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

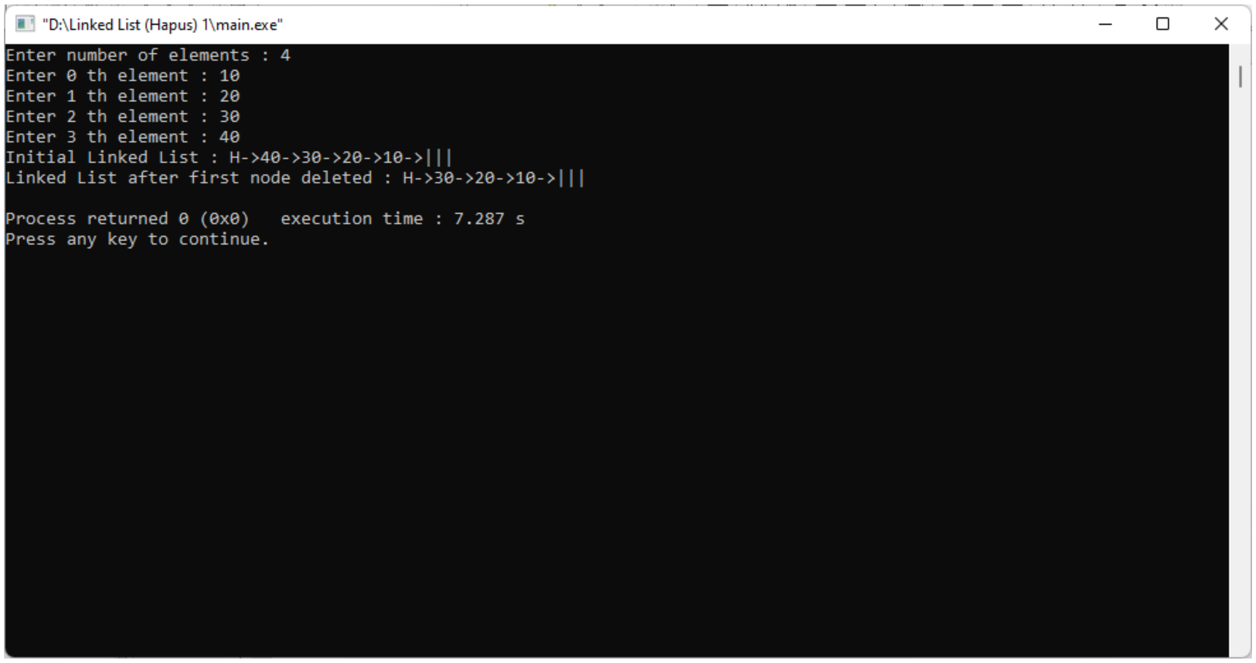
1. Implementasikan dan tentukan output percobaan yang ada dalam modul praktikum ini dan lakukan analisa pada tiap fungsi yang dibuat.

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 :



```
Enter number of elements : 4
Enter 0 th element : 10
Enter 1 th element : 20
Enter 2 th element : 30
Enter 3 th element : 40
Initial Linked List : H->40->30->20->10->|||
Linked List after first node deleted : H->30->20->10->|||

Process returned 0 (0x0)   execution time : 7.287 s
Press any key to continue.
```

Analisa :

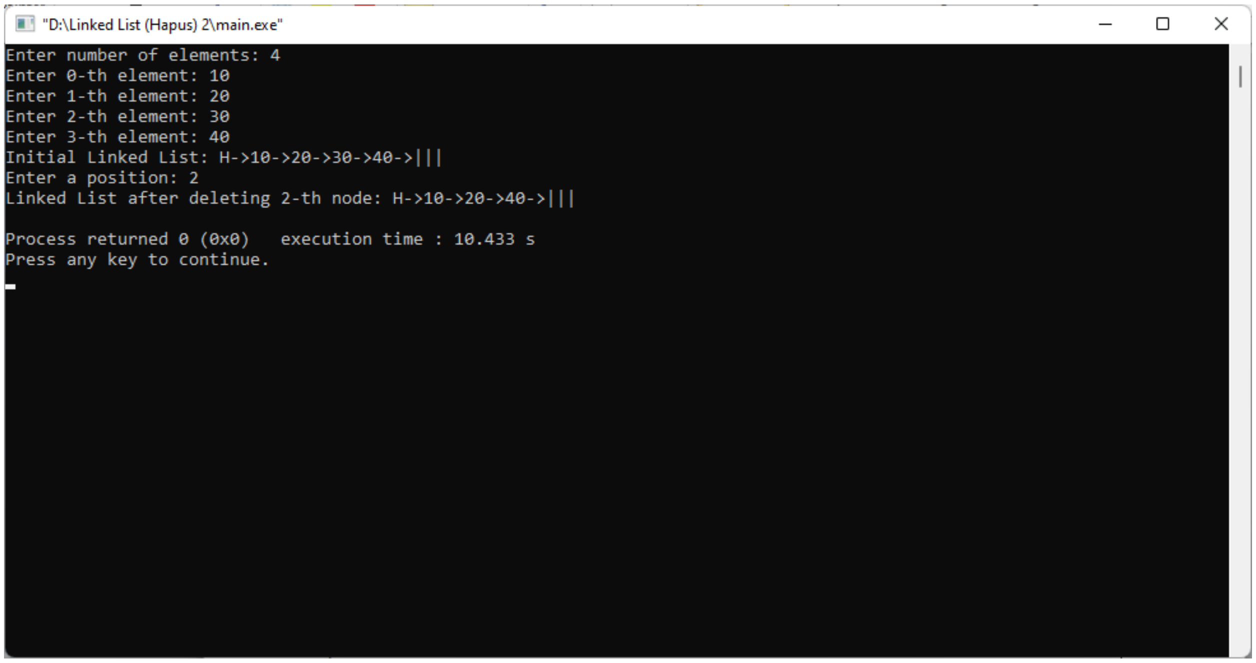
- Linked List First Deleted digunakan untuk menghapus value yang berada didepan.
- Tidak bisa menghapus value yang berada di posisi tengah atau akhir.

➢ Listing Program (N-node Deleted) :

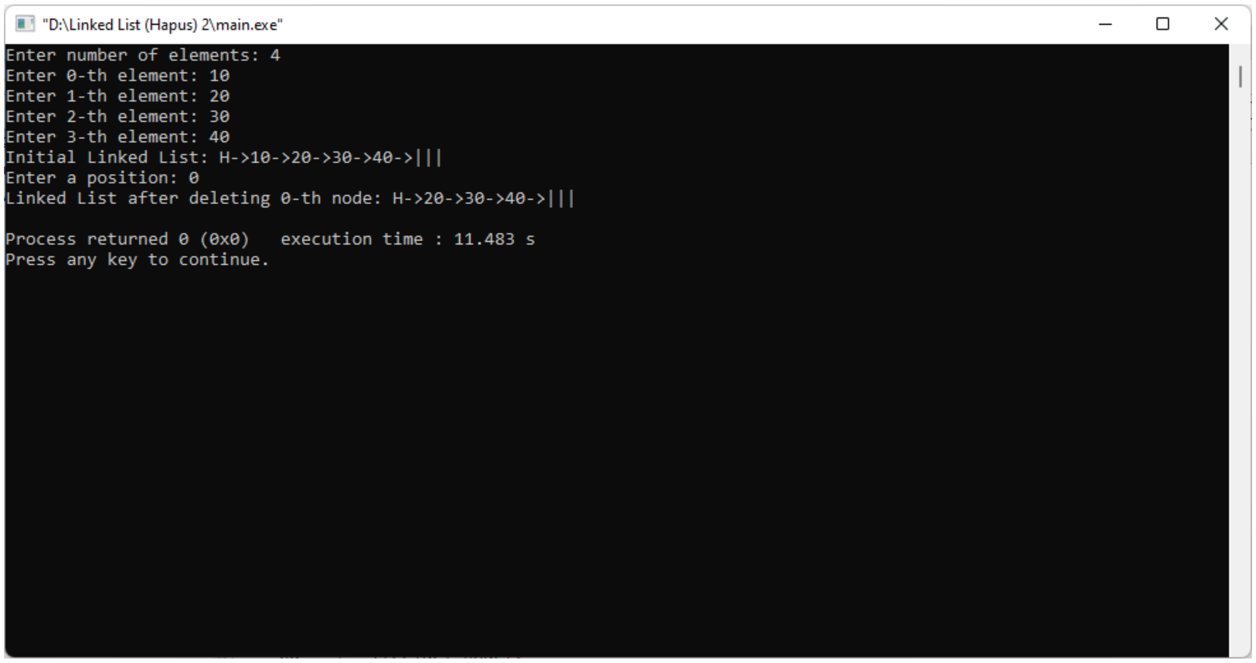
```
#include <stdio.h>
#include <stdlib.h>

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;
```

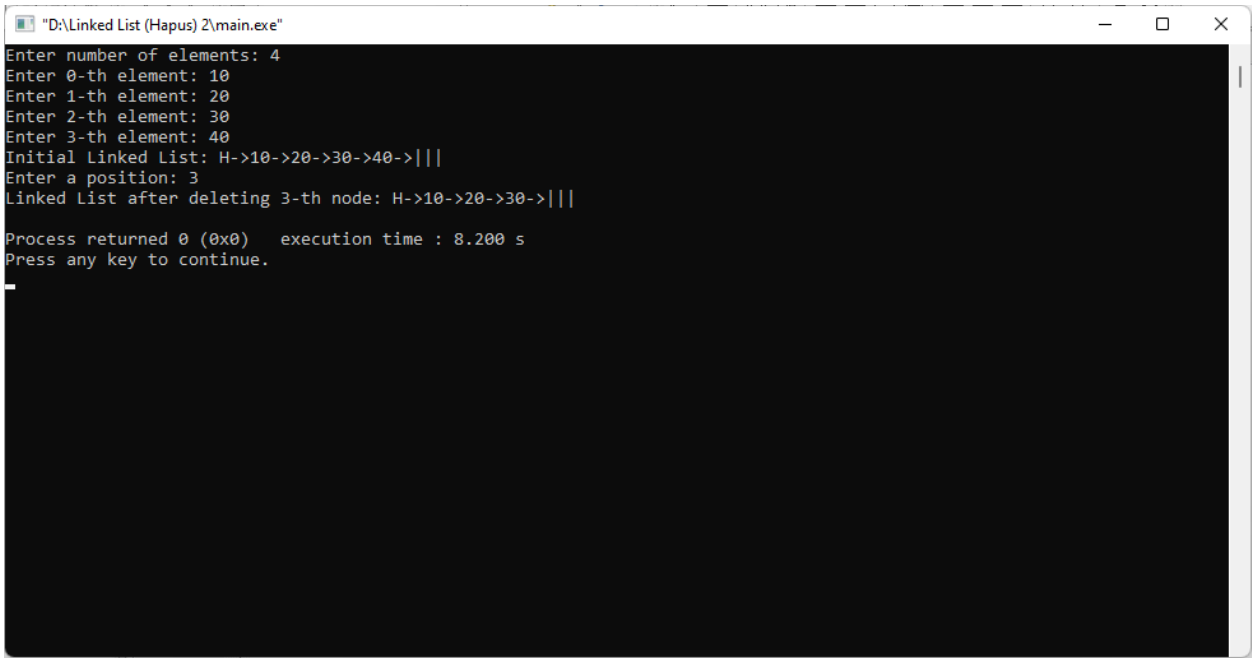
```
     return;
   }
   while(tmp->next) tmp = tmp->next;
   tmp->next = new_node;
}

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 :


"D:\Linked List (Hapus) 2\main.exe"

```
Enter number of elements: 4
Enter 0-th element: 10
Enter 1-th element: 20
Enter 2-th element: 30
Enter 3-th element: 40
Initial Linked List: H->10->20->30->40->|||
Enter a position: 2
Linked List after deleting 2-th node: H->10->20->40->|||

Process returned 0 (0x0)   execution time : 10.433 s
Press any key to continue.
```


"D:\Linked List (Hapus) 2\main.exe"

```
Enter number of elements: 4
Enter 0-th element: 10
Enter 1-th element: 20
Enter 2-th element: 30
Enter 3-th element: 40
Initial Linked List: H->10->20->30->40->|||
Enter a position: 0
Linked List after deleting 0-th node: H->20->30->40->|||

Process returned 0 (0x0)   execution time : 11.483 s
Press any key to continue.
```



```
Enter number of elements: 4
Enter 0-th element: 10
Enter 1-th element: 20
Enter 2-th element: 30
Enter 3-th element: 40
Initial Linked List: H->10->20->30->40->|||
Enter a position: 3
Linked List after deleting 3-th node: H->10->20->30->|||

Process returned 0 (0x0)   execution time : 8.200 s
Press any key to continue.
```

Analisa :

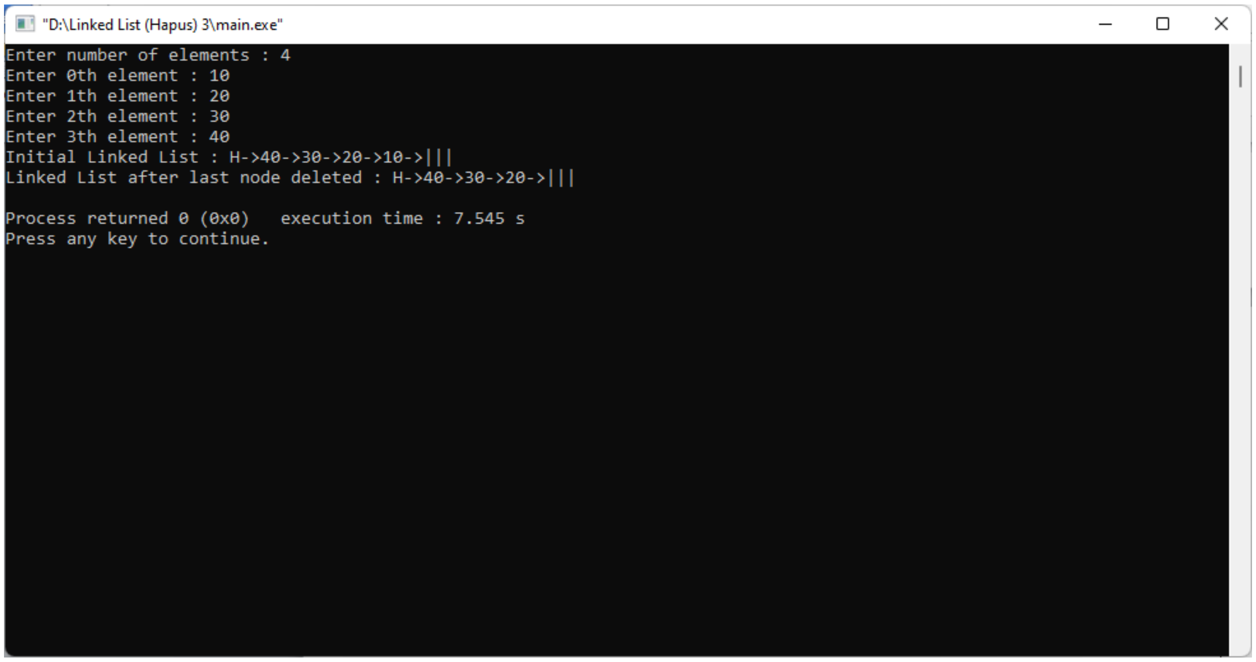
- N-node Deleted digunakan untuk menghapus value di salah satu N-node tersebut.
- N-node Deleted dapat menghapus di awal, di tengah, dan juga di akhir tergantung posisi yang kita inputkan.

➢ 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 :



"D:\Linked List (Hapus) 3\main.exe"

```
Enter number of elements : 4
Enter 0th element : 10
Enter 1th element : 20
Enter 2th element : 30
Enter 3th element : 40
Initial Linked List : H->40->30->20->10->|||
Linked List after last node deleted : H->40->30->20->|||

Process returned 0 (0x0)   execution time : 7.545 s
Press any key to continue.
```

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

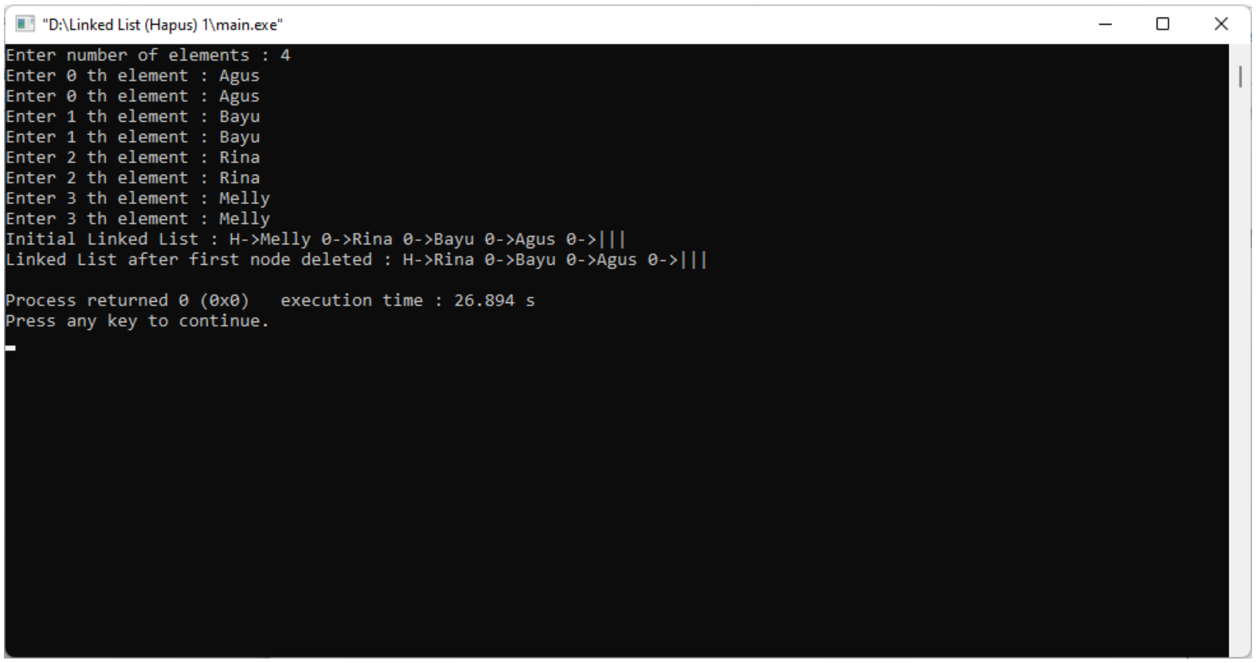
1. Implementasikan 3 algoritma penghapusan node (n node, last node, first node) pada program percobaan yang ada pada modul 2.

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));
```

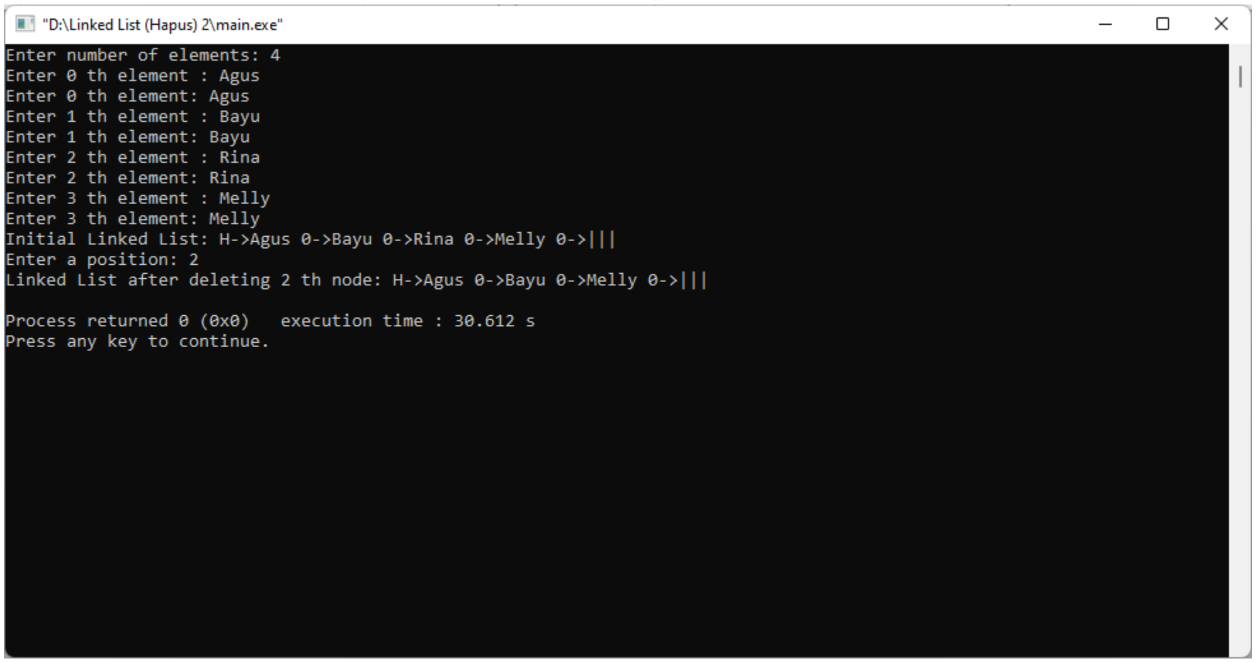
```
    if(new_node == NULL)
    {
        printf("Failed to insert element. Out of memory");
    }
    strcpy(new_node->nama,nama);
    new_node->val = value;
    new_node->next = *head;
    *head = new_node;
}

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 :



```
"D:\Linked List (Hapus) 1\main.exe"
Enter number of elements : 4
Enter 0 th element : Agus
Enter 0 th element : Agus
Enter 1 th element : Bayu
Enter 1 th element : Bayu
Enter 2 th element : Rina
Enter 2 th element : Rina
Enter 3 th element : Melly
Enter 3 th element : Melly
Initial Linked List : H->Melly 0->Rina 0->Bayu 0->Agus 0->|||
Linked List after first node deleted : H->Rina 0->Bayu 0->Agus 0->|||

Process returned 0 (0x0)   execution time : 26.894 s
Press any key to continue.
```

- 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 :


"D:\Linked List (Hapus) 2\main.exe"

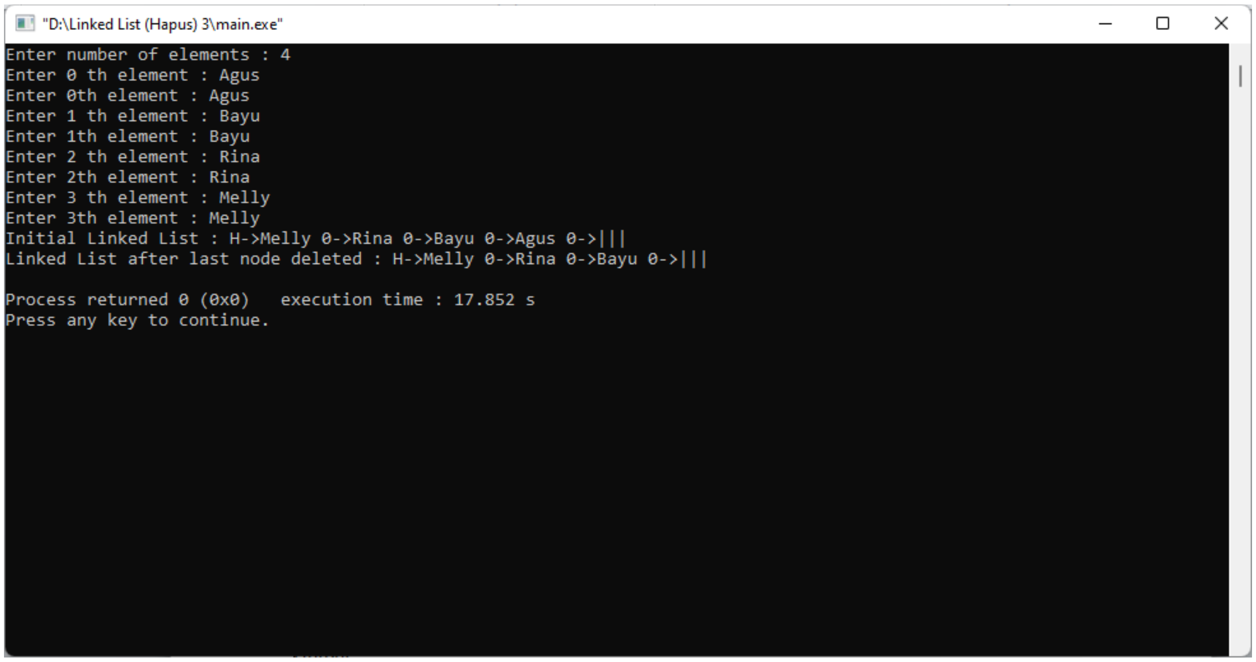
```
Enter number of elements: 4
Enter 0 th element : Agus
Enter 0 th element: Agus
Enter 1 th element : Bayu
Enter 1 th element: Bayu
Enter 2 th element : Rina
Enter 2 th element: Rina
Enter 3 th element : Melly
Enter 3 th element: Melly
Initial Linked List: H->Agus 0->Bayu 0->Rina 0->Melly 0->|||
Enter a position: 2
Linked List after deleting 2 th node: H->Agus 0->Bayu 0->Melly 0->|||

Process returned 0 (0x0)   execution time : 30.612 s
Press any key to continue.
```

- 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 :



"D:\Linked List (Hapus) 3\main.exe"

```
Enter number of elements : 4
Enter 0 th element : Agus
Enter 0th element : Agus
Enter 1 th element : Bayu
Enter 1th element : Bayu
Enter 2 th element : Rina
Enter 2th element : Rina
Enter 3 th element : Melly
Enter 3th element : Melly
Initial Linked List : H->Melly 0->Rina 0->Bayu 0->Agus 0->|||
Linked List after last node deleted : H->Melly 0->Rina 0->Bayu 0->|||

Process returned 0 (0x0)   execution time : 17.852 s
Press any key to continue.
```