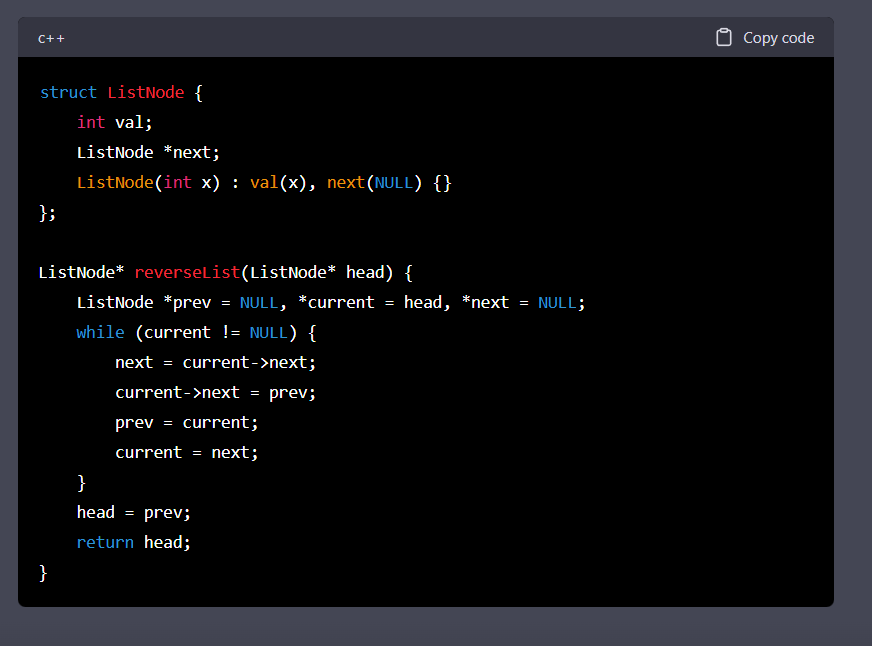
PayPal

1. How do you reverse a singly linked list? (very important question)

To reverse a singly linked list, you can follow these steps:

1. Initialize three pointers: **prev** pointing to NULL, **current** pointing to the head of the linked list, and **next** pointing to NULL.
2. Traverse the linked list using the **current** pointer.
3. For each node, store the next node in the **next** pointer.
4. Update the **next** pointer of the current node to point to the previous node.
5. Update the **prev** pointer to the current node.
6. Update the **current** pointer to the next node.
7. Repeat steps 3-6 until the end of the linked list is reached.
8. Update the head pointer to point to the last node (which is the new head of the reversed list).



1. Find out the minimum element in Stack in O(1) time.

In this implementation, we use the stack data structure from the STL to implement the MinStack class. The s stack holds the actual elements, while the minStack stack keeps track of the minimum elements seen so far.

When we push an element x onto the s stack, we also check if it's less than or equal to the current minimum element (minStack.top()). If it is, we push it onto the minStack.

When we pop an element from the s stack, we check if it's the current minimum element (minStack.top()). If it is, we also pop it from the minStack.

To get the top element of the s stack, we simply return s.top().

To get the minimum element in the s stack, we simply return minStack.top().

All of these operations run in O(1) time, since we are only pushing, popping, or accessing the top element of the stacks.

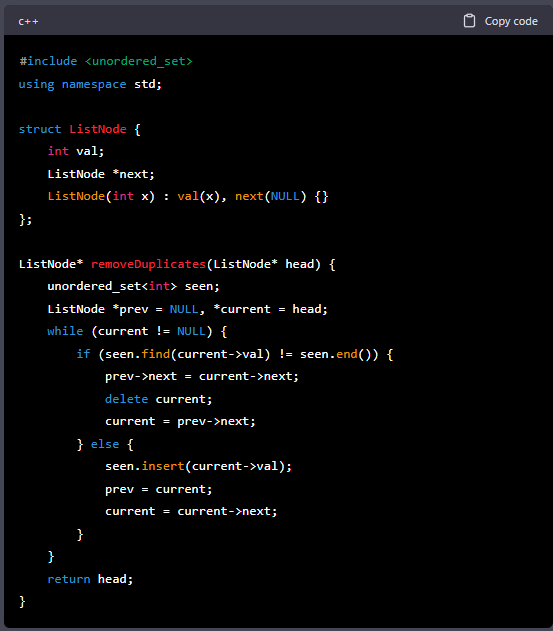
Text

Description automatically generated

1. How are duplicate nodes removed in an unsorted linked list?

To remove duplicate nodes from an unsorted linked list in C++, we can use a hash table to keep track of the elements that have been seen so far.

Here's the implementation of the algorithm to remove duplicate nodes from an unsorted linked list in C++:



In this implementation, we define a ListNode struct to represent each node of the linked list. The removeDuplicates function takes the head pointer of the original linked list as input and returns the new head pointer of the list with duplicates removed.

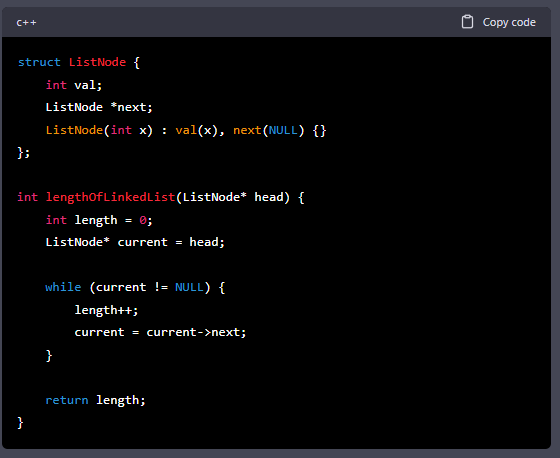
We use an unordered\_set<int> to store the values of the elements that have been seen so far. We initialize the prev and current pointers to NULL and the head of the original linked list, respectively.

Then, we traverse the original linked list using the current pointer. For each node, we check if its value is in the seen set using seen.find(current->val). If it is, we delete the current node by updating the next pointer of the previous node (prev->next) to point to the next node (current->next) and deleting the current node. If it isn't, we insert its value into the seen set, update the prev pointer to the current node, and update the current pointer to the next node.

Finally, we return the head pointer of the modified linked list.

This algorithm runs in O(n) time, where n is the number of nodes in the linked list, since we need to traverse the entire linked list once and perform constant time operations for each node.

1. How do you find the length of a singly linked list?



1. Difference between abstraction and interface.

Abstraction and interface are both important concepts in object-oriented programming, but they serve different purposes.

Abstraction refers to the process of hiding the implementation details of a class or function, and exposing only the essential features that are relevant to the user. It is a way to manage complexity by creating simplified models of complex systems. Abstraction can be achieved in several ways, including through the use of abstract classes and interfaces.

Interfaces, on the other hand, define a set of methods that a class must implement in order to provide a particular functionality. An interface provides a contract between the implementing class and the code that uses it, specifying what methods must be implemented and what their behavior should be. Interfaces can be used to define a common set of methods that multiple classes can implement, allowing them to be used interchangeably.

In summary, abstraction is a general concept that refers to the process of hiding complexity, while interfaces are a specific construct in object-oriented programming that define a set of methods that a class must implement. Abstraction can be achieved using various techniques, including through the use of interfaces.

1. What is Multi-Threading in Operating System?

Multithreading in operating systems is the ability of a CPU to provide concurrent execution of two or more parts of a program called threads. Each thread is a separate sequence of execution within the same process. A process can have multiple threads, each of which can perform a different task.



1. How to find the third node from the end in a singly linked list?
2. To find the third node from the end in a singly linked list, we can use the "two pointers" approach, where we use two pointers that are initially positioned at the head of the list. We first move the first pointer ahead by three nodes, and then move both pointers one node at a time until the first pointer reaches the end of the list. At this point, the second pointer will be pointing to the third node from the end.
3. Here's an example implementation in C++:

Text

Description automatically generated Text

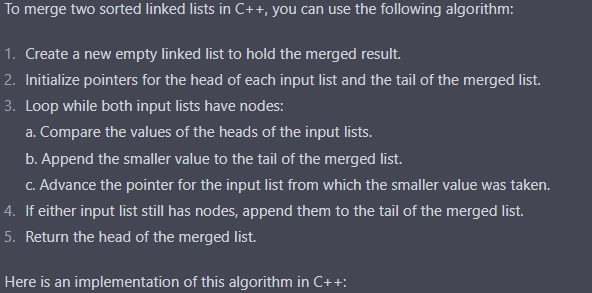
Description automatically generated

1. If you are given an array “X” comprising “n” positive integers, write a function to compute the inversion count of the array. (Youtube it)
2. What do you know about SQL Injection?

SQL injection is a type of security vulnerability that occurs when an attacker is able to inject malicious SQL code into an application that uses SQL (Structured Query Language) to communicate with a database. The vulnerability arises when user input is not properly validated or sanitized before being used in a SQL query.

For example, consider a login page that uses the following SQL query to validate the user's credentials:

1. Given two Linked Lists, “A” and “B,” that are sorted. Write a code to merge the two Linked Lists.



1. How will you implement a post-order traversal on a tree? Can this traversal be done without recursion?

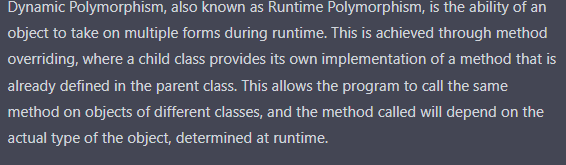
A screenshot of a computer

Description automatically generated with medium confidence

Text

Description automatically generated

1. Explain Dynamic Polymorphism and Runtime Polymorphism with relevant examples. Write program functions to execute each of these properties.



Text

Description automatically generated

1. Explain Method Overriding and Method Overloading by providing examples.

Method Overloading: Method overloading is a feature that allows a class to have more than one method with the same name but with different parameters. The compiler differentiates between the overloaded methods based on the number, types, and order of the parameters. Method overloading is also called compile-time polymorphism or static polymorphism because the decision on which method to call is made at compile time.

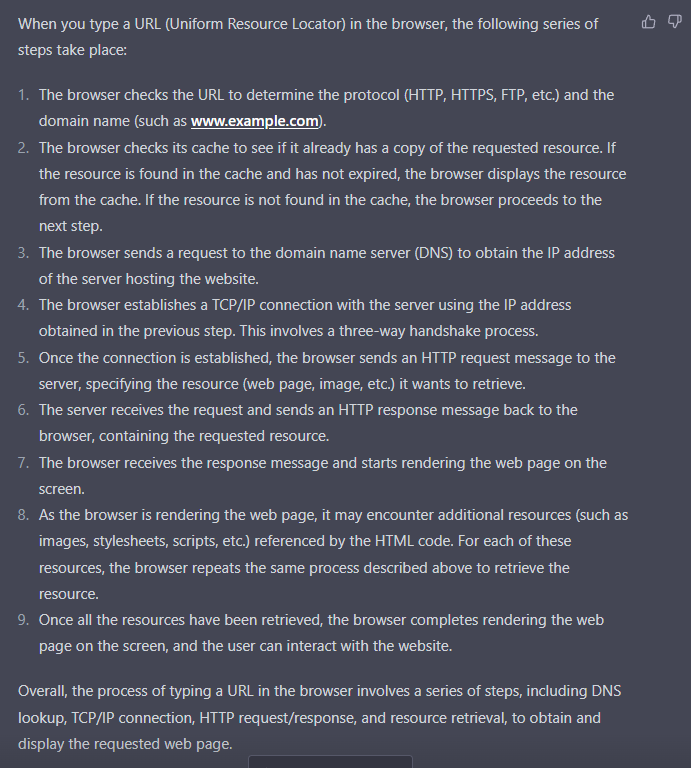
Method Overriding: Method overriding is a feature that allows a subclass to provide its own implementation of a method that is already provided by its parent class. In method overriding, the signature (name, return type, and parameters) of the method in the subclass must match the signature of the method in the superclass.

1. How will you find the missing number in a given integer array of 1 to 100?
2. Remove nth duplicate element from the given linked list. **Example:** 1 -> 2 -> 3 -> 1 -> 2 -> 3 ->1 ->2 ->3, n=2 Here the nth(2nd) duplicate

element in this list is 2.

So, return, 1 -> 3 -> 1 -> 3 -> 1 -> 3

1. Given a binary tree and a number. Your task is to return “true” if the tree has a root- to-leaf path where all values along the path add up to the given number. Return “false” if the path is not found.
2. Find the first non-repeating character in the string (with only lowercase letters) <https://leetcode.com/problems/first-unique-character-in-a-string/>
3. Given a set of non-negative integers and a value sum. Your task is to determine if there is a subset of the given set where the values add up to sum.
4. Find if a linked list is a palindrome [expected time & space complexity: O(n) & O(1)]. <https://leetcode.com/problems/palindrome-linked-list/>
5. What is singleton class and its implementation?
6. What is diamond problem in Java?
7. What happens when you type a URL in the browser?



1. Find the Kth largest element in an array. - [https://leetcode.com/problems/kth-largest-](https://leetcode.com/problems/kth-largest-element-in-an-array/) [element-in-an-array/](https://leetcode.com/problems/kth-largest-element-in-an-array/)