OTH

SOURCE CODE

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

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

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

- Fungsi ini bertanggung jawab untuk membuat node baru dalam linked list dengan nilai data yang diberikan.
- Pertama, ia mengalokasikan memori untuk node baru menggunakan `malloc`.
- Kemudian, ia menginisialisasi nilai data dari node tersebut dengan nilai yang diberikan.
- Pointer `prev` dan `next` dari node baru diatur untuk menunjuk ke node itu sendiri karena pada awalnya, node baru akan menjadi satu-satunya node dalam linked list.
- Akhirnya, ia mengembalikan pointer ke node yang baru saja dibuat.

```c
void insertEnd(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 digunakan untuk memasukkan node baru ke akhir linked list.

- Jika linked list masih kosong (head adalah NULL), maka node baru akan menjadi head.
- Jika tidak, maka ia akan menemukan node terakhir dengan mengakses `prev` dari head.
- Kemudian, ia akan memperbarui pointer `next` dan `prev` untuk menghubungkan node baru dengan node terakhir dan head.
- Akhirnya, ia akan memperbarui head jika diperlukan.

```
void sortList(Node** head) {
    if (*head == NULL || (*head)->next == *head) return;

    Node* current = *head;
    do {
        Node* nextNode = current->next;
        while (nextNode != *head) {
            if (current->data > nextNode->data) {
                int temp = current->data;
                current->data = nextNode->data;
                nextNode->data = temp;
            }
            nextNode = nextNode->next;
        }
        current = current->next;
    } while (current->next != *head);
}
```

1.

- Fungsi ini bertanggung jawab untuk mengurutkan linked list berdasarkan nilai data pada setiap node.
- Ini menggunakan algoritma pengurutan bubble sort.
- Fungsi ini akan berhenti jika linked list kosong atau hanya memiliki satu node.
- Pertama, ia akan memilih node pertama sebagai current dan melanjutkan ke node berikutnya dalam setiap iterasi.
- Selama iterasi, ia membandingkan nilai data current dengan nilai data dari setiap node berikutnya. Jika nilai data current lebih besar dari nilai data node berikutnya, maka mereka akan ditukar.
- Proses ini akan berlanjut hingga seluruh linked list diurutkan.

```
void displayList(Node* head) {
```

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

1.

- Fungsi ini bertanggung jawab untuk menampilkan isi dari linked list ke layar.
- Jika linked list kosong, maka akan mencetak pesan bahwa linked list kosong.
- Jika tidak, ia akan memulai dari head dan mencetak alamat dan nilai data dari setiap node secara berurutan sampai kembali ke head lagi.
- Proses ini akan berhenti ketika seluruh linked list telah dicetak.

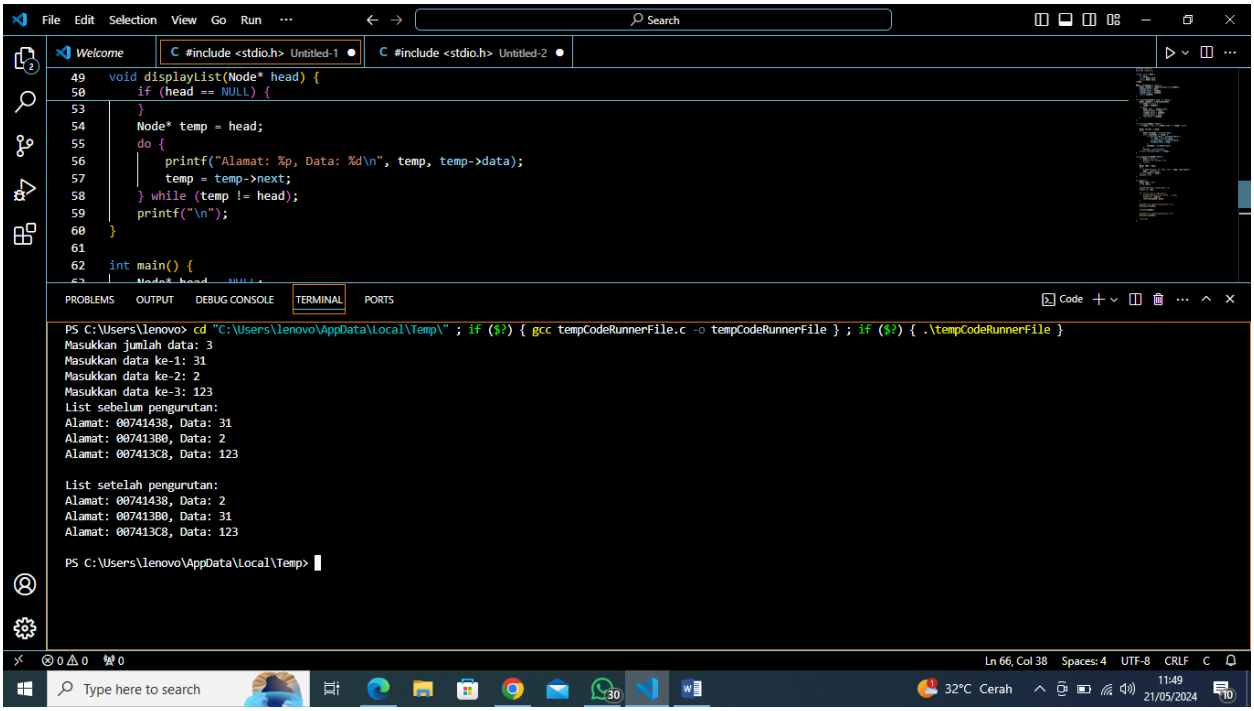
```
int main() {
    Node* head = NULL;
    int N, data;

    printf("Masukkan jumlah data: ");
    scanf("%d", &N);

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

    printf("List sebelum pengurutan:\n");
    displayList(head);

    sortList(&head);

    printf("List setelah pengurutan:\n");
    displayList(head);
    return 0;
```

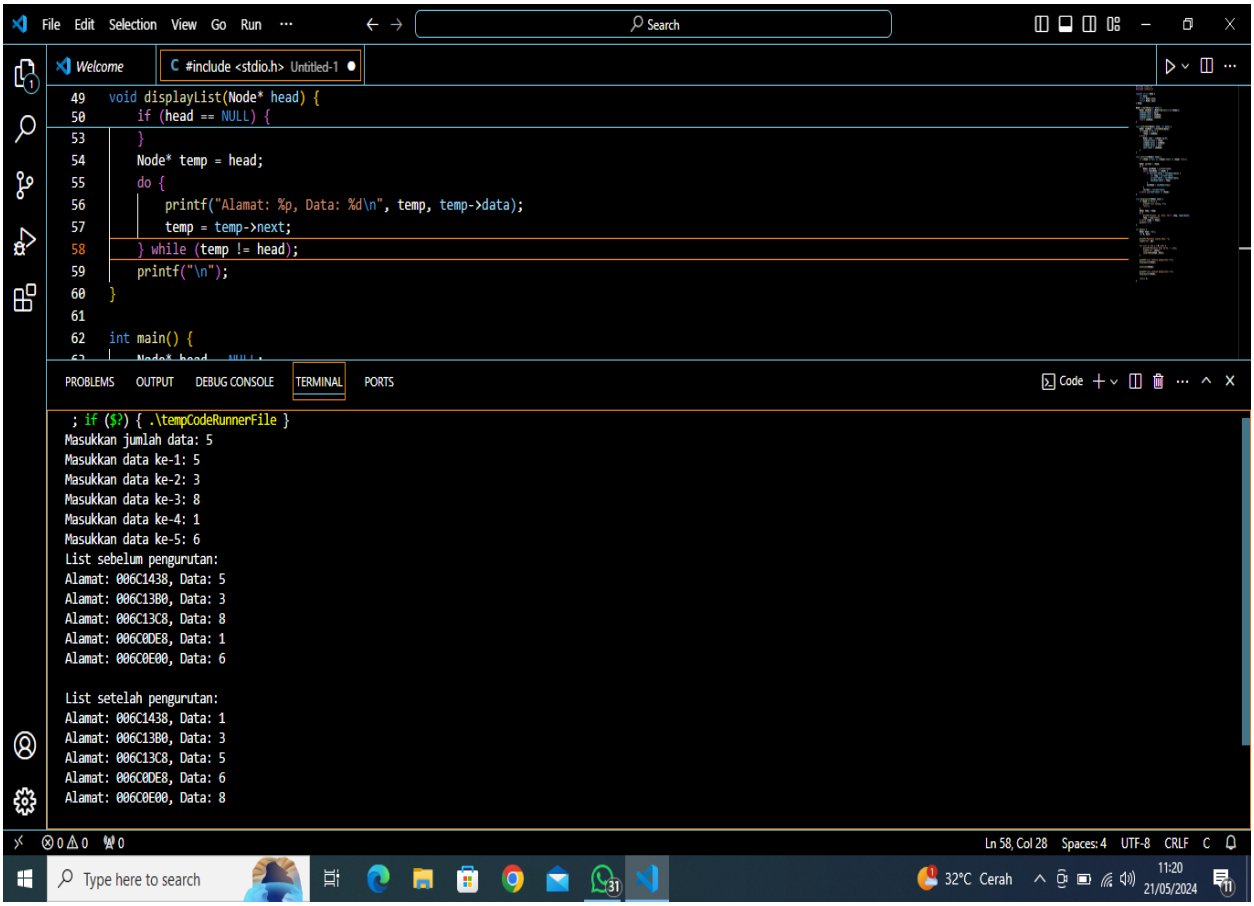
```
}
```

1.

- Fungsi **`main`** adalah fungsi utama dari program.
- Di awal, ia mendeklarasikan pointer **`head`** yang mengacu pada head dari linked list dan variabel **`N`** dan **`data`**.
- Kemudian, ia meminta pengguna untuk memasukkan jumlah data yang ingin dimasukkan ke dalam linked list.
- Setelah itu, ia membaca nilai-nilai data dari pengguna dan memasukkannya ke dalam linked list menggunakan fungsi **`insertEnd`**.
- Setelah linked list diisi, ia menampilkan linked list sebelum diurutkan menggunakan fungsi **`displayList`**.
- Kemudian, ia mengurutkan linked list menggunakan fungsi **`sortList`**.
- Akhirnya, ia menampilkan linked list setelah diurutkan menggunakan fungsi **`displayList`** lagi.

NAMA : GALANG UBAIDILLAH
NIM : 1203230056
KELAS : IF 03-03


File Edit Selection View Go Run
Search
Welcome
#include <stdio.h> Untitled-1
#include <stdio.h> Untitled-2
void displayList(Node* head) {
if (head == NULL) {
}
Node* temp = head;
do {
printf("Alamat: %p, Data: %d\n", temp, temp->data);
temp = temp->next;
} while (temp != head);
printf("\n");
}
int main() {
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code
PS C:\Users\lenovo> cd "C:\Users\lenovo\AppData\Local\Temp\" ; if ($?) { gcc tempCodeRunnerFile.c -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Masukkan jumlah data: 3
Masukkan data ke-1: 31
Masukkan data ke-2: 2
Masukkan data ke-3: 123
List sebelum pengurutan:
Alamat: 00741438, Data: 31
Alamat: 007413B0, Data: 2
Alamat: 007413C8, Data: 123
List setelah pengurutan:
Alamat: 00741438, Data: 2
Alamat: 007413B0, Data: 31
Alamat: 007413C8, Data: 123
PS C:\Users\lenovo\AppData\Local\Temp>
Ln 66, Col 38 Spaces: 4 UTF-8 CRLF C
Type here to search
32°C Cerah
11:49
21/05/2024


File Edit Selection View Go Run
Search
Welcome
#include <stdio.h> Untitled-1
void displayList(Node* head) {
if (head == NULL) {
}
Node* temp = head;
do {
printf("Alamat: %p, Data: %d\n", temp, temp->data);
temp = temp->next;
} while (temp != head);
printf("\n");
}
int main() {
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Code
; if ($?) { .\tempCodeRunnerFile }
Masukkan jumlah data: 5
Masukkan data ke-1: 5
Masukkan data ke-2: 3
Masukkan data ke-3: 8
Masukkan data ke-4: 1
Masukkan data ke-5: 6
List sebelum pengurutan:
Alamat: 006C1438, Data: 5
Alamat: 006C13B0, Data: 3
Alamat: 006C13C8, Data: 8
Alamat: 006C0DE8, Data: 1
Alamat: 006C0E00, Data: 6
List setelah pengurutan:
Alamat: 006C1438, Data: 1
Alamat: 006C13B0, Data: 3
Alamat: 006C13C8, Data: 5
Alamat: 006C0DE8, Data: 6
Alamat: 006C0E00, Data: 8
Ln 58, Col 28 Spaces: 4 UTF-8 CRLF C
Type here to search
32°C Cerah
11:20
21/05/2024