Victoria University
of Bangladesh

MID Term Assessment

Md Bakhtiar Chowdhury

ID: 2121210061

Department: CSE

Semester: Spring -2023

Batch: 21th

Course Title: Algorithms

Course Code: CSI 227

Submitted To:

Umme Khadiza Tithi

Lecturer, Department of Computer Science & Engineering

Victoria University of Bangladesh

Submission Date: 19 April, 2023

Answer to the question no 1(a)

# What is Algorithm? Categories and Characteristics of Algorithm ?

Answer:

Algorithm :An algorithm is a set of guidelines or rules used to solve a problem or carry out a task. It is a methodical approach that may be used to solve a mathematical riddle, run a computer program, or carry out a task like sorting a list.

- An algorithm is a finite lists of instructions in sequence to solve the computation problems.
- An algorithm is a step by step of finite number of process to solve the problems.You can write the algorithms in any language which is understandable to the persons (programmers)
- Branching and repetition are included in the steps of an algorithm.
- This branching and repetition depend on the problem for which Algorithm is developed.
- All the steps of Algorithm during the definition should be written in a human-understandable language which does not depend on any programming language.
- We can choose any programming language to implement the Algorithm.
- Pseudocode and flow chart are popular ways to represent an algorithm.

# Categories of Algorithm

There are several categories of algorithms, including:

- **Sorting algorithms:** These algorithms are used to sort a list of items in a specific order, such as ascending or descending order.
- **Searching algorithms:** These algorithms are used to find a specific item in a list of items.
- **Graph algorithms:** These algorithms are used to solve problems related to graphs, such as finding the shortest path between two nodes.

- **Computational geometry algorithms:** These algorithms are used to solve problems related to geometry, such as finding the area of a polygon.
- **Machine learning algorithms:** These algorithms are used to train and build models that can learn from data and make predictions.

Some characteristics of algorithms include:

- **Input**: An algorithm takes input data or a set of instructions as its input.
- **Output**: An algorithm produces an output or a result based on the input data and the set of instructions.
- **Finiteness**: An algorithm must have a finite number of steps and must terminate after a finite number of iterations.
- **Unambiguous**: An algorithm must have clear and precise instructions that are easy to understand and follow.
- **Deterministic**: An algorithm must produce the same output for the same input, regardless of the environment or the platform used to run it.
- **Effectiveness:**-An algorithm should have effectiveness and produce well defined output of any programs. Effectiveness means, an algorithms should have good method which produce effective output with less time and less storage capacity.

Answer to the question no 1(b)

# Disadvantages of Algorithm and Advantages of Algorithm?

Answer:
Disadvantages of Algorithm:

Complexity: Some problems are too complex to be solved using a single algorithm, requiring multiple algorithms to be used together, which can be challenging to manage.

Limitations: Algorithms are limited by the capabilities of the hardware and software used to execute them. Some problems may be too large or too complex to be solved using existing algorithms.

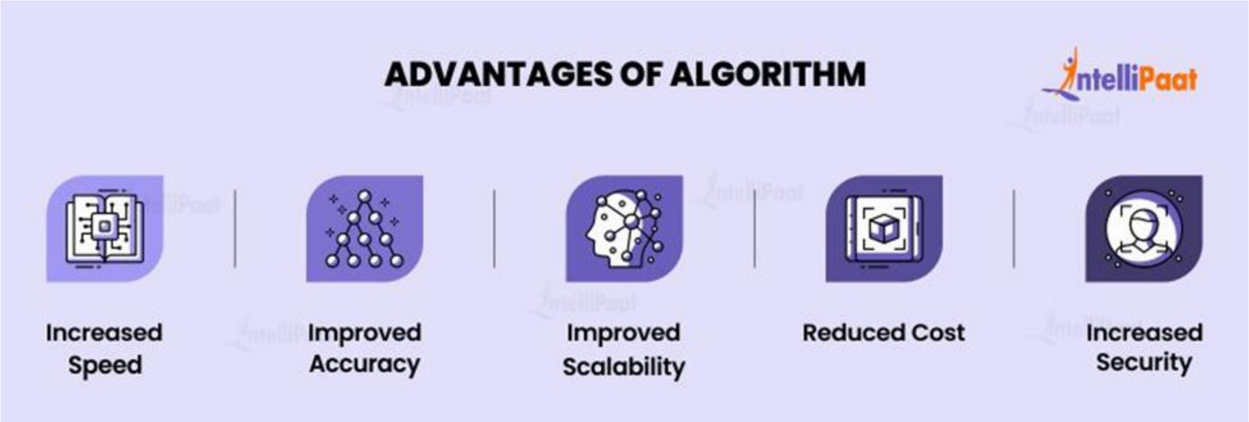
Bias: Algorithms can be biased towards certain types of input data or certain solutions, leading to inaccurate or unfair results.

Difficulty of development: Developing an algorithm requires significant time and expertise, making it difficult for individuals or organizations with limited resources to develop their own algorithms.

Need for updates: Algorithms may need to be updated periodically to keep up with changing requirements or to address bugs and other issues that may arise over time.

Advantages of Algorithm?

In the section above, we looked at some of the characteristics of algorithms. Let's look at some of the advantages of algorithms now:



- **Increased speed:** Algorithms are designed to run faster than other computer programs and the time taken to execute them is far less than that of other programs.
- **Improved accuracy:** Algorithms are designed to be more accurate than other programs and are more reliable when it comes to output.
- **Improved scalability:** Algorithms are designed to scale to large data sets and can process large amounts of data quickly and efficiently.
- **Reduced cost:** Algorithms are designed to reduce the cost associated with certain tasks and can increase the efficiency of business processes.
- **Increased security:** Algorithms are designed to be secure and can help protect data from potential threats.

Answer to the question no 1(c)

# Types of Algorithm? Define Greedy Algorithm ?

Answer:

There are many types of algorithms, and they can be categorized based on their purpose or the problem they solve. Some common types of algorithms include:

1. **Sorting Algorithms:** These algorithms are used to sort a list of items in a specific order, such as ascending or descending order. Examples include bubble sort, quicksort, and merge sort.

2. **Searching Algorithms:** These algorithms are used to find a specific item in a list of items. Examples include linear search, binary search, and interpolation search.

3. **Graph Algorithms:** These algorithms are used to solve problems related to graphs, such as finding the shortest path between two nodes. Examples include Dijkstra's algorithm, breadth-first search, and depth-first search.

4. **Computational Geometry Algorithms:** These algorithms are used to solve problems related to geometry, such as finding the area of a polygon. Examples include the convex hull algorithm and the Euclidean distance algorithm.

5. **Dynamic Programming Algorithms:** These algorithms are used to solve problems by breaking them down into smaller subproblems and solving them iteratively. Examples include the Fibonacci sequence algorithm and the Knapsack problem algorithm.

6. **Divide and Conquer Algorithms:** These algorithms solve problems by breaking them down into smaller subproblems, solving them independently, and then combining the solutions. Examples include the binary search algorithm and the merge sort algorithm.

7. **Greedy Algorithms:** These algorithms solve problems by making locally optimal choices at each step, with the hope of finding a global optimal solution. Examples include the Kruskal's algorithm and the Huffman coding algorithm.

8. **Backtracking Algorithms:** These algorithms solve problems by incrementally building a solution, and when it fails, backtracking to the previous step and trying a different approach. Examples include the N-Queens problem algorithm and the Sudoku solver algorithm.

9. **Randomized Algorithms:** These algorithms use randomness to solve problems, often by sampling a solution space to find a good approximation of the optimal

solution. Examples include the Monte Carlo algorithm and the Quick select algorithm.

These are just a few examples of the many types of algorithms that exist.

Answer to the question no 2(a)

# What is Searching Algorithm and Sorting Algorithm?

Answer:

Searching Algorithm: A searching algorithm is a method for finding a specific item or value in a collection of items, such as an array or a list. There are many types of searching algorithms, but some common ones include linear search, binary search, and interpolation search.

Linear search, also known as sequential search, involves scanning each item in the collection until the target value is found. This method is simple but can be time-consuming for large collections.

Binary search involves dividing the collection in half repeatedly until the target value is found. This method is more efficient than linear search, particularly for large collections, but requires that the collection is sorted beforehand.

Interpolation search is a variation of binary search that works better for collections with uniformly distributed values. It involves estimating the position of the target value based on its value relative to the minimum and maximum values in the collection.

Sorting Algorithm: A sorting algorithm is a method for arranging a collection of items in a specific order, such as ascending or descending order. There are many types of sorting algorithms, but some common ones include bubble sort, insertion sort, and quicksort.

Bubble sort involves repeatedly swapping adjacent elements in the collection until the entire collection is sorted. This method is simple but not very efficient, particularly for large collections.

Insertion sort involves building the sorted list one item at a time by inserting each new item into its correct position. This method is efficient for small collections but not very efficient for larger collections.

Quicksort involves partitioning the collection into smaller sub-collections and sorting each sub-collection separately. This method is efficient for large collections and is often used as a standard sorting algorithm in many programming languages.

Other common sorting algorithms include selection sort, merge sort, and heap sort. The choice of sorting algorithm depends on the size of the collection, the distribution of the values in the collection, and the desired time and space complexity.

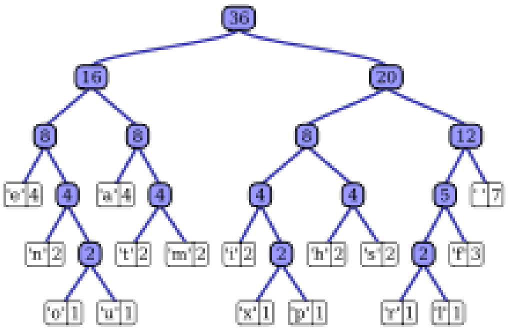
Answer to the question no 2(b)

# Define Huffman Coding ?

Answer:

Huffman coding is a lossless data compression algorithm. The idea is to assign variable-length codes to input characters, lengths of the assigned codes are based on the frequencies of corresponding characters.

The variable-length codes assigned to input characters are Prefix Codes, means the codes (bit sequences) are assigned in such a way that the code assigned to one character is not the prefix of code assigned to any other character. This is how Huffman Coding makes sure that there is no ambiguity when decoding the generated bitstream.



Let us understand prefix codes with a counter example. Let there be four characters a, b, c and d, and their corresponding variable length codes be 00, 01, 0 and 1. This coding leads to ambiguity because code assigned to c is the prefix of codes assigned to a and b. If the compressed bit stream is 0001, the de-compressed output may be “cccd” or “ccb” or “acd” or “ab”.

See this for applications of Huffman Coding.

There are mainly two major parts in Huffman Coding

1. Build a Huffman Tree from input characters.
2. **2.** Traverse the Huffman Tree and assign codes to characters.

Answer to the question no 2(c)

# Define mathematical Algorithm and Graph Algorithm ?

Mathematical Algorithm:

A mathematical algorithm is a series of steps or instructions used to solve a mathematical problem. Mathematical algorithms can be used to solve a wide range of problems, such as finding the roots of a polynomial, solving a system of linear equations, or computing the value of a complex function.

Mathematical algorithms can be expressed in a variety of forms, including pseudocode, flowcharts, or programming languages such as MATLAB or Python. The efficiency and accuracy of a mathematical algorithm depend on its design, implementation, and input data.

Graph Algorithm:

A graph algorithm is a method for analyzing or manipulating a graph, which is a mathematical representation of a set of objects or nodes connected by edges or links. Graph algorithms can be used to solve a wide range of problems, such as finding the shortest path between two nodes, detecting cycles in a graph, or coloring the nodes of a graph.

Graph algorithms can be classified into two main categories: traversal algorithms and optimization algorithms. Traversal algorithms involve visiting all the nodes in a graph, while optimization algorithms involve finding the best or optimal solution to a specific problem.

Some common graph algorithms include depth-first search, breadth-first search, Dijkstra's algorithm, Bellman-Ford algorithm, and Kruskal's algorithm. The choice of graph algorithm depends on the specific problem being solved, the size and structure of the graph, and the desired time and space complexity.

>>>>>>END<<<<<<