LAPORAN PRAKTIKUM

OTH DOUBLE LINKED LIST

Dibuat oleh

Ananda Bintang Saputra (1203230040)

Fakultas Informatika

Prodi Informatika

Telkom University Surabaya 2023/2024

Source Code

```c
#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>



struct Node

{

   struct Node *prev;

   int data;

   struct Node *next;

};



typedef struct Node node;



node *pHead = NULL;

node *pTail = NULL;



node *alokasiNodeBaru()

{

   node *pNew = NULL;

   pNew = (node *)malloc(sizeof(node));

   return (pNew);
```

```
}


void insert(int data)
{
   node *pNew = alokasiNodeBaru();

   if (pNew == NULL)
   {
       printf("\n[ALOKASI GAGAL]");
   }
   else
   {
       pNew->data = data;
       pNew->prev = NULL;
       pNew->next = NULL;

       if (pHead == NULL)
       {
           pHead = pNew;
           pTail = pNew;
           pHead->next = pHead;
           pHead->prev = pHead;
       }
```

```
        else
        {
            pNew->prev = pTail;
            pNew->next = pHead;
            pTail->next = pNew;
            pHead->prev = pNew;
            pTail = pNew;
        }
    }
}

void view()
{
    node *pWalker = pHead;
    int i = 1;

    if (pWalker == NULL)
    {
        printf("\n[DATA KOSONG]");
    }
    else
    {
        printf("\n");
```

```
        while (pWalker != pTail)
        {
            printf("%d ", pWalker->data);
            i++;
            pWalker = pWalker->next;
        }
        printf("%d ", pWalker->data);
    }
    printf("\n");
}

void sortNode(node *pWalker, node *pWalkerNext)
{
    node *temp = NULL;

    if (pWalker->data > pWalkerNext->data)
    {
        if (pWalker == pHead)
        {
            pHead = pWalkerNext;
        }
        if (pWalkerNext == pTail)
        {
```

```
            pTail = pWalker;
        }

        if (pWalker->prev != NULL)
        {
            pWalker->prev->next = pWalkerNext;
        }
        if (pWalkerNext->next != NULL)
        {
            pWalkerNext->next->prev = pWalker;
        }

        temp = pWalkerNext->next;
        pWalkerNext->next = pWalker;
        pWalkerNext->prev = pWalker->prev;
        pWalker->next = temp;
        pWalker->prev = pWalkerNext;
    }
}

void viewWithAddress()
{
    node *pWalker = pHead;
```

```
    int i = 1;



    if (pWalker == NULL)

    {

        printf("\n[DATA KOSONG]");

    }

    else

    {

        printf("\n");

        while (pWalker != pTail)

        {

            printf("Address: %p | Data: %d\n ", pWalker, pWalker->data);

            i++;

            pWalker = pWalker->next;

        }

        printf("Address: %p | Data: %d\n ", pWalker, pWalker->data);

    }

    printf("\n");

}



int main()

{

    node *pNew = NULL;
```

```
    int numOfData, data;



    printf("Masukkan jumlah data: ");
    scanf("%d", &numOfData);
    for (int i = 0; i < numOfData; i++)
    {
        printf("Masukkan data ke-%d: ", i + 1);
        scanf("%d", &data);
        insert(data);
    }



    printf("\nData awal: ");
    viewWithAddress();
    printf("\nData setelah diurutkan: ");
    sortNode(pHead, pHead->next);
    viewWithAddress();



    return 0;
}
```

PENJELASAN

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

struct Node
{
    struct Node *prev;
    int data;
    struct Node *next;
};

typedef struct Node node;

node *pHead = NULL;
node *pTail = NULL;
```

Inisialisasi awal, import library dan membuat struct yang dibutuhkan, serta membuat 2 variabel yang bisa diakses di seluruh file yaitu head dan tail.

```c
node *alokasiNodeBaru()
{
    node *pNew = NULL;
    pNew = (node *)malloc(sizeof(node));
    return (pNew);
}
```

Membuat function alokasi node baru untuk mengalokasikan memori untuk node baru dalam struktur data double linked list.

```c
void insert(int data)
{
    node *pNew = alokasiNodeBaru();

    if (pNew == NULL)
    {
        printf("\n[ALOKASI GAGAL]");
    }
    else
    {
        pNew->data = data;
        pNew->prev = NULL;
        pNew->next = NULL;

        if (pHead == NULL)
        {
            pHead = pNew;
            pTail = pNew;
            pHead->next = pHead;
            pHead->prev = pHead;
        }
        else
        {
            pNew->prev = pTail;
            pNew->next = pHead;
            pTail->next = pNew;
            pHead->prev = pNew;
            pTail = pNew;
        }
    }
}
```

Membuat prosedur insert untuk memasukkan setiap data yang diinput oleh user kedalam linked list. Didalam prosedur ini, data akan diolah dan dikoneksikan antar nodes menggunakan next dan prev untuk kesinambungan antar node. Dan pada head->prev, karena ini adalah

circular double linked list, maka akan di assign ke tail dan sebaliknya, tail->next akan di assign ke head. Setiap data yang dimasukkan akan berada di setelah data sebelumnya (n+1), jadi menggunakan logika insertAfter.

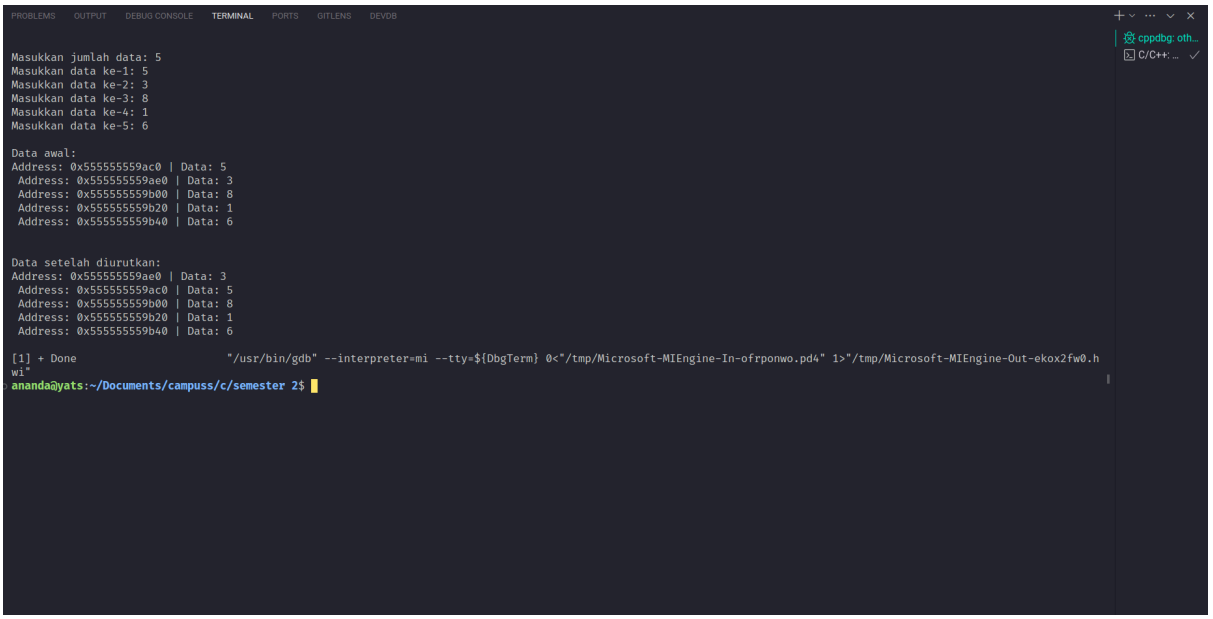
```
void viewWithAddress()
{
    node *pWalker = pHead;
    int i = 1;

    if (pWalker == NULL)
    {
        printf("\n[DATA KOSONG]");
    }
    else
    {
        printf("\n");
        while (pWalker != pTail)
        {
            printf("Address: %p | Data: %d\n ", pWalker, pWalker->data);
            i++;
            pWalker = pWalker->next;
        }
        printf("Address: %p | Data: %d\n ", pWalker, pWalker->data);
    }
    printf("\n");
}
```

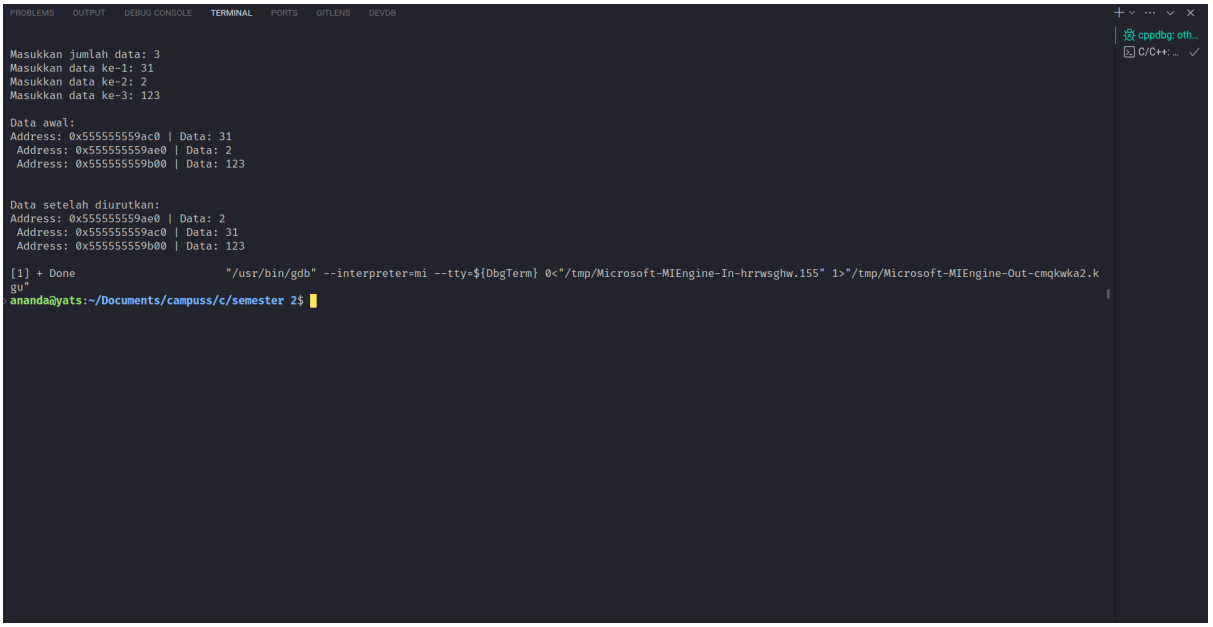
Mengoutput data dengan memanfaatkan head. Karena value dari head sendiri tidak boleh berubah, maka membuat variabel baru yang berfungsi sebagai walker melakukan loop (dalam case ini adalah pWalker). Lalu jika variabel walker tidak sama dengan value tail, maka program akan terus mengoutput walker itu sendiri (yang mana awalnya adalah head, dan head->next, dst hingga tail). Yang dioutput tidak hanya data (value) dari node tersebut, tapi address dari node nya juga.

```
void sortNode(node *pWalker, node *pWalkerNext)
{
    node *temp = NULL;

    if (pWalker->data > pWalkerNext->data)
    {
        if (pWalker == pHead)
        {
            pHead = pWalkerNext;
        }
        if (pWalkerNext == pTail)
        {
            pTail = pWalker;
        }

        if (pWalker->prev != NULL)
        {
            pWalker->prev->next = pWalkerNext;
        }
        if (pWalkerNext->next != NULL)
        {
            pWalkerNext->next->prev = pWalker;
        }

        temp = pWalkerNext->next;
        pWalkerNext->next = pWalker;
        pWalkerNext->prev = pWalker->prev;
        pWalker->next = temp;
        pWalker->prev = pWalkerNext;
    }
}
```

Mengurutkan node, tapi tanpa mengubah data didalam node awal (address tetap). Jika umumnya sorting pada linked list menggunakan variabel temporary untuk menampung sementara data walker sebelumnya, maka untuk sorting ini tanpa menggunakan variabel temporary. Jadi sorting dilakukan dengan mengubah sambungan next dan prev dari setiap node yang dibandingkan.

OUTPUT


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
GITLENS
DEVDB
cppdbg: oth...
C/C++: ...
Masukkan jumlah data: 5
Masukkan data ke-1: 5
Masukkan data ke-2: 3
Masukkan data ke-3: 8
Masukkan data ke-4: 1
Masukkan data ke-5: 6
Data awal:
Address: 0x555555559ac0 | Data: 5
Address: 0x555555559ae0 | Data: 3
Address: 0x555555559b00 | Data: 8
Address: 0x555555559b20 | Data: 1
Address: 0x555555559b40 | Data: 6
Data setelah diurutkan:
Address: 0x555555559ae0 | Data: 3
Address: 0x555555559ac0 | Data: 5
Address: 0x555555559b00 | Data: 8
Address: 0x555555559b20 | Data: 1
Address: 0x555555559b40 | Data: 6
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-ofrponwo.pd4" 1>"/tmp/Microsoft-MIEngine-Out-ekox2fw0.hwi"
anandaayats:~/Documents/campuss/c/semester 2$


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
GITLENS
DEVDB
cppdbg: oth...
C/C++: ...
Masukkan jumlah data: 3
Masukkan data ke-1: 31
Masukkan data ke-2: 2
Masukkan data ke-3: 123
Data awal:
Address: 0x555555559ac0 | Data: 31
Address: 0x555555559ae0 | Data: 2
Address: 0x555555559b00 | Data: 123
Data setelah diurutkan:
Address: 0x555555559ae0 | Data: 2
Address: 0x555555559ac0 | Data: 31
Address: 0x555555559b00 | Data: 123
[1] + Done "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-hrrwsghw.155" 1>"/tmp/Microsoft-MIEngine-Out-cmqkwka2.kgu"
anandaayats:~/Documents/campuss/c/semester 2$