

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\t\tMade by\n\t\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\t---STUDENT LIST---\n");
        printf("No.\t\t\t\t| Name\t\t\t| Grade\n");
        printf("-----\n");
        while(display != NULL){
            printf("%02lld\t\t\t| %s\t\t\t| %.2f\n", display->rollNo, display->name,
display->grade);
            display = display->link;
        }

        printf("\n\t\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("%02lld\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("%lld", &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++){
        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++){
            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;
        case 3:
            printf("\e[1;1H\e[2J");
            insertEnd(); break;
        case 4:
            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;
        case 7:
            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);
            break;
    }
} while(opt != 0);

printf("\e[1;1H\e[2J");
printf("Thankyou");

return 0;
}

```

ScreenShot Hasil

```

PS D:\SUNIB\B25\SEMESTER 2\COURSES\DATA STRUCTURE\LAB\SESSION 2\Exercises> & .\"DS Module 2.exe"

          Made by
          Pristian Budi Dharmawan - 2501983105
                               Ver. 2.03.10

          =====MENU=====

Student data organizer program
-----
1. Display the list
2. Add a node at the beginning
3. Add a node at the end
4. Add a node at given node
5. Delete a node from the beginning
6. Delete a node from the end
7. Delete a node from a given node
0. EXIT
Input your choice: █

```

Menu No. 1 (Belum Memiliki Data/List)

```

404 NOT FOUND
Kindly Input The Data First:)

Press ENTER to continue...█

```

Menu No. 1 (Memiliki Data/List)

```
-----STUDENT LIST-----
No.           | Name           | Grade
-----|-----|-----
2501983105    | Ian            | 4.00

=====This is EOF=====

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...
```

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
-----STUDENT LIST-----
No.           | Name           | Grade
-----|-----|-----
2501234567    | Joko           | 3.00
2501983105    | Ian            | 4.00

=====This is EOF=====

Press ENTER to continue...
```

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 continue...
```

Menu No. 4

```
-----STUDENT LIST-----
No.      | Name      | Grade
-----|-----|-----
2501234567 | Joko      | 3.00
2501983105 | Ian       | 4.00
1234567890 | Fungi     | 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
-----|-----|-----
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 | Cat       | 3.67
2501983105 | Ian       | 4.00
1234567890 | Fungi     | 3.95

=====This is EOF=====

Press ENTER to continue...|
```

Menu No. 6

```
====Delete data Succeeded!====  
Press ENTER to continue...  
  
-----STUDENT LIST-----  
No.           | Name       | Grade  
-----  
1111111111    | Cat        | 3.67  
2501983105    | Ian        | 4.00  
  
=====This is EOF=====  
Press ENTER to continue...
```

Menu No. 7

```
-----STUDENT LIST-----  
No.           | Name       | Grade  
-----  
1111111111    | Cat        | 3.67  
2501983105    | Ian        | 4.00  
  
Position: 2  
  
====Delete data Succeeded!====  
Press ENTER to continue...  
  
-----STUDENT LIST-----  
No.           | Name       | Grade  
-----  
1111111111    | Cat        | 3.67  
  
=====This is EOF=====  
Press ENTER to continue...
```

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.

1. Header
Fungsi ini berfungsi untuk memperindah tampilan setiap program yang saya buat
2. 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
8. 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