NAMA : I MADE SUTA EKA DHARMA

KELAS : IF 03-01

NIM : 1203230072

TUGAS DOUBLE LINKED LIST CIRCULAR

SOURCE CODE

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

// Definisi struktur node
typedef struct Node *Node;
struct Node {
    int data;
    Node prev, next;
};

Node first = NULL, last = NULL; // Variabel global untuk node pertama
dan terakhir

// Fungsi untuk membuat node baru
Node createNode(int data) {
    Node newNode = (Node)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->prev = newNode->next = newNode;
    return newNode;
}

// Fungsi untuk menyisipkan node di akhir list
void insertLast(int data) {
    Node newNode = createNode(data);

    if (first == NULL) {
        first = last = newNode;
    } else {
        newNode->prev = last;
        last->next = newNode;
        newNode->next = first;
        first->prev = newNode;
        last = newNode;
    }
}

// Fungsi untuk menampilkan list
void printList() {
    if (first == NULL) {
```

```
        printf("[]\n\n");
    } else {
        Node temp = first;
        do {
            printf("Address: %p, Data: %d\n", (void*)temp, temp->data);
            temp = temp->next;
        } while (temp != first);
        printf("\n\n");
    }
}

void swap(Node a, Node b) {
  Node aPrev = a->prev;
  Node aNext = a->next;
  Node bPrev = b->prev;
  Node bNext = b->next;

  a->next = bNext;
  a->prev = b;
  b->next = a;
  b->prev = aPrev;
  aPrev->next = b;
  bNext->prev = a;

  // tukar list pertama
  if (a == first) first = b;
  else if (b == first) first = a;

  // tukar list terakhir
  if (a == last) last = b;
  else if (b == last) last = a;
}

void sortList() {
  if (first == NULL || first == first->next) return;

  int repeat;
  Node node;

  do {
    repeat = 0;
    node = first;

    while (node->next != first) {
      if (node->data > node->next->data) {
```

```
        swap(node, node->next);

        repeat = 1;
      } else {

        node = node->next;
      }
    }
  } while (repeat);
}

int main() {
    int total, nilai;

    printf("Masukkan Total: ");
    scanf("%d", &total);

    printf("Masukkan Nilai: ");
    for (int i = 0; i < total; i++){
        scanf("%d", &nilai);
        insertLast(nilai);
    }

    printf("List sebelum pengurutan:\n");
    printList();

    sortList();

    printf("List setelah pengurutan:\n");
    printList();

    return 0;
}
```

OUTPUT

```
Masukkan Total: 5
Masukkan Nilai: 5 8 3 1 6
List sebelum pengurutan:
Address: 00AA15F0, Data: 5
Address: 00AA1598, Data: 8
Address: 00AA15B0, Data: 3
Address: 00AA15C8, Data: 1
Address: 00AA23A8, Data: 6


List setelah pengurutan:
Address: 00AA15C8, Data: 1
Address: 00AA15B0, Data: 3
Address: 00AA15F0, Data: 5
Address: 00AA23A8, Data: 6
Address: 00AA1598, Data: 8
```

PENJELASAN

```
// Definisi struktur node
typedef struct Node *Node;
struct Node {
    int data;
    Node prev, next;
};
```

- Mendefinisikan tipe data Node sebagai pointer ke struct Node yang memiliki tiga anggota: data untuk menyimpan nilai integer, prev untuk menunjuk ke node sebelumnya, dan next untuk menunjuk ke node berikutnya.

```
// Fungsi untuk membuat node baru
Node createNode(int data) {
    Node newNode = (Node)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->prev = newNode->next = newNode;
    return newNode;
}
```

- Membuat node baru dengan mengalokasikan memori dan menginisialisasi data serta pointer prev dan next agar menunjuk pada node itu sendiri.

```c
// Fungsi untuk menyisipkan node di akhir list
void insertLast(int data) {
    Node newNode = createNode(data);

    if (first == NULL) {
        first = last = newNode;
    } else {
        newNode->prev = last;
        last->next = newNode;
        newNode->next = first;
        first->prev = newNode;
        last = newNode;
    }
}
```

- Menambahkan node baru di akhir linked list. Jika list kosong, node baru menjadi node pertama dan terakhir. Jika tidak, menghubungkan node baru ke node terakhir dan memperbarui pointer prev dan next.

```c
// Fungsi untuk menampilkan list
void printList() {
    if (first == NULL) {
        printf("[]\n\n");
    } else {
        Node temp = first;
        do {
            printf("Address: %p, Data: %d\n", (void*)temp, temp->data);
            temp = temp->next;
        } while (temp != first);
        printf("\n\n");
    }
}
```

- Menampilkan seluruh node dalam linked list. Jika list kosong, menampilkan "[]". Jika tidak, mencetak alamat dan data setiap node sampai kembali ke node pertama.

```
void swap(Node a, Node b) {
  Node aPrev = a->prev;
  Node aNext = a->next;
  Node bPrev = b->prev;
  Node bNext = b->next;

  a->next = bNext;
  a->prev = b;
  b->next = a;
  b->prev = aPrev;
  aPrev->next = b;
  bNext->prev = a;

  // tukar list pertama
  if (a == first) first = b;
  else if (b == first) first = a;

  // tukar list terakhir
  if (a == last) last = b;
  else if (b == last) last = a;
}
```

- Menukar dua node dalam list dengan memperbarui pointer prev dan next dari node terkait. Juga memperbarui pointer first dan last jika salah satu node yang ditukar adalah node pertama atau terakhir.

```
void sortList() {
  if (first == NULL || first == first->next) return;

  int repeat;
  Node node;

  do {
    repeat = 0;
    node = first;

    while (node->next != first) {
      if (node->data > node->next->data) {

        swap(node, node->next);

        repeat = 1;
      } else {

        node = node->next;
      }
    }
  } while (repeat);
}
```

- Mengurutkan linked list menggunakan metode bubble sort. Mengecek setiap node dan menukar jika data node lebih besar dari data node berikutnya, diulang sampai tidak ada lagi node yang perlu ditukar.

```c
int main() {
    int total, nilai;

    printf("Masukkan Total: ");
    scanf("%d", &total);

    printf("Masukkan Nilai: ");
    for (int i = 0; i < total; i++){
        scanf("%d", &nilai);
        insertLast(nilai);
    }

    printf("List sebelum pengurutan:\n");
    printList();

    sortList();

    printf("List setelah pengurutan:\n");
    printList();

    return 0;
}
```

- Fungsi utama untuk membaca jumlah node dan data node dari pengguna, memasukkan node ke linked list, menampilkan list sebelum dan sesudah pengurutan