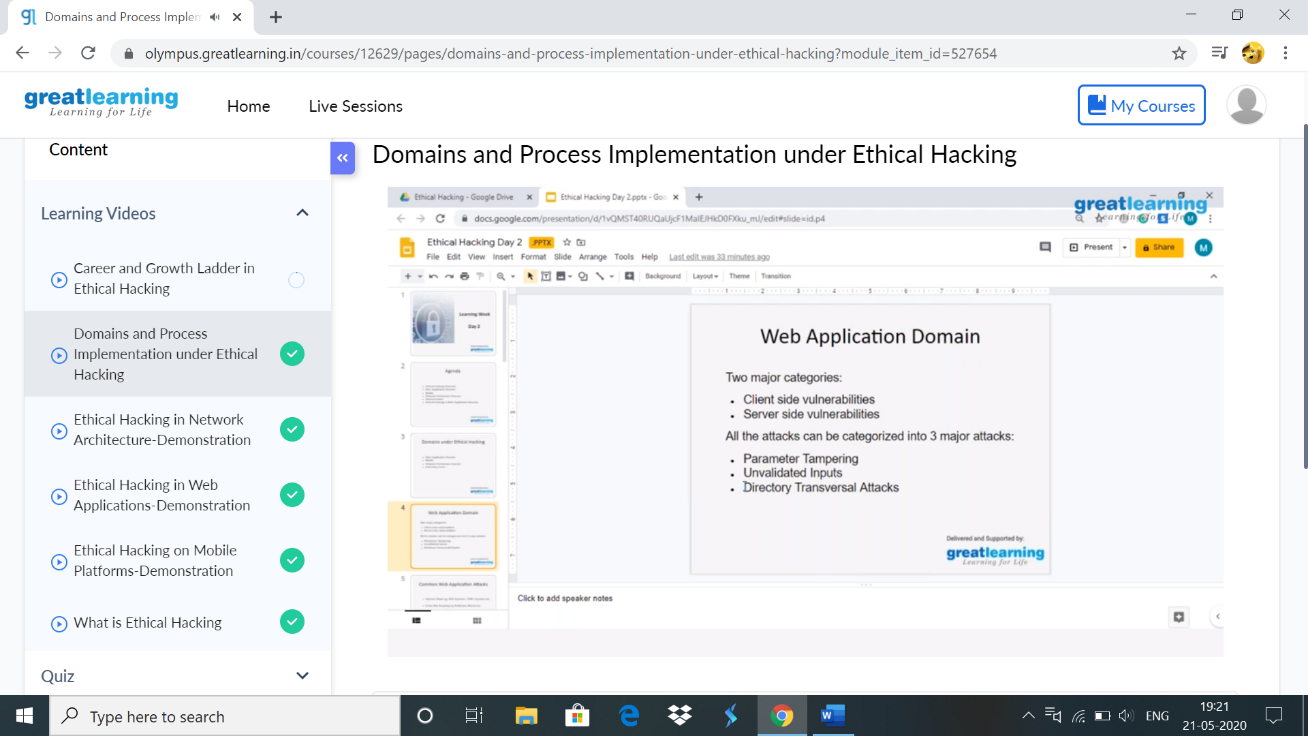
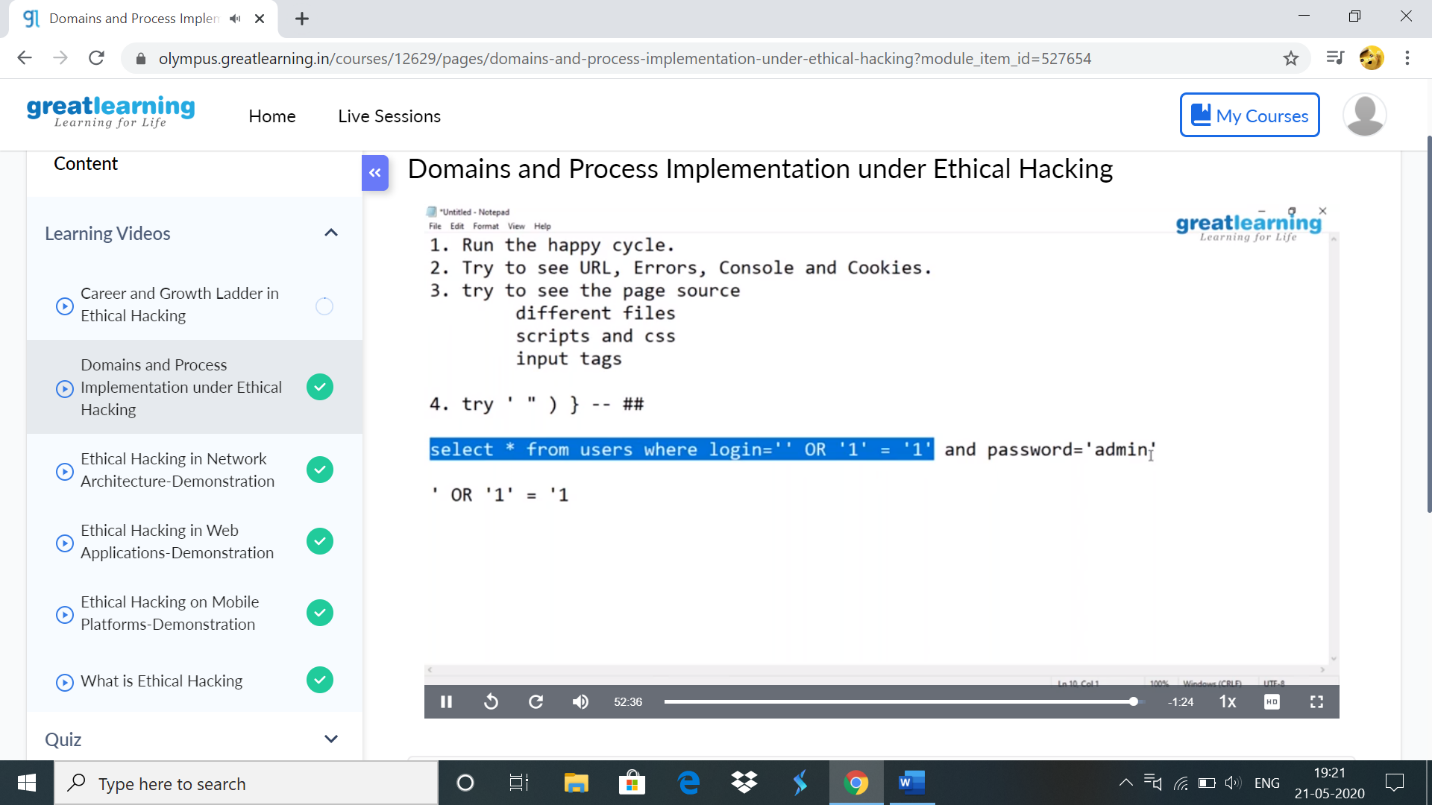
**DAILY ONLINE ACTIVITIES SUMMARY**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Date:** | **21-05-20** | | | | **Name:** | **TANOJ M** | |
| **Sem & Sec** | **VI A** | | | | **USN:** | **4AL16CS113** | |
| **Online Test Summary** | | | | | | | |
| **Subject** | | **OS IA Test** | | | | | |
| **Max. Marks** | | **30** | | **Score** | | **27** | |
| **Certification Course Summary** | | | | | | | |
| **Course** | **Ethical Hacking** | | | | | | |
| **Coding Challenges**   1. Write a C program to construct a singly linked list by removing duplicate elements in the sorted linked list Description: Take a sorted list and traverse the list. Compare the current node element with next adjacent node. If it is same then delete second element, if not retain. Finally print the resulting list. Sample output: Given list {1,2,2,3,3,3,4} Resulting list{1,2,3,4} 2. Create the SLL, and then Reverse the Link in SLL until Head becomes NULL. Each Time Reversing the Link, Head must be moved to next immediate node. | | | | | | | |
| **Certificate Provider** | | | **Great learning** | **Duration** | | | **6 days** |
| **Status:Completed** | | | | | | | |
| **Uploaded the report in Github** | | | | **Yes** | | | |
| **If yes Repository name** | | | | **https://github.com/Tanoj8296/DAILY-STATUS** | | | |
| **Uploaded the report in slack** | | | | **Yes** | | | |

**Online Certification Details**





**Coding Challenge Details**

#include<stdio.h>

#include<stdlib.h>

struct node

{

int data;

struct node\* next;

};

void insert\_elements(struct node\*\* head, int new\_data)

{

struct node\* new\_node = (struct node\*) malloc(sizeof(struct node));

new\_node -> data = new\_data;

new\_node -> next = (\*head);

(\*head) = new\_node;

}

void display\_list(struct node \*node)

{

while (node!=NULL)

{

printf("%d ", node->data);

node = node -> next;

}

void remove\_duplicate\_elements(struct node\* head)

{

struct node\* current = head;

struct node\* next\_next;

if (current == NULL)

return;

while (current -> next != NULL)

{

/\* Compare current node with its next \*/

if (current -> data == current -> next -> data)

{

next\_next = current -> next -> next;

free(current -> next);

current -> next = next\_next;

}

else

{

current = current -> next;

}

}

}

int main()

{

struct node\* head = NULL;

int n;

printf("Enter the total number of elements : ");

scanf("%d", &n);

printf("\nEnter the sorted linked list : ");

int i;

for(i = 0; i < n; i++)

{

int data;

scanf("%d", &data);

insert\_elements(&head, data);

}

printf("\nLinked list before removing duplicates : ");

display\_list(head);

printf("\n");

remove\_duplicate\_elements(head);

printf("\nLinked list after removing duplicates : ");

display\_list(head);

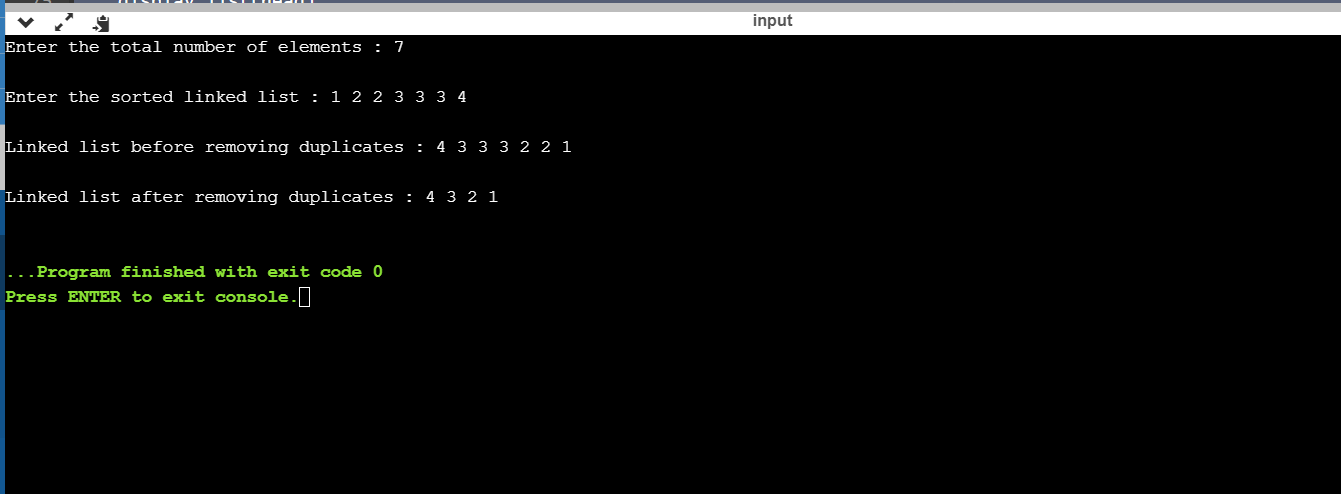
printf("\n");

return 0;

}

**OUTPUT**

|  |
| --- |
|  |

****

2) #include <stdio.h>

#include <stdlib.h>

struct node

{

int num;

struct node \*next;

};

void create(struct node \*\*);

void reverse(struct node \*\*, int);

void release(struct node \*\*);

void display(struct node \*);

int main()

{

struct node \*p = NULL;

int n;

printf("Enter data into the list\n");

create(&p);

printf("Displaying the nodes in the list:\n");

display(p);

printf("Enter the number N to reverse first N node: ");

scanf("%d", &n);

printf("Reversing the list...\n");

if (n > 1)

{

reverse(&p, n - 2);

}

printf("Displaying the reversed list:\n");

display(p);

release(&p);

return 0;

}

void reverse(struct node \*\*head, int n)

{

struct node \*p, \*q, \*r, \*rear;

p = q = r = \*head;

if (n == 0)

{

q = q->next;

p->next = q->next;

q->next = p;

\*head = q;

}

else

{

p = p->next->next;

q = q->next;

r->next = NULL;

rear = r;

q->next = r;

while (n > 0 && p != NULL)

{

r = q;

q = p;

p = p->next;

q->next = r;

n--;

}

\*head = q;

rear->next = p;

}

}

void create(struct node \*\*head)

{

int c, ch;

struct node \*temp, \*rear;

do

{

printf("Enter number: ");

scanf("%d", &c);

temp = (struct node \*)malloc(sizeof(struct node));

temp->num = c;

temp->next = NULL;

if (\*head == NULL)

{

\*head = temp;

}

else

{

rear->next = temp;

}

rear = temp;

printf("Do you wish to continue [1/0]: ");

scanf("%d", &ch);

} while (ch != 0);

printf("\n");

}

void display(struct node \*p)

{

while (p != NULL)

{

printf("%d\t", p->num);

p = p->next;

}

printf("\n");

}

void release(struct node \*\*head)

{

struct node \*temp = \*head;

\*head = (\*head)->next;

while ((\*head) != NULL)

{

free(temp);

temp = \*head;

(\*head) = (\*head)->next;

}

}

**OUTPUT**

