Day 13 - 26th June 2025

Linked Lists:

&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

Linked list in c++

#include <bits/stdc++.h>

using namespace std;

// Define a Node class

class Node{

public:

int data; // Data part of the node

Node\* next; // Pointer to the next node

// Constructor for convenience

Node(int value) : data(value), next(nullptr) {}

};

// Class for singly linked list

class Linkedlist{

private:

Node\* head; // Pointer to the head of the list

public:

// Constructor to initialize an empty list

Linkedlist(){

head = nullptr;

}

// Function to insert a node at the end

void insertAtEnd(int value){

Node\* newNode = new Node(value);

if(head == nullptr){

head = newNode; // If list is empty, make newNode the head

}

else{

Node\* temp = head;

while (temp->next != nullptr){

temp = temp->next; // Traverse to the last node

}

temp->next = newNode; // Link the last node to newNode

}

}

// Function to delete a Node by Value

void deleteByValue(int value){

if(head == nullptr){

return;

}

if(head->data == value){

Node\* temp = head;

head = head->next; // Move head to the next node

delete temp; // Free memory of the deleted node

return;

}

Node\* temp = head;

while(temp->next && temp->next->data != value){

temp = temp->next; // Traverse to find the node to delete

}

if(temp->next){

Node\* nodeToDelete = temp->next;

temp->next = temp->next->next; // Unlink the node

delete nodeToDelete; //Free Memory

}

}

// Function to display the list

void display(){

Node\* temp = head;

while(temp != nullptr){

cout << temp->data << "->";

temp = temp->next;

}

cout << "NULL" <<endl;

}

// Destructor to free all allocated memory

~LinkedList() {

Node\* temp;

while (head) {

temp = head;

head = head->next;

delete temp;

}

}

};

int main() {

LinkedList list;

list.insertAtEnd(10);

list.insertAtEnd(20);

list.insertAtEnd(30);

cout << "Linked List: ";

list.display();

list.deleteByValue(20);

cout << "After Deleting 20: ";

list.display();

return 0;

}

======================================================================================

https://pythontutor.com/render.html#mode=display

By Ahmed.. You can visualise the data structures while executing the code line by line..

================================================================

Task 001

import java.util.LinkedList;

public class Task001\_DS\_LinkedList {

public static void main(String[] args) {

LinkedList<String> fruits = new LinkedList<>();

fruits.add("Apple");

fruits.add("Banana");

fruits.addFirst("Orange");

fruits.addLast("Grapes");

System.out.println("First Element: " + fruits.getFirst());

System.out.println("Last Element: " + fruits.getLast());

fruits.removeFirst();

fruits.removeLast();

for (String fruit : fruits) {

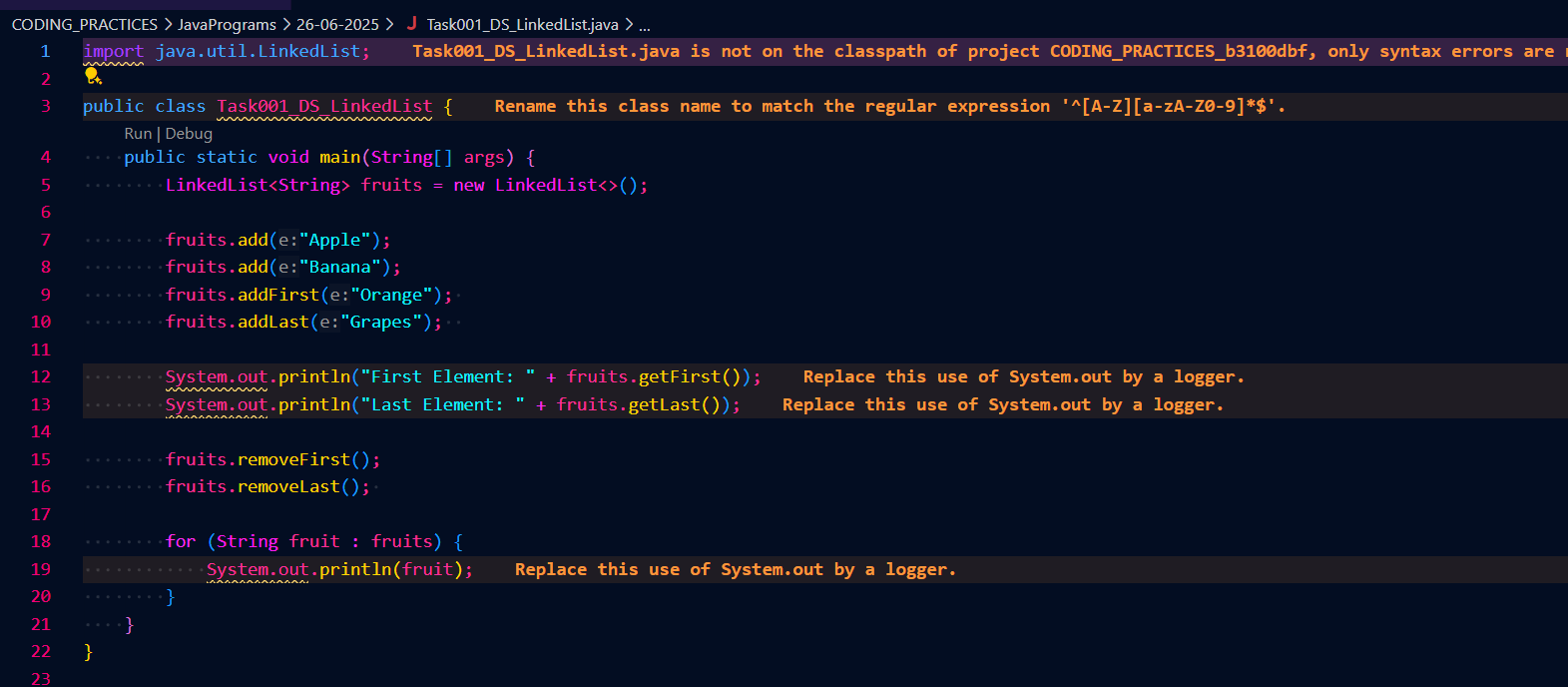
System.out.println(fruit);

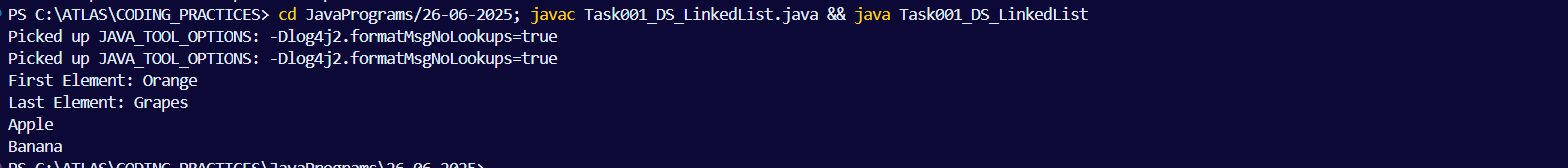
}

}

}

Solution :





======================================================================================================================================================

Task 002 : Try to create a node and add a value to it..

Solution : 