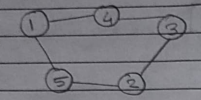
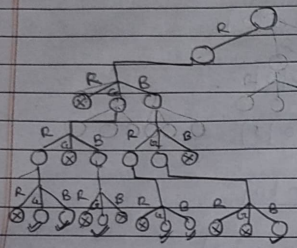
Q.1) e.g.→



m = (R G B).

let's start from vertex 1



Possible ways for coloring are as follows:

1) RGRG
2) RGRB
3) RGBG
4) RBGB
5) RBRG
6) RBRB.

Assignment No. 06

Q.1] Write short note on P, NP, NP Hard and NP complete.

→ Complexity classes are useful in organizing similar types of problems.

The complexity classes are : 1) P class 2) NP class 3) NP Hard class 4) NP Complete class.

The 'P' in the class stands for 'Polynomial Time'.

1) P class → Polynomial Class.

- P in the P class stands for Polynomial Time.
- It is collection of decision problems that can be solved by a deterministic algorithm in polynomial time.

Features of P-class :-

- The solutions to P problems is easy to find.
- P is often a class of computational problems that are solvable.

This class contains the natural problems like,

1) calculating the greatest common divisor.
2) Finding a maximum matching.

2) NP class –

- The NP in NP class stands for Nondeterministic Polynomial Time.
- It is collection of decision problems that can be

solved by a non-deterministic polynomial time Algorithm in

- The solutions of the NP class are hard to find.
- Problems of NP can be verified by a Turing machine in polynomial time.
- This class contains problems that one would like to be able to solve effectively.
1) Boolean satisfiability problem.
2) Hamiltonian path problem.
3) Graph coloring.

3) NP-Hard Class :- An NP-Hard problem is at least as the hardest problem in NP and it is a class of problems such that every problem in NP reduces to NP-hard.

- All NP-hard problems are not in NP.
- It takes long time to check them.
Some of the examples of problems in NP-hard are:
1) Halting problem.
2) Qualified Boolean formulas.

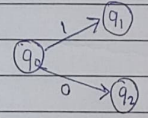
3) NP-Complete class : A problem is NP-complete if it is both NP and NP-hard. NP-complete problems are the hard problems in NP.

- NP-complete problems are special as any problem in NP class can be transformed or reduced into NP-complete problem in polynomial time.
eg. - 1) Decision ~~ves~~ version of 0/1 knapsack.
2) Hamiltonian cycle.

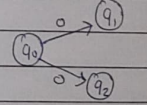
Q.2] Explain Deterministic Algorithm & Non-deterministic Algorithm with example.

→

1) Deterministic algorithm : In a deterministic algorithm, for a given particular input, the computer will always produce the same output going through the same states. ~~but in~~
2) Non-deterministic algorithm : In the case of the non-deterministic algorithm, for the same input, the compiler may produce different output in different runs.
- Non-deterministic algorithms can't solve the problem in polynomial time and can't determine what is the next step.
- The non-deterministic algorithms can show different behaviors for the same input on different execution & there is a degree of randomness to it.



Deterministic Algorithm.



Non-deterministic Algorithm

- In deterministic algorithms, the path of execution for algorithm is same in every execution.

Page No.
Date

- in case of non-deterministic algorithms, the path of execution is not same for algorithm in every execution & could take any random path for its execution.