**TUGAS PRAKTIKUM ALGORITMA & STRUKTUR DATA**

**Jilid 5**



**Oleh :**

**Nama : Rosi Arif Mulyadi**

**NRP : 3121522021**

**Prodi : D3 Teknik Informatika PENS PSDKU Sumenep**

**Kelas : 1 ITA D3 Sumenep**

**Dosen :**

**LUSIANA AGUSTIEN M.Kom**

**POLITEKNIK ELEKTRONIKA NEGERI SURABAYA**

**MODUL 3**

**LINKED LIST (Beberapa Skenario Penghapusan data linkedlist)**

**B. Kegiatan Praktikum**

1. Implementasikan dan tentukan output percobaan yang ada dalam modul praktikum ini dan lakukan analisa pada tiap fungsi yang dibuat.

Jawab :

* Listing Program (First Deleted) :

#include <stdio.h>

#include <stdlib.h>

struct node

{

int val;

struct node \*next;

};

void delete\_first\_node(struct node \*\*head)

{

struct node \*tmp;

if(head == NULL || \*head == NULL)return;

tmp = \*head;

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

free(tmp);

}

void print\_list(struct node \*head)

{

printf("H->");

while(head)

{

printf("%d->", head->val);

head = head->next;

}

printf("|||\n");

}

void insert\_front(struct node \*\*head, int value)

{

struct node \* new\_node = NULL;

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

if(new\_node == NULL)

{

printf("Failed to insert element. Out of memory");

}

new\_node->val = value;

new\_node->next = \*head;

\*head = new\_node;

}

void main()

{

int count = 0, i, val;

struct node \* head = NULL;

printf("Enter number of elements : ");

scanf("%d", &count);

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

{

printf("Enter %d th element : ", i);

scanf("%d", &val);

insert\_front(&head, val);

}

printf("Initial Linked List : ");

print\_list(head);

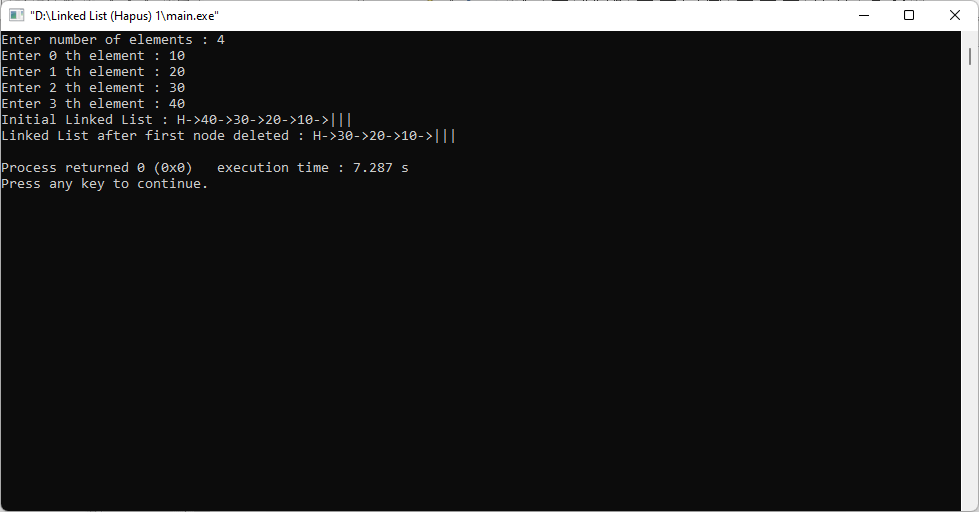
delete\_first\_node(&head);

printf("Linked List after first node deleted : ");

print\_list(head);

}

Output :



Analisa :

* Linked List First Deleted digunakan untuk menghapus value yang berada didepan.
* Tidak bisa menghapus value yang berada di posisi tengah atau akhir.
* Listing Program (N-node Deleted) :

#include <stdio.h>

#include <stdlib.h>

struct node{

int val;

struct node \*next;

};

void delete\_nth\_node(struct node \*\*head, int n) {

struct node \*tmp = NULL;

struct node \*del\_node = NULL;

if(head == NULL || \*head == NULL) return;

tmp = \*head;

if (n == 0) {

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

free(tmp);

return;

}

while(--n > 0 && tmp->next) tmp = tmp->next;

if(tmp->next == NULL) return;

del\_node = tmp->next;

tmp->next = tmp->next->next;

free(del\_node);

}

void print\_list(struct node \*head) {

printf("H->");

while(head)

{

printf("%d->", head->val);

head = head->next;

}

printf("|||\n");

}

void insert\_end(struct node \*\*head, int value) {

struct node\* new\_node = NULL;

struct node\* tmp = \*head;

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

if (new\_node == NULL)

{

printf("Failed to insert element. Out of memory");

}

new\_node->val = value;

new\_node->next = NULL;

if (\*head == NULL) {

\*head = new\_node;

return;

}

while(tmp->next) tmp = tmp->next;

tmp->next = new\_node;

}

void main()

{

int count = 0, i, val, n;

struct node\* head = NULL;

printf("Enter number of elements: ");

scanf("%d", &count);

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

{

printf("Enter %d-th element: ", i);

scanf("%d", &val);

insert\_end(&head, val);

}

printf("Initial Linked List: ");

print\_list(head);

printf("Enter a position: ");

scanf("%d", &n);

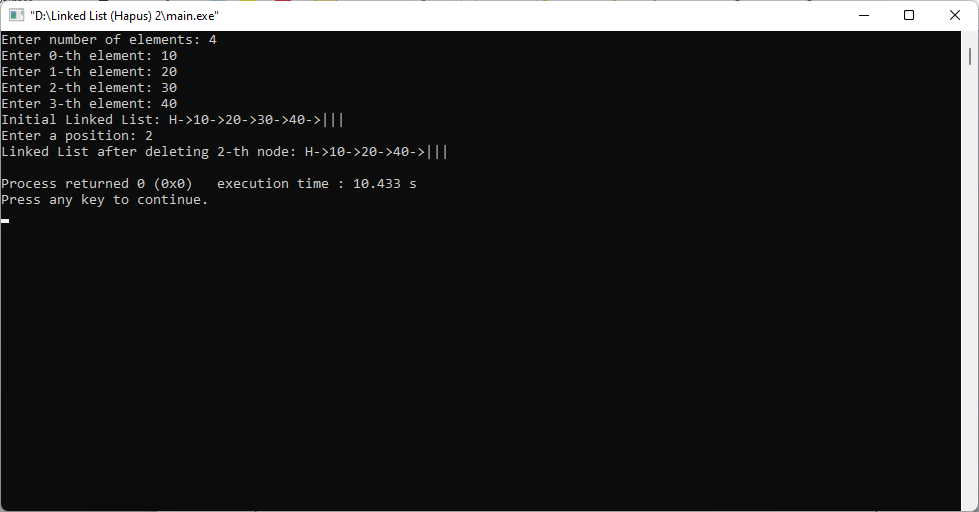
delete\_nth\_node(&head, n);

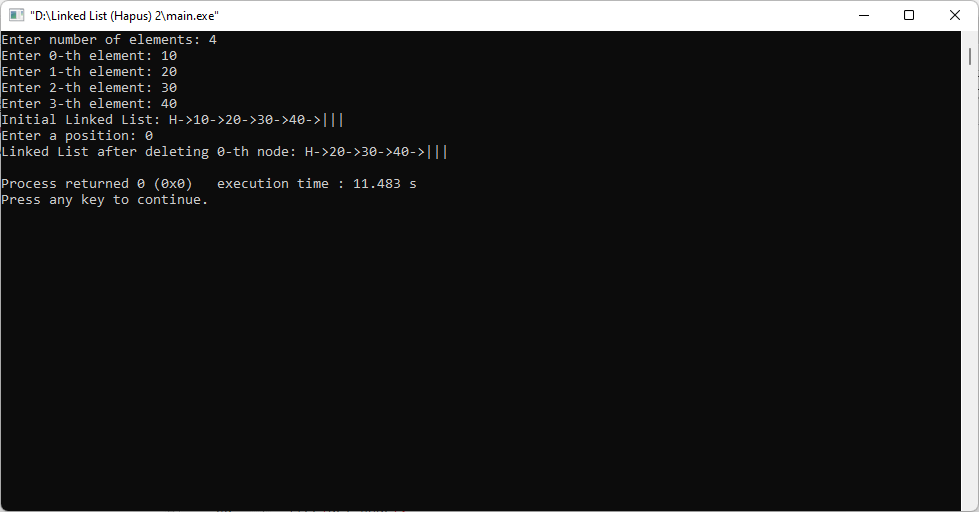
printf("Linked List after deleting %d-th node: ", n);

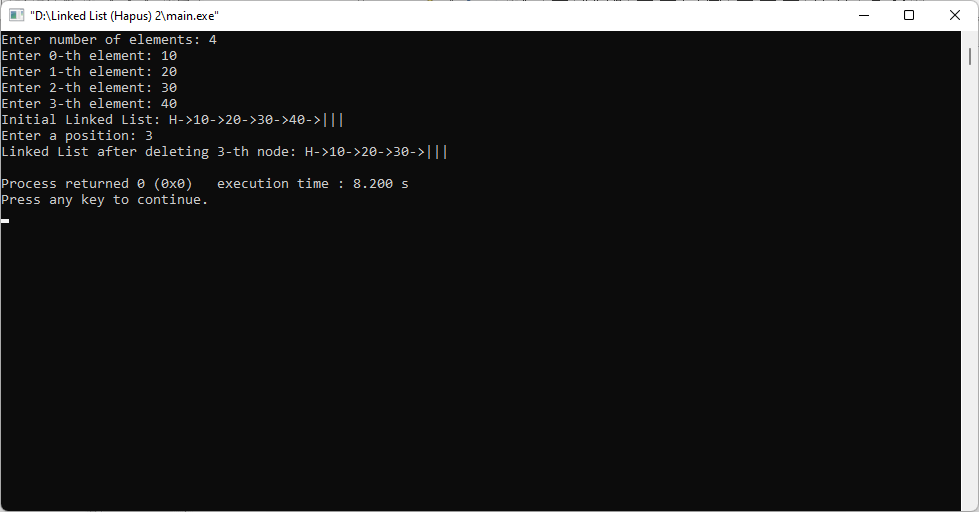
print\_list(head);

}

Output :







Analisa :

* N-node Deleted digunakan untuk menghapus value di salah satu N-node tersebut.
* N-node Deleted dapat menghapus di awal, di tengah, dan juga di akhir tergantung posisi yang kita inputkan.
* Listing Program (Last Deleted) :

#include <stdio.h>

#include <stdlib.h>

struct node

{

int val;

struct node \*next;

};

void delete\_last\_node(struct node \*\*head)

{

struct node \*prev = NULL, \*cur = NULL;

if(head == NULL || \*head == NULL)return;

if((\*head)->next == NULL)

{

free(\*head);

\*head = NULL;

}

prev = \*head;

cur = prev->next;

while(cur->next)

{

prev = prev->next;

cur = cur->next;

}

prev->next = NULL;

free(cur);

}

void print\_list(struct node \*head)

{

printf("H->");

while(head)

{

printf("%d->", head->val);

head = head->next;

}

printf("|||\n");

}

void insert\_front(struct node \*\*head, int value)

{

struct node \* new\_node = NULL;

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

if(new\_node == NULL)

{

printf("Failed to insert element, Out of memory");

}

new\_node->val = value;

new\_node->next = \*head;

\*head = new\_node;

}

void main()

{

int count = 0, i, val;

struct node \* head = NULL;

printf("Enter number of elements : ");

scanf("%d", &count);

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

{

printf("Enter %dth element : ", i);

scanf("%d", &val);

insert\_front(&head, val);

}

printf("Initial Linked List : ");

print\_list(head);

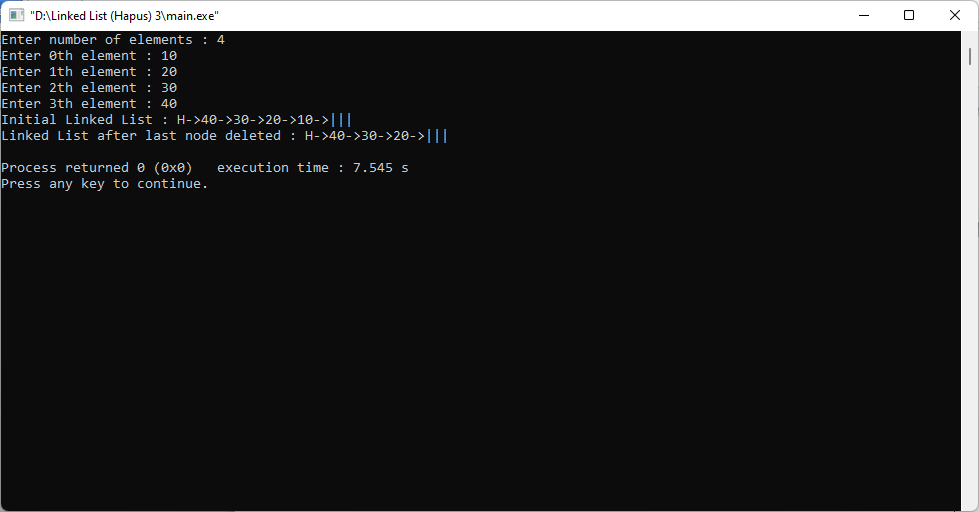
delete\_last\_node(&head);

printf("Linked List after last node deleted : ");

print\_list(head);

}

Output :



Analisa :

* Last Deleted digunakan untuk mengahpus value yang berad di akhir.
* Tidak dapat menghapus value yang berada di awal dan di tengah.

**C. Tugas Praktikum**

1. Implementasikan 3 algoritma penghapusan node (n node, last node, first node) pada program percobaan yang ada pada modul 2.

Jawab :

* Listing Program (First Deleted) :

#include <stdio.h>

#include <stdlib.h>

struct node

{

char nama[50];

int val;

struct node \*next;

};

void delete\_first\_node(struct node \*\*head, char nama[], int value)

{

struct node \*tmp;

if(head == NULL || \*head == NULL)return;

tmp = \*head;

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

free(tmp);

}

void print\_list(struct node \*head)

{

printf("H->");

while(head)

{

printf("%s %i->", head->nama, head->val);

head = head->next;

}

printf("|||\n");

}

void insert\_front(struct node \*\*head, char nama[], int value)

{

struct node \* new\_node = NULL;

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

if(new\_node == NULL)

{

printf("Failed to insert element. Out of memory");

}

strcpy(new\_node->nama,nama);

new\_node->val = value;

new\_node->next = \*head;

\*head = new\_node;

}

void main()

{

int count = 0, i, val;

char nama[10];

struct node \* head = NULL;

printf("Enter number of elements : ");

scanf("%d", &count);

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

{

printf("Enter %d th element : ", i);

scanf("%s", &nama);

fflush(stdin);

printf("Enter %d th element : ", i);

scanf("%d", &val);

fflush(stdin);

insert\_front(&head, nama, val);

}

printf("Initial Linked List : ");

print\_list(head);

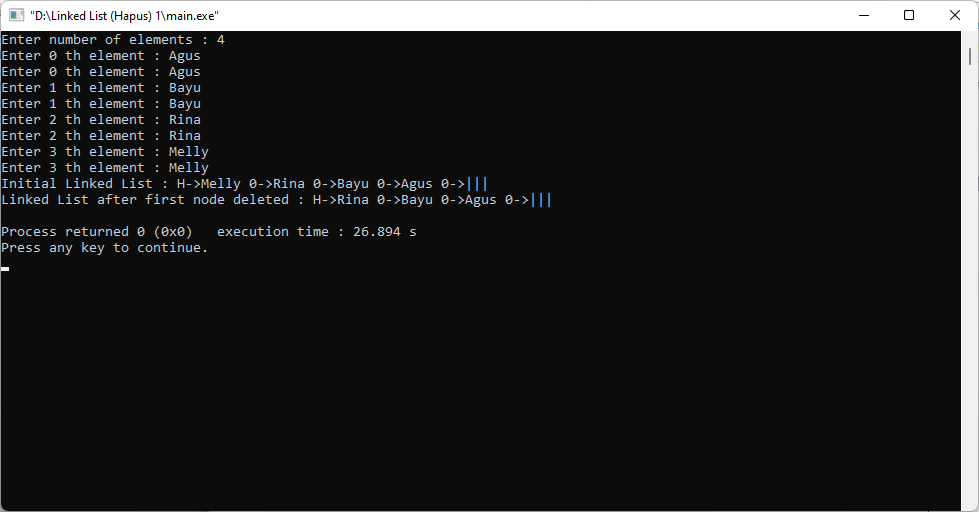
delete\_first\_node(&head, nama, val);

printf("Linked List after first node deleted : ");

print\_list(head);

}

Output :



* Listing Program (N-node Deleted) :

#include <stdio.h>

#include <stdlib.h>

struct node{

char nama[50];

int val;

struct node \*next;

};

void delete\_nth\_node(struct node \*\*head, char nama[], int val) {

struct node \*tmp = NULL;

struct node \*del\_node = NULL;

if(head == NULL || \*head == NULL) return;

tmp = \*head;

if (val == 0) {

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

free(tmp);

return;

}

while(--val > 0 && tmp->next) tmp = tmp->next;

if(tmp->next == NULL) return;

del\_node = tmp->next;

tmp->next = tmp->next->next;

free(del\_node);

}

void print\_list(struct node \*head) {

printf("H->");

while(head)

{

printf("%s %i->", head->nama, head->val);

head = head->next;

}

printf("|||\n");

}

void insert\_end(struct node \*\*head, char nama[], int value) {

struct node\* new\_node = NULL;

struct node\* tmp = \*head;

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

if (new\_node == NULL)

{

printf("Failed to insert element. Out of memory");

}

strcpy(new\_node->nama,nama);

new\_node->val = value;

new\_node->next = NULL;

if (\*head == NULL) {

\*head = new\_node;

return;

}

while(tmp->next) tmp = tmp->next;

tmp->next = new\_node;

}

void main()

{

int count = 0, i, val, n;

char nama[10];

struct node \* head = NULL;

printf("Enter number of elements: ");

scanf("%d", &count);

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

{

printf("Enter %d th element : ", i);

scanf("%s", &nama);

fflush(stdin);

printf("Enter %d th element: ", i);

scanf("%d", &val);

fflush(stdin);

insert\_end(&head, nama, val);

}

printf("Initial Linked List: ");

print\_list(head);

printf("Enter a position: ");

scanf("%d", &n);

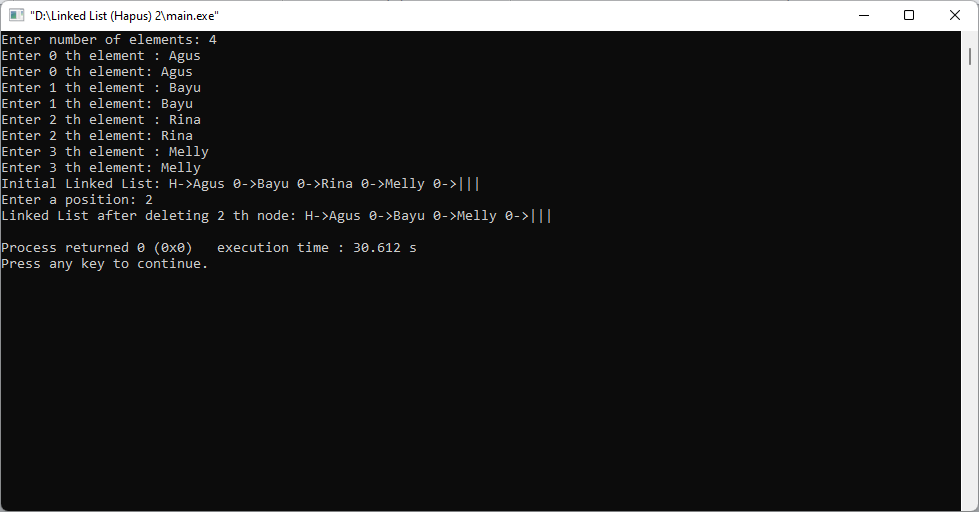
delete\_nth\_node(&head, nama, n);

printf("Linked List after deleting %d th node: ", n);

print\_list(head);

}

Output :



* Listing Program (Last Deleted) :

#include <stdio.h>

#include <stdlib.h>

struct node

{

char nama[50];

int val;

struct node \*next;

};

void delete\_last\_node(struct node \*\*head, char nama[], int value)

{

struct node \*prev = NULL, \*cur = NULL;

if(head == NULL || \*head == NULL)return;

if((\*head)->next == NULL)

{

free(\*head);

\*head = NULL;

}

prev = \*head;

cur = prev->next;

while(cur->next)

{

prev = prev->next;

cur = cur->next;

}

prev->next = NULL;

free(cur);

}

void print\_list(struct node \*head)

{

printf("H->");

while(head)

{

printf("%s %i->", head->nama, head->val);

head = head->next;

}

printf("|||\n");

}

void insert\_front(struct node \*\*head, char nama[], int value)

{

struct node \* new\_node = NULL;

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

if(new\_node == NULL)

{

printf("Failed to insert element, Out of memory");

}

strcpy(new\_node->nama,nama);

new\_node->val = value;

new\_node->next = \*head;

\*head = new\_node;

}

void main()

{

int count = 0, i, val;

char nama[10];

struct node \* head = NULL;

printf("Enter number of elements : ");

scanf("%d", &count);

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

{

printf("Enter %d th element : ", i);

scanf("%s", &nama);

fflush(stdin);

printf("Enter %dth element : ", i);

scanf("%d", &val);

fflush(stdin);

insert\_front(&head, nama, val);

}

printf("Initial Linked List : ");

print\_list(head);

delete\_last\_node(&head, nama, val);

printf("Linked List after last node deleted : ");

print\_list(head);

}

Output :

