TUGAS OTH

M. DAUD ABDUL AZIZ
1203230006

1. 
```c
typedef struct Node {
    int data;
    struct Node* next;
    struct Node* prev;
} Node;
```

- **Node** adalah struktur data yang menyimpan **data** dan dua pointer, **next** dan **prev**, yang menunjuk ke node berikutnya dan sebelumnya dalam list.

2. 
```c
Node* createNode(int data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = data;
    newNode->next = newNode->prev = newNode;
    return newNode;
}
```

- Fungsi ini membuat node baru dengan **data** yang diberikan.
- **newNode->next** dan **newNode->prev** diatur untuk menunjuk ke dirinya sendiri karena node ini akan menjadi node tunggal dalam list sirkular jika ditambahkan ke list kosong.

3. 
```c
void append(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;
    }
}
```

- Fungsi ini menambahkan node baru di akhir list.

- Jika list kosong (***head == NULL**), node baru menjadi satu-satunya node di list.
- Jika list tidak kosong, node baru ditempatkan di antara node terakhir (**last**) dan node pertama (***head**), dengan memperbarui pointer **next** dan **prev** sesuai kebutuhan untuk menjaga sirkularitas.

4.
```c
void printList(Node* head) {
    Node* temp = head;
    if (head != NULL) {
        do {
            printf("Address: %p, Data: %d\n", (void*)temp, temp->data);
            temp = temp->next;
        } while (temp != head);
    }
}
```

- Fungsi ini mencetak semua node dalam list.
- Jika **head** tidak **NULL**, ia memulai dari **head** dan mencetak data setiap node, kemudian berpindah ke node berikutnya (**temp = temp->next**) hingga kembali ke **head**.

5.
```c
void sortList(Node** head) {
    if (*head == NULL) return;
    int swapped;
    Node* ptr1;
    Node* lptr = NULL;

    do {
        swapped = 0;
        ptr1 = *head;

        while (ptr1->next != lptr) {
            if (ptr1->data > ptr1->next->data) {
                int tempData = ptr1->data;
                ptr1->data = ptr1->next->data;
                ptr1->next->data = tempData;
                swapped = 1;
            }
            ptr1 = ptr1->next;
        }
        lptr = ptr1;
    } while (swapped);
}
```

- Fungsi ini mengurutkan list menggunakan metode bubble sort.
- Jika ***head** adalah **NULL**, fungsi langsung keluar.
- Proses berulang hingga tidak ada swap lagi (**swapped = 0**).
- Untuk setiap node, jika **data** dari node tersebut lebih besar dari **data** node berikutnya, data ditukar, dan **swapped** diatur menjadi 1 untuk menandakan bahwa ada perubahan.

6.
```c
int main() {
    int N;
    scanf("%d", &N);

    Node* head = NULL;

    for (int i = 0; i < N; i++) {
        int data;
        scanf("%d", &data);
        append(&head, data);
    }

    printf("Before sorting:\n");
    printList(head);

    sortList(&head);

    printf("After sorting:\n");
    printList(head);

    return 0;
}
```

- Fungsi **main** membaca jumlah node (**N**) yang akan ditambahkan ke list.
- Menggunakan loop, membaca **data** dari input dan menambahkan node dengan **data** tersebut ke list menggunakan **append**.
- Mencetak list sebelum pengurutan, mengurutkan list menggunakan **sortList**, dan kemudian mencetak list setelah pengurutan.

INPUT :

```
PS C:\Users\david louis\OneDrive\Documents\ASD\OTH>
5
5
3
8
1
6
```

OUTPUT :

```
Before sorting:
Address: 00C71678, Data: 5
Address: 00C71690, Data: 3
Address: 00C716A8, Data: 8
Address: 00C724F0, Data: 1
Address: 00C72508, Data: 6
```

INPUT :

```
PS C:\Users\david louis\OneDrive\Documents\ASD\OTH>
3
31
2
123
```

OUTPUT :

```
Before sorting:
Address: 00771678, Data: 31
Address: 00771690, Data: 2
Address: 007716A8, Data: 123
```