Modul 2

Code

```
#include<stdio.h>
#include<stdlib.h>
//Note: printf("\e[1;1H\e[2J"); used to clear the console using regex method
struct node{
    long long int rollNo; //attribute 1
    char name[100]; //attribute 2
    float grade; //attribute 3
    struct node* link;
};
struct node *head = NULL, *tail = NULL;
struct node *display = head;
struct node *inputNode = (struct node *) malloc(sizeof(struct node));
struct node *insert_beg = newNode();
struct node *insert end = newNode();
struct node *insert_given = newNode();
struct node *delete_beg;
struct node *delete_end;
struct node *delete_given;
void header(){
    printf("\t\t\tMade by\n\t\t\tPristian Budi Dharmawan - 2501983105\n");
    printf("\t\t\t\t\t\t\t\tver. 2.03.10");
void displayList(struct node *head){
    struct node *display = head;
    //Validating the list to be displayed
    if(display == NULL){
        printf("404 NOT FOUND\nKindly Input The Data First:)\n\n");
        printf("Press ENTER to continue..."); getchar();
printf("\e[1;1H\e[2J");
    } else{
        printf("\t\t----STUDENT LIST----\n");
        printf("No.\t\t| Name\t\t| Grade\n");
        printf("-----
                                                        ·----\n");
        while(display != NULL){
            printf("%02lld\t\t| %s\t\t| %.2f\n", display->rollNo, display->name,
display->grade);
            display = display->link;
        printf("\n\t\t====This is EOF====\n\n");
        printf("Press ENTER to continue..."); getchar();
printf("\e[1;1H\e[2J");
```

```
void displayData(struct node *head){
    struct node *display = head;
    //Validating the list to be displayed
    if(display == NULL){
       printf("404 NOT FOUND\nKindly Input The Data First:)\n\n");
       printf("Press ENTER to continue..."); getchar();
       printf("\e[1;1H\e[2J");
    } else{
       printf("\t\t----STUDENT LIST----\n");
       printf("No.\t\t\t| Name\t\t| Grade\n");
       printf("------
                                                    -----\n");
       while(display != NULL){
           printf("%0211d\t\t| %s\t\t| %.2f\n", display->rollNo, display->name,
display->grade);
           display = display->link;
    }
//Struct for input new node
struct node *newNode(){
    struct node *inputNode = (struct node *) malloc(sizeof(struct node));
    scanf("%11d", &inputNode->rollNo, printf("Enter Student ID: "));
fflush(stdin);
    scanf("%[^\n]", &inputNode->name, printf("Enter Student Name: "));
fflush(stdin);
    scanf("%f", &inputNode->grade, printf("Enter Student Grades: "));
fflush(stdin);
   inputNode->link = NULL;
   return inputNode;
void insertBeg(){
    struct node *insert beg = newNode();
    //Assigning insert_beg pointer to head
    insert_beg->link = NULL;
   if(head != NULL){
       insert_beg->link = head;
       head = insert beg;
    } else{
       head = insert beg;
       tail = head;
   printf("\n\t\t===Input data Succeeded!===\n\n");
   printf("Press ENTER to continue..."); getchar();
   printf("\e[1;1H\e[2J");
void insertEnd(){
   struct node *insert end = newNode(), *temp;
    //Inverse Package Queue
   insert end->link = 0;
   temp = head;
   while(temp->link != NULL){
       temp = temp->link;
```

```
} temp->link = insert_end;
    printf("\n\t\t===Input data Succeeded!===\n\n");
    printf("Press ENTER to continue..."); getchar();
    printf("\e[1;1H\e[2J");
void insertGiven(){
    int position;
    displayData(head); printf("\n");
    scanf("%d", &position, printf("Position: ")); fflush(stdin);
    struct node *insert given = newNode(), *temp;
    //Connecting Given to Queue
    insert_given->link = 0;
    temp = head;
    for(int i=1; i < position - 1; i++){</pre>
        temp = temp->link;
    } insert given->link = temp->link;
    temp->link = insert given;
    printf("\n\t\t===Input data Succeeded!===\n\n");
    printf("Press ENTER to continue..."); getchar();
    printf("\e[1;1H\e[2J");
void delBeg(){
    struct node *delete_beg;
    //Validating and deleting head
    if(head == NULL){
        printf("404 NOT FOUND\nKindly Input The Data First:)\n\n");
        printf("Press ENTER to continue..."); getchar();
        printf("\e[1;1H\e[2J");
    } else{
        delete_beg = head;
        head = head->link;
        free(delete beg);
        printf("\n\t\t===Delete data Succeeded!===\n\n");
        printf("Press ENTER to continue..."); getchar();
        printf("\e[1;1H\e[2J");
void delEnd(){
    struct node *delete_end, *temp;
    //Validating and deleting tail
    if(head == NULL){
        printf("404 NOT FOUND\nKindly Input The Data First:)\n\n");
        printf("Press ENTER to continue..."); getchar();
        printf("\e[1;1H\e[2J");
    } else{
        delete_end = head;
        while(delete end->link != 0){
```

```
temp = delete end;
            delete_end = delete end->link;
        } free(delete_end);
        temp->link = 0;
       printf("\n\t\t===Delete data Succeeded!===\n\n");
       printf("Press ENTER to continue..."); getchar();
       printf("\e[1;1H\e[2J");
void delGiven(){
    struct node *delete given, *temp = (struct node *) malloc(sizeof(struct
node));
   int position;
   if(head == NULL){
       printf("404 NOT FOUND\nKindly Input The Data First:)\n\n");
       printf("Press ENTER to continue..."); getchar();
       printf("\e[1;1H\e[2J");
    } else{
       displayData(head); printf("\n");
        scanf("%d", &position, printf("Position: ")); fflush(stdin);
        delete given = head;
        //Connecting Given to Queue
        for(int i=1; i < position - 1; i++){</pre>
            delete given = delete given->link;
        } temp = delete_given->link;
        delete given->link = temp->link;
        free(temp);
       printf("\n\t\t===Delete data Succeeded!===\n\n");
       printf("Press ENTER to continue..."); getchar();
       printf("\e[1;1H\e[2J");
int main(){
   int opt;
   printf("\n");
   do{
       header();
       printf("\n\n\t\t=======MENU======\n\n\n");
       printf("Student data organizer program\n");
       printf("----\n"
       printf("1. Display the list\n"); //Requirement 5
       printf("2. Add a node at the beginning\n"); //Requirement 1
       printf("3. Add a node at the end\n"); //Requirement 2
       printf("4. Add a node at given node\n"); //Extra Insertion
       printf("5. Delete a node from the beginning\n"); //Requirement 3
       printf("6. Delete a node from the end\n"); //Requirement 4
       printf("7. Delete a node from a given node\n"); //Extra Deletion
       printf("0. EXIT\n");
       scanf("%d", &opt, printf("Input your choice: ")); fflush(stdin);
       switch (opt){
       case 1:
            printf("\e[1;1H\e[2J");
           displayList(head); break;
```

```
case 2:
        printf("\e[1;1H\e[2J");
        insertBeg(); break;
        printf("\e[1;1H\e[2J");
        insertEnd(); break;
        printf("\e[1;1H\e[2J");
        insertGiven(); break;
    case 5:
        printf("\e[1;1H\e[2J");
        delBeg(); break;
    case 6:
        printf("\e[1;1H\e[2J");
        delEnd(); break;
        printf("\e[1;1H\e[2J");
        delGiven(); break;
    default:
        printf("\e[1;1H\e[2J");
        printf("\n\nThere's no menu no %d\n\n", opt);
} while(opt != 0);
printf("\e[1;1H\e[2J");
printf("Thankyou");
return 0;
```

ScreenShot Hasil

Menu No. 1 (Belum Memiliki Data/List)

```
404 NOT FOUND
Kindly Input The Data First:)
Press ENTER to continue...
```

Menu No. 2

```
Enter Student ID: 2501234567
Enter Student Name: Joko
Enter Student Grades: 3.00

===Input data Succeeded!===

Press ENTER to continue...
```

PRODELIVIS OUTFOI -	EKIVIIIVAL DEBUG CON	SOLL
	-STUDENT LIST	
No.	Name	Grade
2501234567 2501983105	Joko Ian	3.00 4.00
	==This is EOF====	
Press ENTER to cont	inue	

Menu No. 3

```
Enter Student ID: 1234567890
Enter Student Name: Fungi
Enter Student Grades: 3.95

===Input data Succeeded!===
Press ENTER to continue...
```

-			
STUDENT LIST			
No.	Name	Grade	
2501234567	Joko	3.00	
2501983105	Ian	4.00	
1234567890	Fungi	3.95	
	This is EOF	=	
Press ENTER to conti	inue		
	_		

Menu No. 4

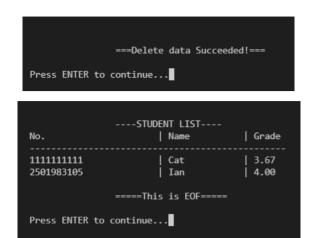
	_		
STUD	ENT LIST		
No.	Name	1	Grade
2501234567	 Joko	٦	3.00
2501983105	Ian	1	4.00
1234567890	Fungi	-1	3.95
Position: 2 Enter Student ID: 1111111111 Enter Student Name: Cat Enter Student Grades: 3.67			
===Input	data Succeeded	!=	
Press ENTER to continue.			

STUDENT LIST No. Name Grade			
NO.	Ivallic	uraue	
2501234567	Joko	3.00	
1111111111	Cat	3.67	
2501983105	Ian	4.00	
1234567890	Fungi	3.95	
=====This is EOF===== Press ENTER to continue			

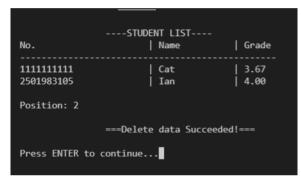
Menu No. 5

===Delete data Succeeded!===
Press ENTER to continue...

	STUDENT LIST	
No.	Name	Grade
1111111111 2501983105 1234567890	Cat Ian Fungi	3.67 4.00 3.95
	==This is EOF====	=
Press ENTER to continue		



Menu No. 7



No.	STUDENT LIST Name	Grade
1111111111	Cat	3.67
	=====This is EOF====	-
Press ENTER to o	ontinue	

Penjelasan Code

Berdasarkan permintaan soal, kami diminta untuk membuat 5 basic requirement dalam program ini. Namun, saya menambahkan 1 extra fungsi insertion dan 1 extra fungsi deletion pada posisi yang diinginkan. Sehingga, saya membuat total 7 menu dalam program saya dan 11 fungsi code.

- Header
 Fungsi ini berfungsi untuk memperindah tampilan setiap program yang saya buat
- Display List
 Fungsi ini berfungsi untuk menampilkan dan mewakili menu nomor 1 dalam program saya.
 Jadi, dalam program ini saya melakukan validasi kemudian print pada bagian logic selection
- 3. Display Data
 Fungsi ini memiliki fungsi yang sama dengan fungsi displayList(node *) hanya saja fungsi ini
 tidak mewakili menu nomor 1. Fungsi ini berguna untuk menampilkan data apa saja yang
 ada dan ditampilkan ketika kita memilih menu nomor 4 dan 7

4. New Node

Fungsi ini berfungsi untuk melakukan input data baru tanpa harus mengetik scanf di setiap fungsi insertion. Sehingga program yang dibuat lebih efisien dan kita hanya perlu untuk memanggil fungsi ini dalam menu insertion

5. Insert Beg

Fungsi ini berfungsi untuk melakukan insertion pada bagian head atau bagian awal dari linked list

6. Insert End

Fungsi ini berfungsi untuk melakukan insertion pada bagian tail melalui algoritma repetition sehingga program akan melakukan traverse sampai akhir kemudian mengisi bagian akhir dari sebuah linked list

7. Insert Given

Fungsi ini memiliki konsep yang sama dengan algoritma repetition pada fungsi insertEnd() hanya saja ketika posisi yang diinginkan sudah sama dengan apa yang diinputkan oleh user maka proses traverse akan berhenti

Del Beg

Fungsi ini berfungsi untuk melakukan deletion pada bagian awal sebuah linked list

9. Del End

Fungsi ini memiliki konsep yang sama dengan algoritma repetition di fungsi insertBeg() dimana program akan melakukan traverse sampai dengan NULL dari sebuah linked list kemudian akan melakukan delete pada bagian akhir sebuah linked list

10. Del Given

Fungsi ini sama dengan fungsi insertGiven() hanya saja berfungsi untuk melakukan deletion

11. Main

Berfungsi untuk mengatur semua fungsi yang sudah dibuat