NAMA : FITRIA NUR ROFIKA

NIM : 1203230097

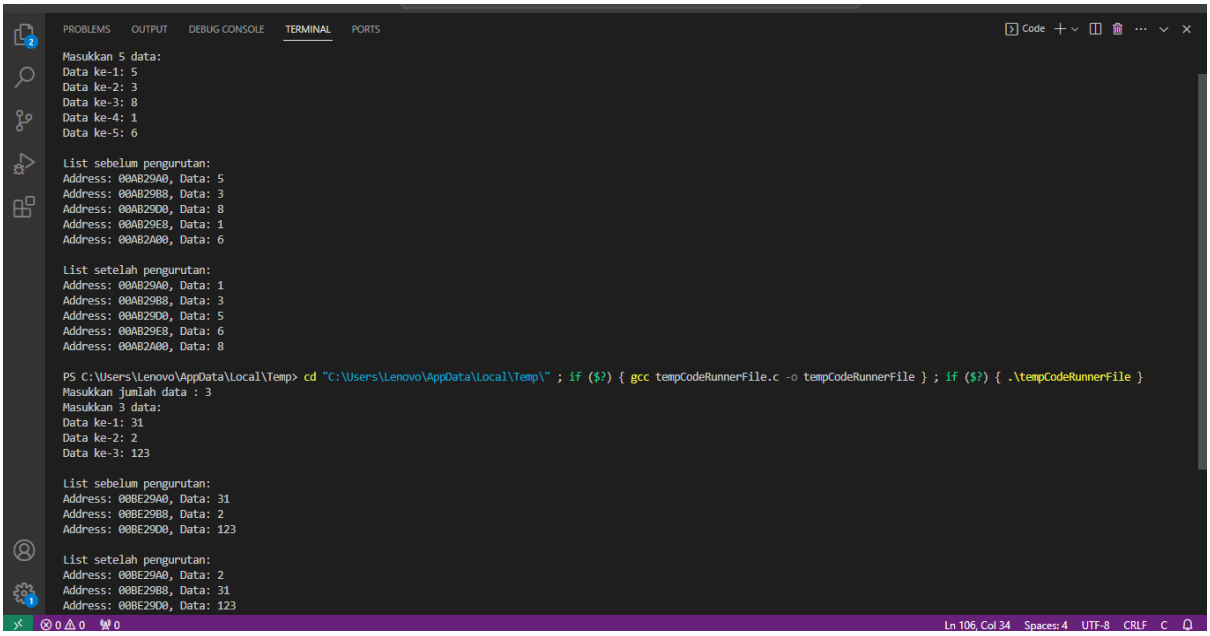
KELAS : IF-03-02

1.

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

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

// Fungsi untuk membuat node baru
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;
}

// Fungsi untuk menambahkan node baru ke dalam list
void insertNode(struct Node** head, int data) {
    struct Node* newNode = createNode(data);
    if (*head == NULL) {
        *head = newNode;
        (*head)->next = *head;
        (*head)->prev = *head;
    } else {
        struct Node* last = (*head)->prev;
        last->next = newNode;
        newNode->prev = last;
        newNode->next = *head;
        (*head)->prev = newNode;
    }
}

// Fungsi untuk mengurutkan list secara ascending
void sortList(struct Node** head) {
    if (*head == NULL) return;

    struct Node *current = *head, *index = NULL;
    int temp;
```

```c
    do {
        index = current->next;

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

        current = current->next;
    } while (current != *head);
}

// Fungsi untuk menampilkan list beserta alamat memori dan data
void displayListWithAddress(struct Node* head) {
    if (head == NULL) return;

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

// Fungsi untuk menghapus seluruh list
void deleteList(struct Node** head) {
    if (*head == NULL) return;

    struct Node *current = *head, *temp = NULL;

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

    *head = NULL;
}

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

    printf("Masukkan jumlah data : ");
```

```c
    scanf("%d", &n);

    if (n < 1 || n > 10) {
        printf("Jumlah data tidak valid.\n");
        return 1;
    }

    printf("Masukkan %d data:\n", n);
    for (int i = 0; i < n; i++) {
        printf("Data ke-%d: ", i + 1);
        scanf("%d", &data);
        insertNode(&head, data);
    }

    printf("\nList sebelum pengurutan:\n");
    displayListWithAddress(head);

    sortList(&head);

    printf("List setelah pengurutan:\n");
    displayListWithAddress(head);

    deleteList(&head);

    return 0;
}
```



PENJELASAN

- Struktur Node mendefinisikan struktur Node dengan integer data, pointer next ke node berikutnya, dan pointer prev ke node sebelumnya.

- createNode Function membuat node baru dengan alokasi memori untuk struct Node, menginisialisasi data, next, dan prev, lalu mengembalikan pointer ke node baru tersebut.

- insertNode Function untuk menambahkan node baru ke dalam circular doubly linked list.

  - createNode digunakan untuk membuat node baru

  - Jika head adalah NULL (list kosong), node baru menjadi head dan menunjuk dirinya sendiri untuk next dan prev.

  - Jika list tidak kosong, node baru ditambahkan di akhir list. Node terakhir diakses dengan (*head)->prev. Node baru dihubungkan dengan last dan head, serta head diperbarui untuk menunjuk ke node baru sebagai node terakhir.

- sortList Function mengurutkan node dalam list secara ascending dengan metode bubble sort.

  - Jika head adalah NULL, fungsi langsung mengembalikan.

  - Dua pointer digunakan current untuk node yang sedang diperiksa dan index untuk node berikutnya.

  - Jika current->data lebih besar dari index->data, data ditukar. Proses berlanjut sampai current mencapai head lagi, memastikan semua node sudah diurutkan.

- displayListWithAddress Function menampilkan list beserta alamat memori dan data.

  - Jika head adalah NULL, fungsi langsung mengembalikan.

  - Pointer temp digunakan untuk melintasi list dan mencetak alamat memori (%p untuk pointer) dan data (%d untuk integer) setiap node sampai kembali ke head.

- deleteList Function menghapus seluruh list dan membebaskan memori yang dialokasikan.

  - Jika head adalah NULL, fungsi langsung mengembalikan.

  - Pointer current digunakan untuk melintasi list dan membebaskan setiap node dengan free sampai kembali ke head.

  - Setelah semua node dihapus, head diatur ke NULL.

- int main

  - Mendeklarasikan pointer head ke NULL dan variabel untuk jumlah data (n) dan data (data).

  - Membaca jumlah data dari pengguna dan memvalidasi bahwa n antara 1 dan 10.

  - Jika jumlah data tidak valid, program mengembalikan nilai 1.

  - Membaca data dari pengguna dan memasukkannya ke dalam list dengan insertNode.

  - Menampilkan list sebelum pengurutan dengan displayListWithAddress.

  - Mengurutkan list dengan sortList.

- Menampilkan list setelah pengurutan dengan displayListWithAddress.

- Menghapus seluruh list dengan deleteList.