Nama: Richie

Nim: 1203230064

Kelas: IF-03-01

Tugas Circular Double Linked List

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

Int data: berfungsi untuk menyimpan data dalam node.

Struct Node* next: berfungsi untuk menunjuk ke node berikutnya.

Struct Node* prev: berfungsi untuk menunjuk ke node sebelumnya.

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

Node* newnode = (node*)malloc (sizeof(Node)): berfungsi untuk mengalokasikan memori untuk node baru.

newNode->data = data: berfungsi untuk mengatur nilai data dari node baru menjadi data.

newNode->next = newNode; newNode->prev = newNode;: berfungsi untuk mengatur pointer next dan prev dari node baru untuk menunjuk ke node itu sendiri.

return newNode;: berfungsi untuk mengembalikan pointer ke node baru yang telah dibuat.

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

Node* newNode = createNode(data): berfungsi untuk membuat node baru dengan nilai data yang diberikan.

if (*head == NULL) { *head = newNode; }: Jika list kosong, maka node baru menjadi elemen pertama dalam list.

else: Jika list tidak kosong, maka node baru ditambahkan sebelum elemen pertama dalam list.

```
void printList(Node* head) {
    if (head == NULL) return;
    Node* temp = head;
    do {
        printf("Address: %p, Data: %d\n", (void*)temp, temp->data);
        temp = temp->next;
    } while (temp != head);
    printf("\n");
}
```

if (head == NULL) return;: Jika list kosong, maka fungsi ini tidak melakukan apa-apa dan segera kembali.

Node* temp = head;: Ini mendefinisikan pointer sementara temp dan menginisialisasinya dengan head.

Printf: berfungsi untuk mencetak alamat memori dari node saat ini (temp) dan nilai data dari node tersebut.

temp = temp->next;: berfungsi untuk memindahkan temp ke node berikutnya dalam list

while (temp != head);: Ini adalah loop yang berjalan selama kita belum kembali ke elemen pertama dalam list.

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

    int swapped;
    Node* ptr1;
    Node* lptr = NULL;

    do {
        swapped = 0;
        ptr1 = *head;

        while (ptr1->next != *head) {
            if (ptr1->data > ptr1->next->data) {
                // Swap nodes, not just data
                Node* temp = ptr1->next;
                ptr1->next = temp->next;
                temp->next->prev = ptr1;
                temp->prev = ptr1->prev;
                ptr1->prev->next = temp;
                ptr1->prev = temp;
                temp->next = ptr1;

                if (*head == ptr1) {
```

```
                    *head = temp;
                }

                swapped = 1;
            } else {
                ptr1 = ptr1->next;
            }
        }
        lptr = ptr1;
    } while (swapped);
}
```

if (*head == NULL) return;: Jika list kosong, maka fungsi ini tidak melakukan apa-apa dan segera kembali.

while (swapped);: Ini adalah loop yang berjalan selama ada setidaknya satu pertukaran yang dilakukan dalam satu iterasi penuh dari list.

if (ptr1->data > ptr1->next->data): Jika data dalam node saat ini lebih besar dari data dalam node berikutnya, maka kedua node tersebut ditukar.

else { ptr1 = ptr1->next; }: Jika data dalam node saat ini tidak lebih besar dari data dalam node berikutnya, maka kita pindah ke node berikutnya dalam list.

Output:

```
5
5
3
8
1
6
List sebelum pengurutan:
Address: 00781598, Data: 5
Address: 007815B0, Data: 3
Address: 007815C8, Data: 8
Address: 007815F0, Data: 1
Address: 00781608, Data: 6

List setelah pengurutan:
Address: 007815F0, Data: 1
Address: 007815B0, Data: 3
Address: 00781598, Data: 5
Address: 00781608, Data: 6
Address: 007815C8, Data: 8
```

```
3
31
2
123
List sebelum pengurutan:
Address: 00CB1598, Data: 31
Address: 00CB15B0, Data: 2
Address: 00CB15C8, Data: 123

List setelah pengurutan:
Address: 00CB15B0, Data: 2
Address: 00CB1598, Data: 31
Address: 00CB15C8, Data: 123
```