

LAPORAN PRAKTIKUM ASD

OTH Circular Double Linked List

NAMA : Fandi Ardiansyah

KELAS : IF-03-03

NIM : 1203230079

1. Source Code

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

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

struct Node* createNode(int data) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->next = NULL;
    newNode->prev = NULL;
    return newNode;
}

void insertAtBeginning(struct Node** head, int data) {
    struct Node* newNode = createNode(data);
    if (*head == NULL) {
        *head = newNode;
        (*head)->next = *head;
        (*head)->prev = *head;
    } else {
        newNode->next = *head;
        newNode->prev = (*head)->prev;
        (*head)->prev->next = newNode;
        (*head)->prev = newNode;
        *head = newNode;
    }
}

void displayList(struct Node* head) {
    if (head == NULL) {
        printf("List is empty.\n");
        return;
    }
}
```

```

    struct Node* current = head;
    printf("List (memory address & data):\n");
    do {
        printf("(%p, %d) ", current, current->data);
        current = current->next;
    } while (current != head);
    printf("\n");
}

void sortList(struct Node** head) {
    if (*head == NULL) return;

    struct Node* current;
    struct Node* last = (*head)->prev;
    int swapped;

    do {
        swapped = 0;
        current = *head;

        while (current->next != *head) {
            if (current->data > current->next->data) {
                int temp = current->data;
                current->data = current->next->data;
                current->next->data = temp;
                swapped = 1;
            }
            current = current->next;
        }
    } while (swapped);
}

int main() {
    struct Node* head = NULL;
    int N, data;

    printf("Enter the number of data (N): ");
    scanf("%d", &N);

    printf("Enter %d data:\n", N);
    for (int i = 0; i < N; ++i) {
        scanf("%d", &data);
        insertAtBeginning(&head, data);
    }

    printf("List before sorting:\n");
    displayList(head);
}

```

```

sortList(&head);

printf("List after sorting:\n");
displayList(head);

struct Node* current = head;
do {
    struct Node* temp = current;
    current = current->next;
    free(temp);
} while (current != head);

return 0;
}

```

2. Output

```

Enter the number of data (0) : 4
Enter 4 data:
5
5
5
5
4
List before sorting:
List (memory address & data):
(00042300, 4) (00042308, 1) (00042300, 4) (00042308, 3) (00042300, 5) (00042308, 5)
List after sorting:
List (memory address & data):
(00042300, 1) (00042308, 2) (00042300, 5) (00042308, 5) (00042300, 4) (00042308, 0)
PS C:\Users\aridya\OneDrive\Documents\ALSTHURDAT> ./P18.exe
Enter the number of data (0) : 4
Enter 4 data:
2
10
-3
123
List before sorting:
List (memory address & data):
(00001300, 123) (00001304, 2) (00001308, 10) (0000130C, 3)
List after sorting:
List (memory address & data):
(00001300, 2) (00001308, 3) (0000130C, 123) (00001310, 123)

```

3. Penjelasan

1. `#include <stdio.h>`: Mengimpor pustaka standar untuk input dan output.
2. `#include <stdlib.h>`: Mengimpor pustaka standar untuk fungsi alokasi memori dinamis (`'malloc'`, `'free'`).

3-6. Definisi struktur ``Node``: Struktur untuk list ganda melingkar, dengan ``data`` menyimpan nilai, dan ``next`` dan ``prev`` adalah pointer ke node berikutnya dan sebelumnya.

8-13. ``createNode``: Fungsi untuk membuat node baru. Menerima data dan mengalokasikan memori untuk node baru, serta menginisialisasi ``next`` dan ``prev`` ke ``NULL``.

15-28. `insertAtBeginning`: Fungsi untuk menambah node di awal list. Jika list kosong, node baru menunjuk ke dirinya sendiri untuk ``next`` dan ``prev``. Jika tidak, node baru dihubungkan dengan benar ke node yang ada.

30-41. `displayList`: Fungsi untuk menampilkan isi list. Memeriksa jika list kosong, lalu mencetak alamat memori dan data setiap node dengan iterasi melalui list melingkar.

43-58. `sortList`: Fungsi untuk mengurutkan list dengan bubble sort. Menggunakan do-while loop untuk melakukan iterasi sampai tidak ada lagi pertukaran (swapped). Data node bertukar jika tidak dalam urutan yang benar.

60-83. `main`: Fungsi utama program.

- Mendeklarasikan pointer ``head`` dan variabel ``N`` untuk jumlah data.
- Meminta input jumlah data.
- Meminta input data dan memasukkan setiap nilai ke dalam list dengan ``insertAtBeginning``.
- Menampilkan list sebelum pengurutan.
- Mengurutkan list dengan ``sortList``.
- Menampilkan list setelah pengurutan.
- Membebaskan memori yang dialokasikan untuk list dengan mengiterasi dan menghapus setiap node.