out.println("Area is " +r.printArea());
                System.out.println("Area is " + t.printArea());
                System.out.println("Area is " +c.printArea());
        }
}
```

```
Area:30
Area:40
Area:2270
```

viva questions.
What is singleton class in Java and how can we make a class singleton?
What is the difference between Array list and vector in Java?
What is the difference between equals() and == in Java?
What are the differences between Heap and Stack Memory in Java?
Why is Java a platform independent language?
Why is Java not a pure object oriented language?

4.
Aim: Write a program to implement Knapsack problem using greedy method.

Algorithm:

Step 1: Start.

Step 2: Read theprofit and weight vlues from keyboard

Step 3:calculate max pi/wi values

Step 4:print the maximum profit

Step 5: Stop.

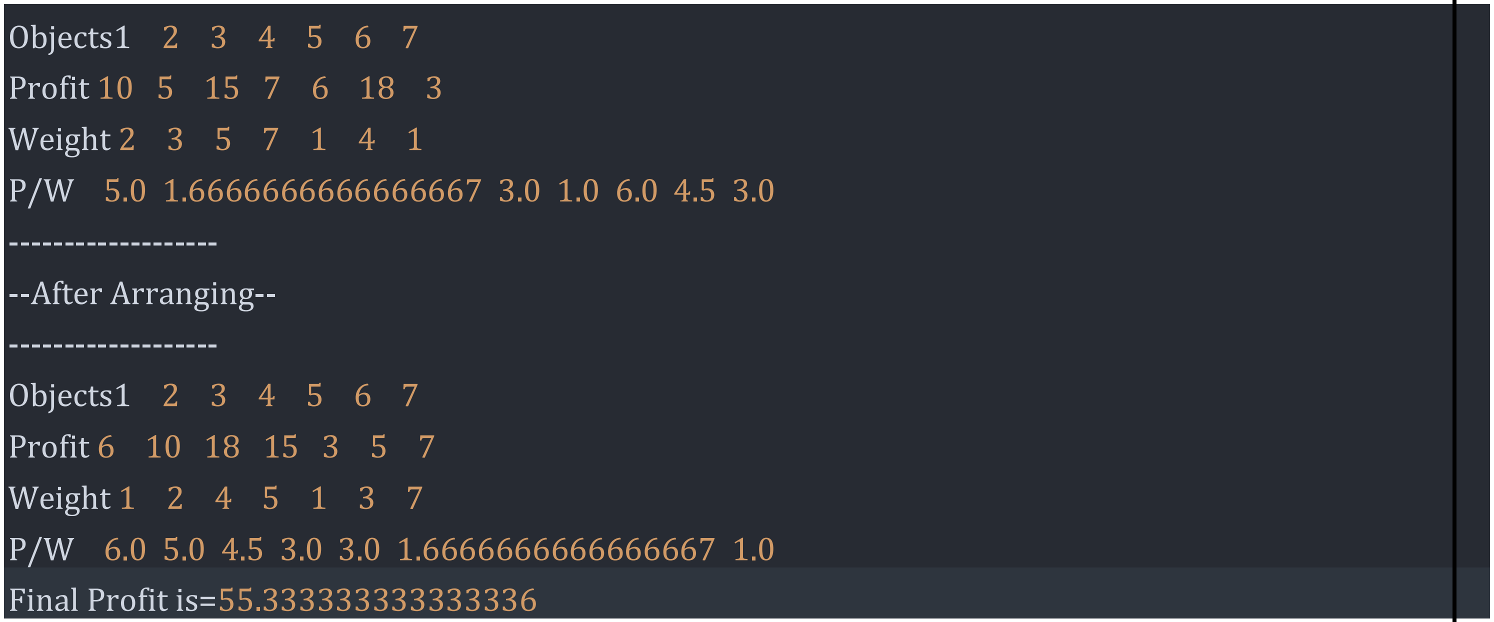
Program:

```
import java.util.Scanner;
class Knapsack
{
public static void main(String[] args)
{
Scanner sc=new Scanner(System.in);
int object,m;
System.out.println("Enter the Total Objects");
object=sc.nextInt();
int weight[]=new int[object];
int profit[]=new int[object];
for(int i=0;i<object;i++)
{
System.out.println("Enter the Profit");
profit[i]=sc.nextInt();
System.out.println("Enter the weight");
weight[i]=sc.nextInt();
}
System.out.println("Enter the Knapsack capacity");
m=sc.nextInt();
double p_w[]=new double[object];
for(int i=0;i<object;i++)
{
p_w[i]=(double)profit[i]/(double)weight[i];
}
```

```
System.out.println("");
System.out.println("-------------------");
System.out.println("-----Data-Set------");
System.out.print("-------------------");
System.out.println("");
System.out.print("Objects");
for(int i=1;i<=object;i++)
{
System.out.print(i+"    ");
}
System.out.println();
System.out.print("Profit ");
for(int i=0;i<object;i++)
{
System.out.print(profit[i]+"    ");
}
System.out.println();
System.out.print("Weight ");
for(int i=0;i<object;i++)
{
System.out.print(weight[i]+"    ");
}
System.out.println();
System.out.print("P/W    ");
for(int i=0;i<object;i++)
{
System.out.print(p_w[i]+"  ");
}
for(int i=0;i<object-1;i++)
{
for(int j=i+1;j<object;j++)
{
if(p_w[i]<p_w[j])
{
double temp=p_w[j];
p_w[j]=p_w[i];
p_w[i]=temp;
int temp1=profit[j];
profit[j]=profit[i];
profit[i]=temp1;
int temp2=weight[j];
weight[j]=weight[i];
weight[i]=temp2;
}
}
}
System.out.println("");
System.out.println("-------------------");
System.out.println("--After Arranging--");
System.out.print("-------------------");
System.out.println("");
System.out.print("Objects");
for(int i=1;i<=object;i++)
{
System.out.print(i+"    ");
}
System.out.println();
System.out.print("Profit ");
for(int i=0;i<object;i++)
{
System.out.print(profit[i]+"    ");
}
System.out.println();
System.out.print("Weight ");
for(int i=0;i<object;i++)
{
System.out.print(weight[i]+"    ");
}
System.out.println();
System.out.print("P/W    ");
for(int i=0;i<object;i++)
{
System.out.print(p_w[i]+"  ");
}
int k=0;
double sum=0;
System.out.println();
while(m>0)
{
if(weight[k]<m)
{
sum+=1*profit[k];
m=m-weight[k];
}
else
{
double x4=m*profit[k];
double x5=weight[k];
double x6=x4/x5;
sum=sum+x6;
m=0;
}
k++;
}
System.out.println("Final Profit is="+sum);
}
}
```

Output:

```
Enter the Total Objects
7
Enter the Profit
10
Enter the weight
2
Enter the Profit
5
Enter the weight
3
Enter the Profit
15
Enter the weight
5
Enter the Profit
7
Enter the weight
7
Enter the Profit
6
Enter the weight
1
Enter the Profit
18
Enter the weight
4
Enter the Profit
3
Enter the weight
1
Enter the Knapsack capacity
15
-------------------
-----Data-Set------
-------------------
```



```
Objects1   2   3   4   5   6   7
Profit 10  5   15  7   6   18  3
Weight 2   3   5   7   1   4   1
P/W   5.0  1.6666666666666667  3.0  1.0  6.0  4.5  3.0
------------------
--After Arranging--
------------------
Objects1   2   3   4   5   6   7
Profit 6   10  18  15  3   5   7
Weight 1   2   4   5   1   3   7
P/W   6.0  5.0  4.5  3.0  3.0  1.6666666666666667  1.0
Final Profit is=55.333333333333336
```

Viva questions:
Define the term algorithm and state the criteria the algorithm should satisfy.
Write order of an algorithm and the need to analyze the algorithm.
Define asymptotic notations: big ‘Oh’, omega and theta?
List the two different types of recurrence
State the best case and worst case analysis for linear search

5. Aim:Write a program to implement Prim’s minimum cost spanning tree using Greedy Method
Algorithm:
Step 1: Start.

Step 2: Read the no of vertices,edges along with cost from the keyboard.

Step 3:find out minimum cost spanning tree using prims algorithm

Step 4:print minimum cost

Step 5: Stop.

Program:
```
import java.util.Scanner;
public class PRIM
{
public static void main(String[] args)
{
int cost[][]=new int[10][10];
int i, j, mincost = 0;
Scanner in = new Scanner(System.in);
System.out.println("********* PRIMS ALGORITHM *********");
System.out.println("Enter the number of nodes");
int n = in.nextInt();
System.out.println("Enter the cost matrix");
for(i=1; i<=n; i++){
```

```
for(j=1; j<=n; j++){
cost[i][j] = in.nextInt();
}
}
System.out.println("The entered cost matrix is");
for(i=1; i<=n; i++)
{
for(j=1; j<=n; j++)
{
System.out.print(cost[i][j]+"\t");
}
System.out.println();
}
System.out.println("Minimum Spanning Tree Edges and costs are");
mincost=prims(cost,n,mincost);
System.out.print("The minimum spanning tree cost is:");
System.out.print(+mincost);
}
static int prims(int cost[][],int n,int mincost)
{
int nearV[]=new int[10],t[][]=new int[10][3],u = 0,i,j,k;
for(i=2; i<=n; i++)
nearV[i]=1;
nearV[1]=0;
for(i=1; i<n; i++)
{
int min=999;
for(j=1;j<=n;j++)
{
if(nearV[j]!=0 && cost[j][nearV[j]]<min)
{
min=cost[j][nearV[j]];
u=j;
}
}
t[i][1] = u;
t[i][2] = nearV[u];
mincost += min;
nearV[u] = 0;
for(k=1; k<=n; k++){
if(nearV[k] != 0 && cost[k][nearV[k]] > cost[k][u])
nearV[k] = u;
}
System.out.print(i+") Minimum edge is ("+t[i][1]);
System.out.println(","+t[i][2]+") and its cost is :"+min);
}
return mincost;
}
}
```

Output:

```
********* PRIMS ALGORITHM *********
Enter the number of nodes
3
Enter the cost matrix
0
2
999
2
0
1
999
1
0
The entered cost matrix is
0 2 999
2 0 1
999 1 0
Minimum Spanning Tree Edges and costs are
1) Minimum edge is (2,1) and its cost is :2
2) Minimum edge is (3,2) and its cost is :1
The minimum spanning tree cost is:3
```

Viva questions:If f(n)=5n2 + 6n + 4, then prove that f(n) is O(n2)
Give the recurrence equation for the worst case behavior of merge sort.
Compute the average case time complexity of quick sort
Define algorithm correctness
Describe best case, average case and worst case efficiency of an algorithm?
Explain the term amortized efficiency
Define order of growth

6 Aim:.Write a program to implement Kruskal’s minimum cost spanning tree using Greedy Method
Algorithm:
Step 1: Start.

Step 2: Read the no of vertices,edges along with cost from the keyboard.

Step 3:find out minimum cost spanning tree using kruskals algorithm

Step 4:print minimum cost

Step 5: Stop.

Program:

```
import java.util.Scanner;
public class KRUSKAL
{
public static void main(String[] args)
{
int cost[][]=new int[10][10];
int i, j,mincost=0;
Scanner in = new Scanner(System.in);
```

```
System.out.println("********* KRUSKAL'S ALGORITHM *******");
System.out.println("Enter the number of nodes: ");
int n = in.nextInt();
System.out.println("Enter the cost matrix");
for(i=1;i<=n;i++){
for(j=1;j<=n;j++){
cost[i][j] = in.nextInt();
}
}
System.out.println("The entered cost matrix is");
for(i=1;i<=n;i++){
for(j=1;j<=n;j++){
System.out.print(cost[i][j]+"\t");
}
System.out.println();
}
mincost=kruskals(n,mincost,cost);
System.out.println("The minimum spanning tree cost is:");
System.out.println(mincost);
}
static int kruskals(int n,int mincost,int cost[][] )
{
int ne = 1,a=0,u=0,b=0,v=0,min;
int parent[]=new int[10];
while(ne < n){
min=999;
for(int i=1; i<=n; i++)
{
for(int j=1; j<=n; j++)
{
if(cost[i][j] < min){
min = cost[i][j];
a=u=i;
b=v=j;
}
}
}
while(parent[u]>0)
u = parent[u];
while(parent[v]>0)
v = parent[v];
if(u != v)
{
System.out.print((ne++)+">minimum edge is :");
System.out.println("("+a+","+b+") and its cost is:"+min);
mincost += min;
parent[v] = u;
}
cost[a][b] = cost[b][a] = 999;
}
return mincost;
}
}
```

Output:

```
********* KRUSKAL'S ALGORITHM *******
Enter the number of nodes:
3
Enter the cost matrix
0
2
6
2
0
2
6
2
0
The entered cost matrix is
0 2 6
2 0 2
6 2 0
1>minimum edge is :(1,2) and its cost is:2
2>minimum edge is :(2,3) and its cost is:2
The minimum spanning tree cost is:
4
```

Viva questions:

How do you measure the algorithm running time?

Describe the role of space complexity and time complexity of a program ?

Explain algorithm design technique?

Use step count method and analyze the time complexity when two n×n matrices are added

What is meant by divide and conquer? Give the recurrence relation for divide and conquer.

7. Aim:Write a program to implement Job sequencing with deadlines using Greedy Method

Algorithm:

Step 1: Start.

Step 2: Read the no of jobs,profits,weights from the keyboard.

Step 3:find out sequence of jobs according to its deadline

Step 4:print maximum profit

Step 5: Stop.

Program:

```
import java.util.*;
class job
{
int p; //............for profit of a job
int d; //............for deadline of a job
int v; //............for checking if that job has been selected
```

```
job()
{
p=0;
d=0;
v=0;
}
job(int x,int y,int z) // parameterised constructor
{
p=x;
d=y;
v=z;
}
}
class js
{
static int n;
static int out(job jb[],int x)
{
for(int i=0;i<n;++i)
if(jb[i].p==x)
return i;
return 0;
}
public static void main(String args[])
{
Scanner scr=new Scanner(System.in);
System.out.println("Enter the number of jobs");
n=scr.nextInt();
int max=0; // this is to find the maximum deadline
job jb[]=new job[n];
for(int i=0;i<n;++i)
{
System.out.println("Enter profit and deadline(p d)");
int p=scr.nextInt();
int d=scr.nextInt();
if(max<d)
max=d; // assign maximum value of deadline to "max" variable
jb[i]=new job(p,d,0); //zero as third parameter to mark that initially it is unvisited
}
//accepted jobs from user
for(int i=0;i<=n-2;++i)
{
for(int j=i;j<=n-1;++j)
{
if(jb[i].d>jb[j].d)
{
job temp=jb[i];
jb[i]=jb[j];
jb[j]=temp;
}
}
}
// sorting process ends
System.out.println("The jobs are as follows ");
for(int i=0;i<n;++i)
System.out.println("Job "+i+" Profit = "+jb[i].p+" Deadline = "+jb[i].d);
// jobs displayed to the user
int count;
int hold[]=new int[max];
for(int i=0;i<max;++i)
hold[i]=0;
for(int i=0;i<n;++i)
{
count=0;
for(int j=0;j<n;++j)
{
if(count<jb[j].d && jb[j].v==0 && count<max && jb[j].p>hold[count])
{
int ch=0;
if(hold[count]!=0)
{
ch=out(jb,hold[count]);
jb[ch].v=0;
}
hold[count]=jb[j].p;
jb[j].v=1;
++count;
}
}
}
int profit=0;
for(int i=0;i<max;++i)
profit+=hold[i];
System.out.println("The maximum profit is "+profit);
}
}
```

OUTPUT

```
Enter the number of jobs
4
Enter profit and deadline(p d)
70 2
Enter profit and deadline(p d)
12 1
Enter profit and deadline(p d)
```

```
18 2
Enter profit and deadline(p d)
35 1
The jobs are as follows
Job 0 Profit = 12 Deadline = 1
Job 1 Profit = 35 Deadline = 1
Job 2 Profit = 18 Deadline = 2
Job 3 Profit = 70 Deadline = 2
The maximum profit is 105
```

Viva questions:
Define Control Abstraction and write the computing time of divide and conquer.
List out any two drawbacks of binary search algorithm.
What are the drawbacks of Merge Sort algorithm.
Discuss about union operation on sets
Describe find operation on sets

8.Aim:Write a program to implement Single source shortest path problem using Greedy Method

Algorithm:
Step 1: Start.

Step 2: Read the no of vertices,edges along with cost from the keyboard.

Step 3:find out shortest distance from source to destination

Step 4:print minimum cost

Step 5: Stop.

Program:

```
import java.util.Scanner;
public class Dijkstras {
public static void main(String[] args)
{
int i, j;
int dist[]=new int[10], visited[]=new int[10];
int cost[][]=new int[10][10], path[]=new int[10];
Scanner in = new Scanner(System.in);
System.out.println("**** DIJKSTRA'S ALGORITHM ******");
System.out.println("Enter the number of nodes: ");
int n= in.nextInt();
System.out.println("Enter the cost matrix");
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
cost[i][j] = in.nextInt();
System.out.println("The entered cost matrix is");
for(i=1;i<=n;i++)
{
for(j=1;j<=n;j++)
{
```

```
System.out.print(cost[i][j]+"\t");
}
System.out.println();
}
System.out.println("Enter the source vertex: ");
int sv = in.nextInt();
dij(cost,dist,sv,n,path,visited);
printpath(sv,n,dist,path,visited );
System.out.println("\n********* *************** *********");
}
static void dij(int cost[][],int dist[],int sv,int n,int path[],int visited[])
{
int count = 2,min,v=0;
for(int i=1; i<=n; i++)
{
visited[i]=0;
dist[i] = cost[sv][i];
if(cost[sv][i] == 999)
path[i] = 0;
else
path[i] = sv;
}
visited[sv]=1;
while(count<=n)
{
min = 999;
for(int w=1; w<=n; w++)
if((dist[w]< min) && (visited[w]==0))
{
min = dist[w];
v = w;
}
visited[v] = 1;
count++;
for(int w=1; w<=n; w++)
{
if((dist[w]) >(dist[v] + cost[v][w]))
{
dist[w] = dist[v] + cost[v][w];
path[w] = v;
}
}
}
}
static void printpath(int sv,int n,int dist[],int path[],int visited[])
{
for(int w=1; w<=n; w++)
{
if(visited[w] == 1 && w != sv)
```

```
{
System.out.println("The shortest distance between ");
System.out.println(sv+"-> ="+w+" is :"+ dist[w]);
int t=path[w];
System.out.println("The path is:");
System.out.print(" "+w);
while(t != sv)
{
System.out.print("<-->"+t);
t=path[t];
}
System.out.print("<-->"+sv);
}
}
}
}
```

OUTPUT

```
Enter the number of nodes:
4
Enter the cost matrix
0 5 0 999
5 0 1 999
999 1 999 2
999 999 5 0
The entered cost matrix is
0 5 0 999
5 0 1 999
999 1 999 2
999 999 5 0
Enter the source vertex:
1
The shortest distance between
1-> =2 is :1
The path is:
2<-->3<-->1The shortest distance between
1-> =3 is :0
The path is:
3<-->1The shortest distance between
1-> =4 is :2
The path is:
4<-->3<-->1
```

Viva questions:
Define spanning tree and minimal spanning tree
What do mean by depth first search
Define breadth first search
Differentiate Breadth first search and depth first search
Describe AND/OR graph

9.Aim:Write a program to implement All pairs shortest path using Dynamic Programming

Algorithm:
Step 1: Start.

Step 2: Read the no of vertices,edges along with cost from the keyboard.

Step 3:find out minimum cost among all existing vertices

Step 4:print minimum cost among all existing vertices

Step 5: Stop.

Program:

```
import java.util.Scanner;
class Floyd {
public static void main(String[] args)
{
int a[][]=new int[10][10];
int i, j;
Scanner in = new Scanner(System.in);
System.out.println("***********FLOYD'SALGORITHM**********");
System.out.println("Enter the number of vertices: ");
int n = in.nextInt();
System.out.println("Enter the adjacency matrix");
for (i=1;i<=n;i++)
for (j=1;j<=n;j++)
a[i][j] = in.nextInt();
System.out.println("Entered adjacency matrix is: ");
for(i=1;i<=n;i++)
{
for(j=1; j<=n; j++)
{
System.out.print(a[i][j]+"\t");
}
System.out.println();
}
floyd(a,n);
System.out.println("All pair shortest path matrix:");
for (i=1; i<=n; i++)
{
for (j=1; j<=n; j++)
System.out.print(a[i][j]+"\t");
System.out.println();
}
}
static void floyd(int a[][],int n)
{
for (int k=1; k<=n; k++)
{
for (int i=1; i<=n; i++)
for (int j=1; j<=n; j++)
a[i][j] = min(a[i][j], a[i][k] + a[k][j]);
}
}
static int min(int a,int b)
{
if(a>b)
return b;
else
return a;
}
}
```

Output:

```
***********FLOYD'SALGORITHM**********
Enter the number of vertices:
4
Enter the adjacency matrix
0
1
3
1
2
0
5
999
4
7
999
1
33
2
1
3
Entered adjacency matrix is:
0 1 3 1
2 0 5 999
4 7 999 1
33 2 1 3
All pair shortest path matrix:
0 1 2 1
2 0 4 3
4 3 2 1
4 2 1 2
```

Viva questions:

Explain game tree

Define an articulation point?

Define a connected and bi-connected component.

Write an algorithm for breadth first search . Give example

Explain depth first search algorithm with example

Discuss various tree traversal techniques with examples

10.Aim:Write a program to implement Optimal Binary Search Tree using Dynamic Programming

Algorithm:

Step 1: Start.

Step 2: Read the no of keywords along with their cost

Step 3:find out the values of root,weight,cost

Step 4:print OBST

Step 5: Stop.

Program:

```
import java.io.*;
import java.util.*;
class Optimal
{
public int p[];
public int q[];
public int a[];
public int w[][];
public int c[][];
public int r[][];
public int n;
int front,rear,queue[];
public Optimal(int SIZE)
{
p=new int[SIZE];
q= new int[SIZE];
a=new int[SIZE];
w=new int[SIZE][SIZE];
c=new int[SIZE][SIZE];
r=new int[SIZE][SIZE];
queue=new int[SIZE];
front=rear=-1;
}
/* This function returns a value in the range r[i][j-1] to r[i+1][j] SO that the cost c[i][k-1] + c[k][j] is
minimum */
public int Min_Value(int i, int j)
{
int m,k=0;
int minimum = 32000;
for (m=r[i][j-1] ; m<=r[i+1][j] ; m++)
{
if ((c[i][m-1]+c[m][j]) < minimum)
{
minimum = c[i][m-1] + c[m][j];
k = m;
}
}
return k;
}
/* This function builds the table from all the given probabilities It basically computes C,r,W values */
public void OBST()
{
int i, j, k, l, m;
for (i=0 ; i<n ; i++)
```

```
{
// Initialize
w[i][i] = q[i];
r[i][i] = c[i][i] = 0;
// Optimal trees with one node
w[i][i+1] = q[i] + q[i+1] + p[i+1];
r[i][i+1] = i+1;
c[i][i+1] = q[i] + q[i+1] + p[i+1];
}
w[n][n] = q[n];
r[n][n] = c[n][n] = 0;
// Find optimal trees with m nodes
for (m=2 ; m<=n ; m++)
{
for (i=0 ; i<=n-m ; i++)
{
j = i+m;
w[i][j] = w[i][j-1] + p[j] + q[j];
k = Min_Value(i,j);
c[i][j] = w[i][j] + c[i][k-1] + c[k][j];
r[i][j] = k;
}
}
}
/*This function builds the tree from the tables made by the OBST function */
public void build_tree()
{
int i, j, k;
System.out.print("The Optimal Binary Search Tree For The Given Nodes Is ....\n");
System.out.print("\n The Root of this OBST is :: "+r[0][n]);
System.out.print("\n The Cost Of this OBST is :: "+c[0][n]);
System.out.print("\n\n\tNODE\tLEFT CHILD\tRIGHT CHILD");
System.out.println("\n -------------------------------------------------------");
queue[++rear] = 0;
queue[++rear] = n;
while(front != rear)
{
i = queue[++front];
j = queue[++front];
k = r[i][j];
System.out.print("\n "+k);
if (r[i][k-1] != 0)
{
System.out.print(" "+r[i][k-1]);
queue[++rear] = i;
queue[++rear] = k-1;
}
else
System.out.print(" -");
if(r[k][j] != 0)
{
System.out.print(" "+r[k][j]);
queue[++rear] = k;
queue[++rear] = j;
}
else
System.out.print(" -");
}
System.out.println("\n");
}
}
/* This is the main function */
class OBSTDemo
{
public static void main (String[] args )throws IOException,NullPointerException
{
Optimal obj=new Optimal(10);
int i;
System.out.print("\n Optimal Binary Search Tree \n");
System.out.print("\n Enter the number of nodes ");
obj.n=getInt();
System.out.print("\n Enter the data as ....\n");
for (i=1;i<=obj.n;i++)
{
System.out.print("\n a["+i+"]");
obj.a[i]=getInt();
}
for (i=1 ; i<=obj.n ; i++)
{
System.out.println("p["+i+"]");
obj.p[i]=getInt();
}
for (i=0 ; i<=obj.n ; i++)
{
System.out.print("q["+i+"]");
obj.q[i]=getInt();
}
obj.OBST();
obj.build_tree();
}
public static String getString() throws IOException
{
InputStreamReader input = new InputStreamReader(System.in);
BufferedReader b = new BufferedReader(input);
String str = b.readLine();//reading the string from console
return str;
}
public static char getChar() throws IOException
{
String str = getString();
return str.charAt(0);//reading first char of console string
}
public static int getInt() throws IOException
{
String str = getString();
return Integer.parseInt(str);//converting console string to numeric value
}
}
```

OUTPUT:

```
Optimal Binary Search Tree
Enter the number of nodes 4
Enter the data as ....
```

```
a[1] 1
a[2] 2
a[3] 3
a[4] 4
p[1] 3
p[2] 3
p[3] 1
p[4] 1
q[0] 2
q[1] 3
q[2] 1
q[3] 1
q[4] 1
The Optimal Binary Search Tree For The Given Nodes Is ....
The Root of this OBST is :: 2
The Cost Of this OBST is :: 32
NODE LEFT CHILD RIGHT CHILD
-------------------------------------------------------
2 1 3
1 - -
3 - 4
4 - -
```

Viva questions:
List out any two drawbacks of binary search algorithm.
What are the drawbacks of Merge Sort algorithm.
Discuss about union operation on sets
Describe find operation on sets
Define spanning tree and minimal spanning tree

11aim:Write a program to implement 0/1 Knapsack problem using Dynamic Programming
Algorithm:
Step 1: Start.

Step 2: Read the no of items along with their profit and weight the keyboard.

Step 3:find out the total profit

Step 4:print maximum profit

Step 5: Stop.

Program:

```
import java.util.Scanner;
public class Knapsack6a
{
static final int MAX = 20; // max. no. of objects
static int w[]; // weights 0 to n-1
static int p[]; // profits 0 to n-1
static int n;
// no. of objects
static int M;
// capacity of Knapsack
static int V[][];
```

```
// DP solution process -table
static int Keep[][]; // to get objects in optimal solution
public static void main(String args[])
{
w = new int[MAX];
p = new int[MAX];
V = new int [MAX][MAX];
Keep = new int[MAX][MAX];
int optsoln;
ReadObjects();
for (int i = 0; i <= M; i++)
V[0][i] = 0;
for (int i = 0; i <= n; i++)
V[i][0] = 0;
optsoln = Knapsack();
System.out.println("Optimal solution = " + optsoln);
}
static int Knapsack()
{
int r; // remaining Knapsack capacity
for (int i = 1; i <= n; i++)
for (int j = 0; j <= M; j++)
if ((w[i] <= j) && (p[i] + V[i -1][j -w[i]] > V[i -1][j]))
{
V[i][j] = p[i] + V[i -1][j -w[i]];
Keep[i][j] = 1;
}
else
{
V[i][j] = V[i -1][j];
Keep[i][j] = 0;
}
// Find the objects included in the Knapsack
r = M;
System.out.println("Items = ");
for (int i = n; i > 0; i--) // start from Keep[n,M]
if (Keep[i][r] == 1)
{
System.out.println(i + " ");
r = r -w[i];
}
System.out.println();
return V[n][M];
}
static void ReadObjects()
{
Scanner scanner = new Scanner(System.in);
System.out.println("Knapsack Problem -Dynamic Programming Solution: ");
System.out.println("Enter the max capacity of knapsack: ");
M = scanner.nextInt();
System.out.println("Enter number of objects: ");
n = scanner.nextInt();
System.out.println("Enter Weights: ");
for (int i = 1; i <= n; i++)
w[i] = scanner.nextInt();
System.out.println("Enter Profits: ");
for (int i = 1; i <= n; i++)
p[i] = scanner.nextInt();
scanner.close();
}
}
```

Viva questions:
How do you measure the algorithm running time?
Describe the role of space complexity and time complexity of a program ?
Explain algorithm design technique?
Use step count method and analyze the time complexity when two n×n matrices are added
What is meant by divide and conquer? Give the recurrence relation for divide and conquer.

12 Aim:Write a program to implement n-Queen's problem using backtracking method.
Algorithm:
Step 1: Start.

Step 2: Read the no of queens

Step 3:find out solution for N-queens

Step 4:print solution

Step 5: Stop.

Program:

```
package backtracking;
public class NQueens2DArray {
public static void main(String[] args) {
placeQueens(4); // Lets take example of 4*4
}
private static void placeQueens(intgridSize){
//If Grid is 1*1 or 2*2 or 3*3 then solution is not possible as,
//In 1*1 or 2*2 grid, Queen placed in 1st row at any position will attack queen placed at all the
positions in row 2.
//In 3*3 grid, Queen placed in 1st row and 2nd row for all combinations position will attack
queen placed at all the positions in row 3.
if(gridSize<4){
System.out.println("No Solution available");
}else{
int[][] board = new int[gridSize][gridSize];
placeAllQueens(board, 0);
```

```
printBoard(board);
}
}
private static booleanplaceAllQueens(int board[][], int row){
if(row>=board.length){
return true;
}
booleanisAllQueensPlaced = false;
for (int j = 0; j <board.length; j++) {
if(isSafe(board, row, j)){
board[row][j] = 1;
isAllQueensPlaced = placeAllQueens(board, row+1);
}
if(isAllQueensPlaced){
break;
}else{
board[row][j] = 0;
}
}
return isAllQueensPlaced;
}
private static booleanisSafe(int board[][], int row, int col){
//Check Left Upper Diagonal
for (inti = row-1, j = col-1; i>= 0 && j >= 0; i--, j--) {
if(board[i][j] == 1){
return false;
}
}
//Check Right Upper Diagonal
for (inti = row-1, j = col+1; i>= 0 && j <board.length; i--, j++) {
if(board[i][j] == 1){
return false;
}
}
//Check in same Column
for (inti = row-1; i>= 0; i--) {
if(board[i][col] == 1){
return false;
}
}
return true;
}
private static void printBoard(int[][] board){
for (int row = 0; row <board.length; row++) {
for (int col = 0; col <board.length; col++) {
if(board[row][col] == 1){
System.out.print("Q ");
}else{
System.out.print("_ ");
}
}
System.out.println();
}
System.out.println("");
}
}
```

Output:

```
_ Q _ _
_ _ _ Q
Q _ _ _
_ _ Q _
```

Viva questions:
What are the drawbacks of Merge Sort algorithm.
Discuss about union operation on sets
Describe find operation on sets
Define spanning tree and minimal spanning tree
What do mean by depth first search

13 Aim:.Write a program to implement Sum of subsets using backtracking method
Algorithm:
Step 1: Start.

Step 2: Read the set along with M the keyboard.

Step 3:find out sequence of sub set

Step 4:print solution

Step 5: Stop.

Program:

```
import java.util.Scanner;
public class Subset
{
static int c=0;
public static void main(String[] args)
{
int w[]=new int[10];
int n, d, i, sum=0;
int x[]=new int[10];
Scannerin=new Scanner(System.in);
System.out.println("********** SUBSET PROBLEM ************");
System.out.println("Enter the number of elements: ");
n=in.nextInt();
System.out.println("Enter the elements in increasing order");
for(i=0;i<n;i++)
w[i]=in.nextInt();
System.out.println("Enter the value of d: ");
d=in.nextInt();
for(i=0;i<n;i++)
sum=sum+w[i];
System.out.println("SUM ="+sum);
if(sum < d || w[0] > d){
System.out.println("Subset is not possible ! ");
System.exit(0);
}
subset(0,0,sum,x,w,d);
if(c==0)
System.out.println("Subset is not possible ! ");
System.out.println("\n********** ********* *************");
}
static void subset(int cs, int k, int r,int x[],int w[],int d)
{
x[k] = 1;
if(cs+w[k] == d)
{
c++;
System.out.print("\nSolution "+c+" is {");
for(int i=0;i<=k;i++)
if(x[i] == 1)
{
System.out.print(w[i]+" ");
}
System.out.print("}");
}
else if((cs + w[k] + w[k+1]) <= d)
subset(cs + w[k], k+1, r-w[k],x,w,d);
if((cs + r -w[k]) >=d && (cs + w[k+1]) <= d)
{
x[k] = 0;
subset(cs, k+1, r-w[k],x,w,d);
}
}
}
```

Output:

```
********** SUBSET PROBLEM ************
Enter the number of elements:
5
Enter the elements in increasing order
1 2 3 6 8
Enter the value of d:
9
SUM =20
Solution 1 is {1 2 6 }
Solution 2 is {1 8 }
Solution 3 is {3 6 }
```

Viva questions:

Define breadth first search

Differentiate Breadth first search and depth first search

Describe AND/OR graph

Explain game tree
Define an articulation point?

14Aim:.Write a program to implement Graph Colouring using backtracking method
Algorithm:
Step 1: Start.

Step 2: Read the no of vertices,edges from the keyboard.

Step 3:assign colors to each vertex

Step 4: print coloring of the

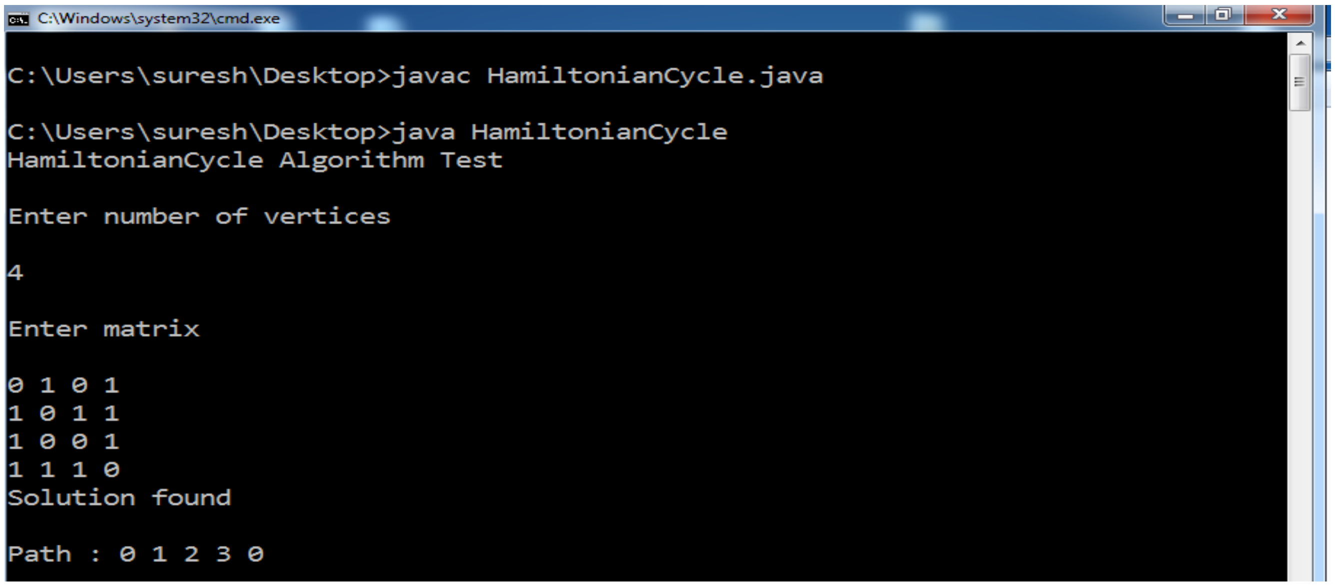
Step 5: Stop.

Program:
```
  import java.util.Scanner;
      import java.util.Arrays;

      /** Class HamiltonianCycle **/
      public class HamiltonianCycle
      {
         private int V, pathCount;
         private int[] path;
         private int[][] graph;
         /** Function to find cycle **/
         public void findHamiltonianCycle(int[][] g)
         {
            V = g.length;
            path = new int[V];

            Arrays.fill(path, -1);
            graph = g;
            try
            {
               path[0] = 0;
               pathCount = 1;
               solve(0);
                System.out.println("No solution");
            }
            catch (Exception e)
            {
               System.out.println(e.getMessage());
```

```
        display();
    }
}
/** function to find paths recursively **/
public void solve(int vertex) throws Exception
{
    /** solution **/
    if (graph[vertex][0] == 1 && pathCount == V)
        throw new Exception("Solution found");
    /** all vertices selected but last vertex not linked to 0 **/
    if (pathCount == V)
        return;

    for (int v = 0; v < V; v++)
    {
        /** if connected **/
        if (graph[vertex][v] == 1 )
        {
            /** add to path **/
            path[pathCount++] = v;
            /** remove connection **/
            graph[vertex][v] = 0;
            graph[v][vertex] = 0;

            /** if vertex not already selected  solve recursively **/
            if (!isPresent(v))
                solve(v);

            /** restore connection **/
            graph[vertex][v] = 1;
            graph[v][vertex] = 1;
            /** remove path **/
            path[--pathCount] = -1;
        }
    }
}
/** function to check if path is already selected **/
```

```
    public boolean isPresent(int v)
    {
        for (int i = 0; i < pathCount - 1; i++)
            if (path[i] == v)
                return true;
        return false;
    }
    /** display solution **/
    public void display()
    {
        System.out.print("\nPath : ");
        for (int i = 0; i <= V; i++)
            System.out.print(path[i % V] +" ");
        System.out.println();
    }
    /** Main function **/
    public static void main (String[] args)
    {
        Scanner scan = new Scanner(System.in);
        System.out.println("HamiltonianCycle Algorithm Test\n");
        /** Make an object of HamiltonianCycle class **/
        HamiltonianCycle hc = new HamiltonianCycle();

        /** Accept number of vertices **/
        System.out.println("Enter number of vertices\n");
        int V = scan.nextInt();

        /** get graph **/
        System.out.println("\nEnter matrix\n");
        int[][] graph = new int[V][V];
        for (int i = 0; i < V; i++)
            for (int j = 0; j < V; j++)
                graph[i][j] = scan.nextInt();
        hc.findHamiltonianCycle(graph);
}
}
```

OutPut:



```
C:\Users\suresh\Desktop>javac HamiltonianCycle.java

C:\Users\suresh\Desktop>java HamiltonianCycle
HamiltonianCycle Algorithm Test

Enter number of vertices

4

Enter matrix

0 1 0 1
1 0 1 1
1 0 0 1
1 1 1 0
Solution found

Path : 0 1 2 3 0
```

Viva questions:
Define breadth first search
Differentiate Breadth first search and depth first search
Describe AND/OR graph
Explain game tree
Define an articulation point?

15.Write a program to implement Travelling sales person using branch and bound, dynamic programming

Algorithm:
Step 1: Start.

Step 2: Read the no of city's along with cost from the keyboard.

Step 3:visit all city's exactly once and return back to initial city

Step 4:print total tour cost

Step 5: Stop.

program
```
import java.util.Scanner;
public class Tsp
{
public static void main(String[] args)
{
int c[][]=new int[10][10], tour[]=new int[10];
Scanner in = new Scanner(System.in);
int i, j,cost;
System.out.println("**** TSP DYNAMIC PROGRAMMING *******");
System.out.println("Enter the number of cities: ");
int n = in.nextInt();
```

```
if(n==1)
{
System.out.println("Path is not possible");
System.exit(0);
}
System.out.println("Enter the cost matrix");
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
c[i][j] = in.nextInt();
System.out.println("The entered cost matrix is");
for(i=1;i<=n;i++)
{
for(j=1;j<=n;j++)
{
System.out.print(c[i][j]+"\t");
}
System.out.println();
}
for(i=1;i<=n;i++)
tour[i]=i;
cost = tspdp(c, tour, 1, n);
System.out.println("The accurate path is");
for(i=1;i<=n;i++)
System.out.print(tour[i]+"->");
System.out.println("1");
System.out.print("The accurate mincost is "+cost);
System.out.println("******* ****************************");
}
static int tspdp(int c[][], int tour[], int start, int n)
{
int mintour[]=new int[10], temp[]=new int[10], mincost=999,
ccost, i, j, k;
if(start == n-1)
{
return (c[tour[n-1]][tour[n]] + c[tour[n]][1]);
}
for(i=start+1; i<=n; i++)
{
for(j=1; j<=n; j++)
temp[j] = tour[j];
temp[start+1] = tour[i];
temp[i] = tour[start+1];
if((c[tour[start]][tour[i]]+(ccost=tspdp(c,temp,start+1,n)))<mincost)
{
mincost = c[tour[start]][tour[i]] + ccost;
for(k=1; k<=n; k++)
mintour[k] = temp[k];
}
}
```

```
for(i=1; i<=n; i++)
tour[i] = mintour[i];
return mincost;
}
}
```

Output:

```
**** TSP DYNAMIC PROGRAMMING *******
Enter the number of cities:
4
Enter the cost matrix
0 1 3 6
1 0 2 3
3 2 01
6 3 1 0
The entered cost matrix is
0 1 3 6
1 0 2 3
3 2 0 1
6 3 10
The accurate path is
1->2->4->3->1
```

Viva questions:

Define breadth first search

Differentiate Breadth first search and depth first search

Describe AND/OR graph

Explain game tree

Define an articulation point?