Page No.
Date

NAME: AFNAN ATTAR PRN:- F1911200B CLASS:- BE COMP II
SUBJECT:- DAA

UNIT 1 and 2 Qbank

* UNIT 1

Q1) What is an algorithm.

Ans 1. An algorithm is a procedure used for solving a problem or performing a computation.

2. Algorithms act as an exact list of instructions that conduct specified actions step by step in either hardware or software routines.

3. Algorithms can be expressed as natural languages, programming languages, pseudocode, flowcharts and control tables.

4. Natural language expressions are rare, as they are more ambiguous.

5. Programming languages are normally used for expressing algorithms executed by a computer.

6. Machine learning is a good example of an algorithm, as it uses multiple algorithms to predict outcomes without being explicitly programmed to do so.

Q2) List and explain the terms to be consider while designing an algorithm.

Ans An algorithm must satisfy the following designing:-

I) Input:-
The algorithm must have input values from a specific set.

II) Output:-
The algorithm must produce the output values from a

specified set of input values.

3. Finiteness:
For any input, the algorithm must terminate after a finite number of steps.

4. Definiteness:
All steps of the algorithm must be precisely defined.

5. Effectiveness:
It must be possible to perform each step correctly and in a finite amount of time.
We must also take care about space and time complexity of algorithm.

Q 3) What are the iterative algorithm design issues?

Ans Iterative algorithm design issues :-

I] Use of loops :-

1. We need to set loop variables to value which are appropriate for solving smallest instance of problem.
2. To find iterative condition as loop variables are changed in every iteration.
3. Loop termination condition occurs when we know how many times to iterate.

II] Efficency of algorithm :-

1. Removing redundant computations outside loop.
2. Referencing of array element.
3. Inefficency due to late termination.
4. Early detection of designed output conditions.

III] Estimating and specifying execution time :-

1. Performance is measured by computational model.
2. Reflects specified input conditions.
3. Independent of specific programming language.

Page No.
Date

* UNIT : 2

Q1) What is meant by upper bound?

Ans 1. Let $U(n)$ be the running time of an algorithm A, then $g(n)$ is the Upper Bound of A if there exist two constants C and N such that $U(n) \leq C * g(n)$ for $n > N$.

2. Upper bound of an algorithm is shown by the asymptotic notation called Big-Oh (O).

3. According to upper bound theory for an upper bound $U(n)$ of an algorithm, we can always solve the problem at most $U(n)$ time.

4. Time taken by a known algorithm to solve a problem with worse case input gives us the upper bound.

Q2) Differentiate between polynomial and non-polynomial problems.

Ans	Polynomial problems	Non-polynomial problems
1.	These set of problems which can be solved in polynomial time by deterministic algorithms.	These set of problems can be solved in non-deterministic polynomial time.
2.	Problem belongs to class P if it is easy to find solution for problem.	Problem belongs to NP if solution is easy to verify but hard to find.
3.	Can be solved and verified in polynomial time.	Can be verified in polynomial time but cannot be solved.
4.	All P problems are deterministic	All NP problems are non-determinis

6.	P problems are subset of NP problems.	NP problems are superset of P problems.
8.	Examples : Selection sort, linear search.	Example: TSP, Knapsack problem.

Q3) Write a short note on NP problems :-

Ans 1. A problem is called NP (Nondeterministic polynomial) if its solution can be guessed and verified in polynomial time; nondeterministic means that no particular rule is followed to make the guess.

2. If a problem is NP and all other NP problems are polynomial-time reducible to it, the problem is NP-Complete

3. Thus, finding an efficient algorithm for any NP-Complete problem implies that an efficient algorithm can be found for all such problems, since any problem belonging to this class can be recast into any other member of the class.

4. It is not known whether any polynomial-time algorithms will ever be found for NP-complete problems and determining whether these problems and determining whether these problems are tractable or intractable remains one of the most important questions in theoretical computer science