Program for Interview Preparation

Week-1 - Strings and Arrays

Arrays

Arrays are the simplest data structure and the most likely to be used in computing. They represent collections of objects of the same type.

- We typically implement arrays as a single contiguous memory block that stores these elements sequentially.
- Each element on an array is accessed using an index. For example: Given an array A = [2,4,6] of size 3, the value 2 has an index of 0. We could also say 2 is in location 0 of the array. The value 4 has an index of 1 and the value 6 has an index of 2.
- Accessing elements in an array is a very fast operation. (O(1))

By default an array is not a dynamic data structure. It is not designed to grow or shrink in size as it's being used. However C++ standard library provides std::vector<T> which is a dynamic array and std::deque<T> which is an efficient double ended queue.
Common uses of arrays include:

- To efficiently perform computations on all elements in parallel.
- Aggregation, i.e. a computation on an array that yields a single result, like summing all elements.
- To sort or select elements.

Sorting the arrays
Sorting is often required in many problems. There are many different implementations, all with different performance characteristics and constraints.
Major classes of commonly used sorting algorithms:

O(n^2) family: selection sort, bubble sort, insertion sort
O(nlogn) family: merge sort, heap sort (Merge sort visualisation) (Merge sort code explanation)

Stable vs Online vs In-Place sorting
Stable: A sorting algorithm that maintains the relative order of numbers/records in the case of a tie.

Online: An algorithm that can process and maintain sorted order, without having the entire input available from the start.
In-place: An algorithm that transforms input using no additional data structure, overwriting the input itself. However, a small constant extra space used for variables is allowed. Selection sort in an In-place sorting algorithm.

Preparing for the Interview

- ➔ Practice array problems thoroughly, many interview questions involve arrays.
- ➔ Know the performance characteristics of the common sorting algorithms.
- ➔ Know how to use the standard library sorting APIs and their specifics on the language you will use.
- ➔ Different interview questions have different constraints in which some sorting algorithm can't be used.
- ➔ Example: If the input can only be read once and you have to sort it as you go, you need an online sorting algorithm.
- ➔ Know how sorting algorithms work in detail. Some interview questions need algorithms which are immediate steps on some sorting algorithms.
    - ◆ **Examples**: Merge sorted lists: this is an intermediate step in merge sort which comes up somewhat frequently in interview questions.
    - ◆ Partition: move an element in an array into position k where a[i] <= a[k] for i < k and a[i] >= a[k] for i > k. This is a step used in the canonical quick sort implementation and may come handy on some interview questions.

Practice Questions: (in random order)

1. https://leetcode.com/problems/maximum-subarray/
2. https://leetcode.com/problems/sort-colors/
3. https://leetcode.com/problems/best-time-to-buy-and-sell-stock/
4. https://practice.geeksforgeeks.org/problems/inversion-of-array-1587115620/1/
5. https://www.interviewbit.com/problems/subarray-with-given-xor/
6. https://leetcode.com/problems/longest-substring-without-repeating-characters/
7. https://practice.geeksforgeeks.org/problems/largest-subarray-with-0-sum/1
8. https://leetcode.com/problems/search-a-2d-matrix/
9. https://www.geeksforgeeks.org/move-zeroes-end-array/
10. https://www.interviewbit.com/problems/next-permutation/

[For those who want more]

Strings

Strings are a special kind of array, namely made out of characters for storing textual data.

- We treat strings separately from arrays because certain operations which commonly applied to strings do not make sense for general arrays
- For example -- comparison, joining, splitting, searching for substrings, replacing one string by another, parsing, etc.

Preparing for the Interview

- Know how strings are represented in memory
- Understand basic operators such as comparison, copying, matching, joining (concatenation), splitting, etc.
- Advanced string processing algorithms often use hash tables and dynamic programming
- Familiarize yourself with your programming language of choice's suite of string utility functions and any weird quirks

Practice Questions:

1. https://www.interviewbit.com/problems/longest-common-prefix/
2. https://www.interviewbit.com/problems/add-binary-strings/
3. https://www.interviewbit.com/problems/minimum-characters-required-to-make-a-string-palindromic/
4. https://www.interviewbit.com/problems/reverse-the-string/
5. https://www.interviewbit.com/problems/longest-palindromic-substring/
6. https://www.interviewbit.com/problems/amazing-subarrays/
7. https://www.geeksforgeeks.org/lexicographically-first-palindromic-string/
8. https://www.interviewbit.com/problems/atoi/
9. https://www.interviewbit.com/problems/integer-to-roman/

[For those who want more]

Number Theory

Some important number theory concepts that are often used in problem solving efficiently.

Modular Arithmetic

There are many problems which require printing the answer modulo some large prime number (for example: 10^9 + 7). For such problems and many more, keep these properties [1] [2] in mind.

Sieve of Eratosthenes

This is an important technique to find all the prime numbers from 1 to N in (N(log(logN))) time. If this is done naively, it would take O(N√N) time.

Prime numbers and sieve of Eratosthenes

Extended Euclidean Algorithm

It helps you to find the GCD of two numbers efficiently.

Euclidean Algorithm

Practice Questions:

1. https://leetcode.com/problems/count-primes/
2. https://www.interviewbit.com/problems/all-factors/
3. https://leetcode.com/problems/x-of-a-kind-in-a-deck-of-cards/