**LAPORAN PRAKTIKUM**

**SESI IV**

**PRAKTIKUM COMP6362 – DATA STRUCTURES**

**KELAS BC20**



Oleh :

2440059495 – Bismo Agung Tri Achmad Bramantyo

**SEMESTER GENAP 2020/2021  
BINA NUSANTARA UNIVERSITY**

**MALANG**

## **Kode Program**

|  |
| --- |
| Source Code |

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

struct admin

{

    int no;

    char name[20];

    char type[20];

    char address[30];

    char phone[12];

    struct admin \*next;

}\*current, \*newnode;

struct admin \*head = NULL;

struct admin \*top = NULL;

int number = 0;

char message\_non\_available[]=("\nThere are no Information Available\n");

void insert(){

    char name[20], address[30], phone[12], type[20];

     newnode = (struct admin\*)malloc(sizeof(struct admin));

        do{

        printf("\nInput name of person/company/society[3...20]: ");

        scanf("%[^\n]", &\*name); getchar();

        }while(strlen(name) < 3 || strlen(name) > 20);

        do{

        printf("\nInput the Type of New Entry Name [person/company/society]: ");

        scanf("%[^\n]", &\*type);  getchar();

        }while(strcmp(type,"person") != 0 && strcmp(type,"company")!=0 && strcmp(type,"society")!=0);

        do

        {

        printf("\nInput Address of New Entry[6...20]: ");

        scanf("%[^\n]", &\*address); getchar();

        } while (strlen(address)<6 || strlen(address)>20);

        do

        {

        printf("\nInput Phone of New Entry Name [9...12]: ");

        scanf("%[^\n]", &\*phone); getchar();

        } while (strlen(phone)<9 || strlen(phone)> 12);

     strcpy(newnode->name, name);

     strcpy(newnode->type,type);

     strcpy(newnode->address, address);

     strcpy(newnode->phone, phone);

     newnode->no = number + 1;

     if (head == NULL){

         head = top = newnode;

     }

     else{

         top->next = newnode;

         top = newnode;

     }

     top->next = NULL;

     number = number +1;

     getchar();

     printf("\nAdd New Information Success\n");

     getchar();

     system("cls");

}

void delete\_node(){

    current = head;

    if (head == NULL){

        printf("%s", message\_non\_available);

        getchar();

        system("cls");

    }

    else if (head == top){

        head = top = NULL;

        printf("The information about %s(%s) is being delete\n", current->name, current->type);

        getchar();

        system("cls");

        free(current);

    }

    else{

        struct admin \*bantu = NULL;

        bantu = head;

        while(bantu->next != top){

            bantu = bantu->next;

        }

        current = top;

        top = bantu;

        printf("The information about %s(%s) is being delete\n", current->name, current->type);

        getchar();

        system("cls");

        top->next = NULL;

        free(current);

      }

}

void pop(){

    current=top;

    if(top== NULL){

        printf("\n");

    }

    else if(top==head){

        head = top = NULL;

        free(current);

    }

    else{

    struct admin \*bantu = NULL;

    bantu = head;

    while(bantu->next != top){

        bantu = bantu->next;

    }

    top = bantu;

    free(current);

    top->next = NULL;

    }

}

void delete\_all(){

    current = head;

    if (head == NULL){

        printf("%s", message\_non\_available);

        getchar();

        system("cls");

    }

    else{

    while (head != NULL){

        pop();

    }

    printf("\nAll information has been deleted\n");

    getchar();

    system("cls");

    }

    }

void display(){

    current = head;

    printf("\n\t-----INFORMATION LIST-----\n\n\n");

    printf("------+----------------------+----------+----------------------+----------------\n");

    printf("| No. | Name                 | Type     |   Address            |   Phone      |\n");

    printf("------+----------------------+----------+----------------------+----------------\n");

    if(head==NULL){

    printf("------+----------------------+----------+----------------------+----------------\n\n");

    getchar();

    system("cls");

    }

    else{

   do{

    printf("| %2d. | %-20s | %-8s | %-20s | %-12s |\n", current->no, current->name, current->type, current->address, current->phone);

    current = current->next;

    } while(current != NULL);

    printf("------+-----------------------+----------+---------------------+---------------\n\n");

    getchar();

    system("cls");

 }

}

void main(){

    int menu;

    while(menu != 5){

    printf("INFROMATION ADMISNISTRATOR\n");

    printf("===========================\n\n");

    printf("1. Information List\n");

    printf("2. Add (PUSH) New Information\n");

    printf("3. Delete (POP) Lastest Information\n");

    printf("4. Delete All (POP ALL) Information\n");

    printf("5. Exit\n");

    printf("\n\nInput Menu[1/2/3/4/5]: ");

    scanf("%d", &menu); fflush(stdin);

    switch (menu)

    {

    case 1:

        display();

        break;

    case 2:

        insert();

        break;

    case 3:

       delete\_node();

       number = number - 1;

       break;

    case 4:

       delete\_all();

       number = 0;

       break;

    case 5:

        printf("\n--------Thank you, Program Closed-----------\n");

        break;

    }

getchar();

system("cls");

 }

}

## **Deskripsi Program**

* Libray yang dipakai

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

* Struct

struct admin

{

    int no;

    char name[20];

    char type[20];

    char address[30];

    char phone[12];

    struct admin \*next;

}\*current, \*newnode;

struct admin \*head = NULL;

struct admin \*top = NULL;

Disini saya menggunakan struct bernama admin yang memiliki beberapa variabel diantara nya int no untuk menyimpan nomor dari list data, kemudian char name untuk menyimpan nama di setiap node, type untuk menyimpan tipe, address untuk menyimpan alamat, phone untuk menyimpan nomor telepon, kemudian ada pointer next untuk mengarahkan ke node selanjutnya, lalu ada pointer current yang membantu pada proses deleting dan newnode membantu saat proses inserting, kemudian ada pointer head dan top(tail) untuk display dan deleting, kedua pointer ini saya NULL kan agar tidak menunjuk ke memori dimanapun sebelum digunakan.

* Variabel Universal

int number = 0;

char message\_non\_available[]=("\nThere are no Information Available\n");

int number berfungsi untuk mengatur number list / no , char message\_non\_available berfungsi untuk menyimpan string *error message*  yang akan digunakan dalam fungsi-fungsi nantinya.

* Fungsi

Fungsi insert :

void insert(){

    char name[20], address[30], phone[12], type[20];

     newnode = (struct admin\*)malloc(sizeof(struct admin));

        do{

        printf("\nInput name of person/company/society[3...20]: ");

        scanf("%[^\n]", &\*name); getchar();

        }while(strlen(name) < 3 || strlen(name) > 20);

        do{

        printf("\nInput the Type of New Entry Name [person/company/society]: ");

        scanf("%[^\n]", &\*type);  getchar();

        }while(strcmp(type,"person") != 0 && strcmp(type,"company")!=0 && strcmp(type,"society")!=0);

        do

        {

        printf("\nInput Address of New Entry[6...20]: ");

        scanf("%[^\n]", &\*address); getchar();

        } while (strlen(address)<6 || strlen(address)>20);

        do

        {

        printf("\nInput Phone of New Entry Name [9...12]: ");

        scanf("%[^\n]", &\*phone); getchar();

        } while (strlen(phone)<9 || strlen(phone)> 12);

     strcpy(newnode->name, name);

     strcpy(newnode->type,type);

     strcpy(newnode->address, address);

     strcpy(newnode->phone, phone);

     newnode->no = number + 1;

     if (head == NULL){

         head = top = newnode;

     }

     else{

         top->next = newnode;

         top = newnode;

     }

     top->next = NULL;

     number = number +1;

     getchar();

     printf("\nAdd New Information Success\n");

     getchar();

     system("cls");

}

Fungsi void insert() secara garis besar berfungsi untuk membuat node baru.

char name[20], address[30], phone[12], type[20];

beberapa variabel integer digunakan untuk menampung data sementara kemudian data tersebut akan dimasukan kedalam node baru.

     newnode = (struct admin\*)malloc(sizeof(struct admin));

alokasi memori sebelum membuat node baru.

        do{

        printf("\nInput name of person/company/society[3...20]: ");

        scanf("%[^\n]", &\*name); getchar();

        }while(strlen(name) < 3 || strlen(name) > 20);

        do{

        printf("\nInput the Type of New Entry Name [person/company/society]: ");

        scanf("%[^\n]", &\*type);  getchar();

        }while(strcmp(type,"person") != 0 && strcmp(type,"company")!=0 && strcmp(type,"society")!=0);

        do

        {

        printf("\nInput Address of New Entry[6...20]: ");

        scanf("%[^\n]", &\*address); getchar();

        } while (strlen(address)<6 || strlen(address)>20);

        do

        {

        printf("\nInput Phone of New Entry Name [9...12]: ");

        scanf("%[^\n]", &\*phone); getchar();

        } while (strlen(phone)<9 || strlen(phone)> 12);

Disini user diminta untuk menginput data data yamng diminta seperti nama, tipe, alamat dan nomor telepon. User akan terus diminta menginput selama data yang dimasukan tidak memenuhi standard, dengan cara data akan dibaca dan disimpan di variabel sementara lalu dilakuakan pemeriksaan.

* Standard nama : panjang kalimat minimal 3 dan tidak lebih dari 20 (spasi dibaca)
* Standard tipe : person , company , society -> huruf kecil semua tidak berspasi
* Standard alamat: panjang kalimat minimal 6 dan tidak lebih dari 20(spasi dibaca)
* Standard nomor telephone : minimal 9 dan tidak lebih dari 12 (spasi dibaca)
* strcpy(newnode->name, name);
* strcpy(newnode->type,type);
* strcpy(newnode->address, address);
* strcpy(newnode->phone, phone);
* newnode->no = number + 1;

disini semua data yang memenuhi standard yang disimpan di variabel sementara akan dimasukan ke dalam data di node baru, khusus untuk data no pada node baru akan di assign menggunakan int number + 1 yang nantinya akan menjadi nomor urut.

     if (head == NULL){

         head = top = newnode;

     }

     else{

         top->next = newnode;

         top = newnode;

     }

     top->next = NULL;

     number = number +1;

     getchar();

     printf("\nAdd New Information Success\n");

     getchar();

     system("cls");

}

Disini terjadi percabangan kondisi, ketika data kosong maka pointer head dan top (tail) akan menunjuk pada node baru. Jika ada data maka pointer next dari node top akan menunjuk ke newnode, lalu pointer top akan berpindah ke node baru. Lalu pointer next dari top akan di NULL kan, number + 1 suapaya nomor urut berikutnya bisa lebih besar.

Fungsi delete:

void delete\_node(){

    current = head;

    if (head == NULL){

        printf("%s", message\_non\_available);

        getchar();

        system("cls");

    }

    else if (head == top){

        head = top = NULL;

        printf("The information about %s(%s) is being delete\n", current->name, current->type);

        getchar();

        system("cls");

        free(current);

    }

    else{

        struct admin \*bantu = NULL;

        bantu = head;

        while(bantu->next != top){

            bantu = bantu->next;

        }

        current = top;

        top = bantu;

        printf("The information about %s(%s) is being delete\n", current->name, current->type);

        getchar();

        system("cls");

        top->next = NULL;

        free(current);

      }

}

Fungsi delete\_node() secara garis besar mempunyai fungsi yang sama dengan pop\_tail yaitu menghapus data paling akhir yang ditambahkan.

Pada fungsi ini pointer current akan menunjuk pada node yang dtunjuk oleh pointer head, ketika head kosong (tidk ada data) maka program akan menampilakn error message.

Selanjutnya program akan mengecek apakah data hanya ada satu, apabila iya maka pointer head dan top akan di NULL kan, lalu node dihapus.

Apabila tidak maka pointer bantu akan didefinisikan, kemudian pointer bantu akan menunjuk node head, selama pointer next dari node yang ditunjuk oleh pointer bantu tidak menunjuk ke top, maka pointer bantu akan bergeser terus ke 1 data selanjutnya. Ketika sudah mencapai node tujuan maka pointer current akan menujuk node top, lalu top berpindah menujuk ke bantu, dan next dari top di NULL kan, lalu current di hapus.

Void pop():

Fungsi ini memiliki fungsi yang sama dengan funsgi delete\_node tapi fungsi ini hanya digunakan saat rekursi di void delete\_all(), dan current nya langsung menunjuk ke top

void pop(){

    current=top;

    if(top== NULL){

        printf("\n");

    }

    else if(top==head){

        head = top = NULL;

        free(current);

    }

    else{

    struct admin \*bantu = NULL;

    bantu = head;

    while(bantu->next != top){

        bantu = bantu->next;

    }

    top = bantu;

    free(current);

    top->next = NULL;

    }

}

Void delete\_all():

void delete\_all(){

    current = head;

    if (head == NULL){

        printf("%s", message\_non\_available);

        getchar();

        system("cls");

    }

    else{

    while (head != NULL){

        pop();

    }

    printf("\nAll information has been deleted\n");

    getchar();

    system("cls");

    }

    }

pada fungsi ini semua data yang ada akan di delete / hapus.

Pointer current akan menunjuk ke node head, kemudian dicek, apabila data kosong maka program akan menampilkan error message, selain itu program akan menghapus semua data dengan cara memanggil fungsi pop, selama data di node head tidak kosong.

Void display():

void display(){

    current = head;

    printf("\n\t-----INFORMATION LIST-----\n\n\n");

    printf("------+----------------------+----------+----------------------+----------------\n");

    printf("| No. | Name                 | Type     |   Address            |   Phone      |\n");

    printf("------+----------------------+----------+----------------------+----------------\n");

    if(head==NULL){

    printf("------+----------------------+----------+----------------------+----------------\n\n");

    getchar();

    system("cls");

    }

    else{

   do{

    printf("| %2d. | %-20s | %-8s | %-20s | %-12s |\n", current->no, current->name, current->type, current->address, current->phone);

    current = current->next;

    } while(current != NULL);

    printf("------+-----------------------+----------+---------------------+---------------\n\n");

    getchar();

    system("cls");

 }

}

Fungsi ini untuk menampilkan list data yang telah disimpan.

Pointer current akan menujuk head, kemudian prgram akan mencetak header list, setelah itu program akan mengecek apakah data kosong, apabila iya program akan mencetak footer, sehingga tidak ada data yang di print, kemudian apabilaada data program akan mencetak list data sesuai field dan nomor yang terurut, data yang diprint akan diberi batas spasi maksimum sehingga tabel tidak rusak.

Fungsi main():

void main(){

    int menu;

    while(menu != 5){

    printf("INFROMATION ADMISNISTRATOR\n");

    printf("===========================\n\n");

    printf("1. Information List\n");

    printf("2. Add (PUSH) New Information\n");

    printf("3. Delete (POP) Lastest Information\n");

    printf("4. Delete All (POP ALL) Information\n");

    printf("5. Exit\n");

    printf("\n\nInput Menu[1/2/3/4/5]: ");

    scanf("%d", &menu); fflush(stdin);

    switch (menu)

    {

    case 1:

        display();

        break;

    case 2:

        insert();

        break;

    case 3:

       delete\_node();

       number = number - 1;

       break;

    case 4:

       delete\_all();

       number = 0;

       break;

    case 5:

        printf("\n--------Thank you, Program Closed-----------\n");

        break;

    }

getchar();

system("cls");

 }

}

Pada fungsi main ini hanya ada 1 variabel yaitu menu bertipe int, berguna saat switch case.

User akan diminta menginput menu, selain angka 5, program tidak akan berhenti, selain angka 1 2 3 4, program akan kembali ke menu awal.

Untuk case 1 program akan menjalankan fungsi display.

Untuk case 2 program akan menjalankan fungsi insert.

Untuk case 3 program akan menjalankan fungsi delete\_node

Untuk case 4 program akan menjalankan fungsi delete all.

Case 5 untuk menghentikan eksekusi program dan menutup program.

## **Bukti Presentasi**

