# Name: - AdityaBachawad

# Roll No: - 01

**Batch: - S1 Sub:- D.S**

**Teacher Assignment 2**

## Write a program for Merging of two sorted linked list.

**Soln: -**

Code=>

#include <stdio.h> #include <stdlib.h> struct Node

{

int data;

struct Node\* next;

};

void printList(struct Node\* head)

{

struct Node\* ptr = head; while (ptr)

{

printf("%d —> ", ptr->data); ptr = ptr->next;

}

printf("NULL\n");

}

void push(struct Node\*\* head, int data)

{

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node)); newNode->data = data;

newNode->next = \*head;

\*head = newNode;

}

struct Node\* sortedMerge(struct Node\* a, struct Node\* b)

{

if (a == NULL) {

return b;

}

else if (b == NULL) { return a;

}

struct Node\* result = NULL; if (a->data <= b->data)

{

result = a;

result->next = sortedMerge(a->next, b);

}

else { result = b;

result->next = sortedMerge(a, b->next);

}

return result;

}

int main(void)

{

int keys[] = { 1, 2, 3, 4, 5, 6, 7 };

int n = sizeof(keys)/sizeof(keys[0]); struct Node \*a = NULL, \*b = NULL; for (int i = n - 1; i >= 0; i = i - 2) { push(&a, keys[i]);

}

for (int i = n - 2; i >= 0; i = i - 2) { push(&b, keys[i]);

}

printf("First List: "); printList(a); printf("Second List: "); printList(b);

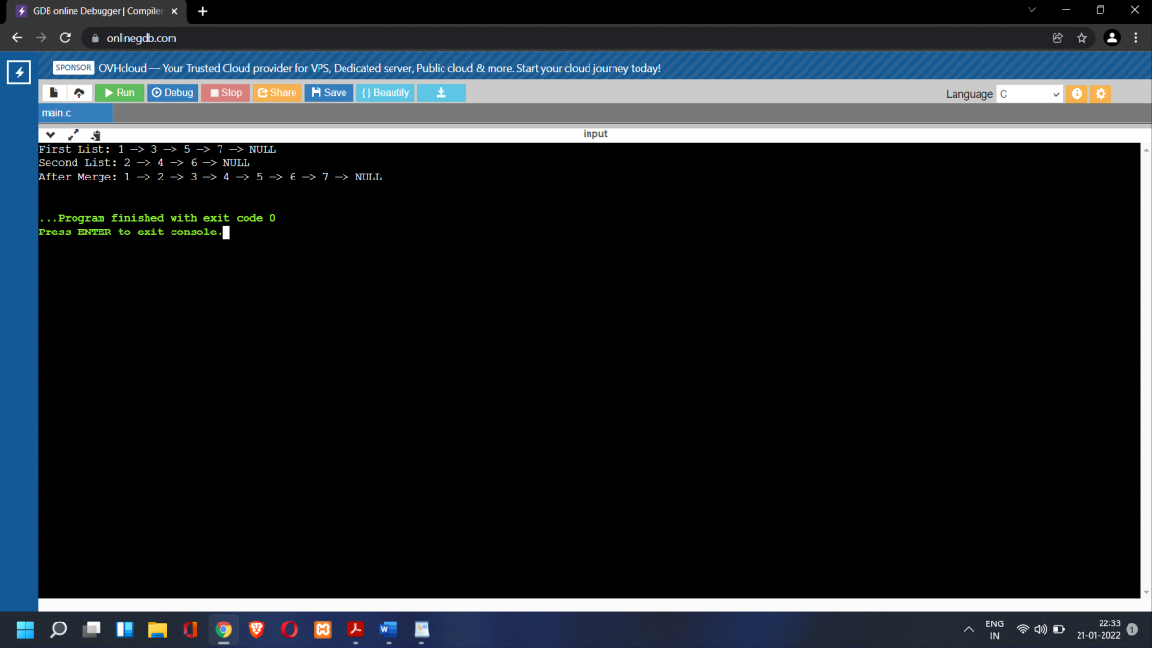
struct Node\* head = sortedMerge(a, b);

printf("After Merge: "); printList(head);

return 0;

}

Output=>



## Write a program for finding height of binary tree.

**Soln:-**

Code=>

#include <stdio.h> #include <stdlib.h> struct node {

int data;

struct node\* left; struct node\* right;

};

int maxDepth(struct node\* node)

{

if (node == NULL) return -1;

else {

int lDepth = maxDepth(node->left); int rDepth = maxDepth(node->right); if (lDepth > rDepth)

return (lDepth + 1); else

return (rDepth + 1);

}

}

struct node\* newNode(int data)

{

struct node\* node

= (struct node\*)malloc(sizeof(struct node)); node->data = data;

node->left = NULL; node->right = NULL; return (node);

}

int main()

{

struct node\* root = newNode(1); root->left = newNode(2);

root->right = newNode(3); root->left->left = newNode(4);

root->left->right = newNode(5);

printf("Height of tree is %d", maxDepth(root)); getchar();

return 0;

}

Output=>

