**STRUKTUR DATA**

(UAS)

****

Nama : Prames Ray Lapian

NPM : 140810210059

Dikumpulkan tanggal :

6 Juni 2022

UNIVERSITAS PADJADJARAN

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

Program Studi INFORMATIKA

2022

1. Soal 2

#include <iostream>

using namespace std;

const int maxElemen = 255;

struct Stack

{

    int isi[maxElemen];

    int TOP;

};

Stack S;

void createStack (Stack& S)

{

    S.TOP = -1;

}

void push (Stack& S, int elemenBaru)

{

    if (S.TOP == maxElemen-1)

    {

        cout <<"Stack Overflow" << endl;

    }

    else

    {

        S.TOP += 1;

        S.isi[S.TOP] = elemenBaru;

    }

}

void pop(Stack& S, int& elemenHsl)

{

    if (S.TOP < 0)

    {

        cout << "Stack Underflow " << endl;

    }

    else

    {

        elemenHsl= S.isi[S.TOP];

        S.TOP -= 1;

    }

}

void traversal(Stack& S)

{

    int idxBantu = S.TOP;

    while (idxBantu >= 0)

    {

        cout << "\_\_" << endl

             << S.isi[idxBantu] << endl;

        idxBantu -= 1;

    }

}

void swap(int& a, int& b){

    int temp = a;

    a = b;

    b = temp;

}

void BubbleSort(Stack& S)

{

    for(int i=0; i<S.TOP-1; i++)

    {

        for(int j=0; j<S.TOP-i-1; j++)

        {

            if(S.isi[j]<S.isi[j+1])

            {

                swap(S.isi[j], S.isi[j+1]);

            }

        }

    }

}

main()

{

    Stack myTumpukan;

    int temp;

    cout << "PROGRAM STACK ARRAY" << endl;

    createStack(myTumpukan);

    push(myTumpukan, 6);

    push(myTumpukan, 9);

    push(myTumpukan, 8);

    push(myTumpukan, 10);

    push(myTumpukan, 1);

    cout << "\nSebelum Sort:" << endl;

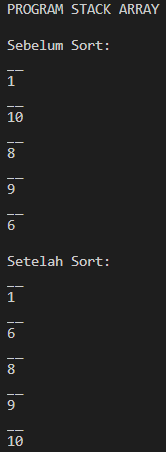
    traversal(myTumpukan);

    cout << "\nSetelah Sort:" << endl;

    BubbleSort(myTumpukan);

    traversal(myTumpukan);

}



1. Soal 3

#include <iostream>

#include <string.h>

#include <iomanip>

using namespace std;

struct Pegawai

{

    char Nama[30];

};

struct Node

{

    Pegawai info;

    Node\* next;

    Node\* prev;

};

typedef Node \*pointer;

typedef pointer List;

void createElement(pointer& pBaru)

{

    pBaru = new Node;

    cout << "\nData pegawai yang ingin ditambahkan:" << endl;

    cout << "Nama   : "; cin.get(pBaru->info.Nama,30); cin.ignore();

    pBaru->next = NULL;

}

void insertFirst(List& first, pointer pBaru)

{

    if (first == NULL)

    {

        first = pBaru;

    }

    else

    {

        pointer pBantu = first;

        while (pBantu->next != NULL)

        {

            pBantu = pBantu->next;

        }

        pBantu->next = pBaru;

        pBaru->prev = pBantu;

    }

}

void traversal(List first)

{

    if (first == NULL)

    {

        cout << "\nList kosong!" << endl;

    }

    else

    {

        pointer pBantu = first;

        cout << endl;

        cout << setw(30) << "NAMA" << endl;

        do

        {

            cout << setw(30) << pBantu->info.Nama << endl;

            pBantu = pBantu->next;

        }

        while (pBantu != NULL);

    }

}

void sambungList(List& first1, List& first2)

{

    if (first1 == NULL)

    {

        first1 = first2;

    }

    else if (first2 == NULL)

    {

        first2 = first1;

    }

    else

    {

        pointer pBantu1 = first1;

        pointer pBantu2 = first2;

        while (pBantu1->next != NULL)

        {

            pBantu1 = pBantu1->next;

        }

        pBantu1->next = first2;

        pBantu2->prev = pBantu1;

    }

}

int main()

{

    List first1 = NULL, first2 = NULL;

    pointer pBaru;

    createElement(pBaru);

        insertFirst(first1, pBaru);

    createElement(pBaru);

        insertFirst(first1, pBaru);

    createElement(pBaru);

        insertFirst(first1, pBaru);

    createElement(pBaru);

        insertFirst(first2, pBaru);

    createElement(pBaru);

        insertFirst(first2, pBaru);

    createElement(pBaru);

        insertFirst(first2, pBaru);

    sambungList(first1, first2);

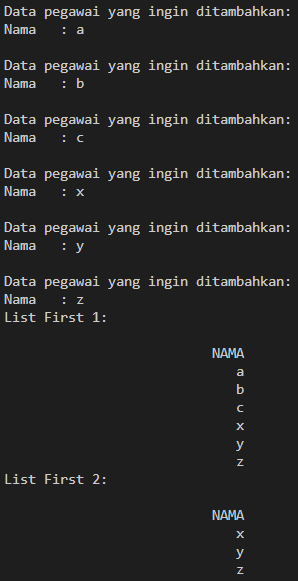
    cout << "List First 1:" << endl;

    traversal(first1);

    cout << "List First 2:" << endl;

    traversal(first2);

}



1. Soal 4

#include <iostream>

#include <iomanip>

#include <string.h>

using namespace std;

struct Karyawan

{

    char namaKar[50];

    Karyawan \*nextKar;

};

struct NIP

{

    char NIPKar[20];

    Karyawan \*firstNIP;

    NIP \*nextNIP;

};

struct Divisi

{

    char divPeg[50];

    NIP \*firstNIP;

    Divisi \*nextDiv;

};

typedef Karyawan \*pKaryawan;

typedef NIP \*pNIP;

typedef Divisi \*pDiv;

typedef pDiv List;

char menu()

{

    char opsi;

    cout << "======================================" << endl

         << "          MENU PROGRAM Divisi        " << endl

         << "======================================" << endl

         << "1. Input Set Data Divisi             " << endl

         << "2. Input Data Divisi                 " << endl

         << "3. Input Data NIP                     " << endl

         << "4. Input Data Karyawan                  " << endl

         << "5. Hapus Data Divisi                 " << endl

         << "6. Hapus Data NIP                     " << endl

         << "7. Hapus Data Karyawan                  " << endl

         << "8. Tampilkan Seluruh Data             " << endl

         << "0. Keluar Program                     " << endl

         << "======================================" << endl << endl

         << "opsi\t: "; cin >> opsi;

    return opsi;

}

void createList(List &first)

{

    first = NULL;

}

void createDivisi(pDiv &newDiv)

{

    newDiv = new Divisi;

    cout << "Nama Divisi : "; cin.getline(newDiv->divPeg, 50);

    newDiv->firstNIP = NULL;

    newDiv->nextDiv = NULL;

}

void createNIP(pNIP &newNIP)

{

    newNIP = new NIP;

    cout << "NIP : "; cin.getline(newNIP->NIPKar, 20);

    newNIP->firstNIP = NULL;

    newNIP->nextNIP = NULL;

}

void createKaryawan(pKaryawan &newKaryawan)

{

    newKaryawan = new Karyawan;

    cout << "Karyawan : "; cin.getline(newKaryawan->namaKar, 50);

    newKaryawan->nextKar = NULL;

}

void insertLastDivisi(List &first, pDiv newDiv)

{

    if (first == NULL)

    {

        first = newDiv;

    }

    else

    {

        pDiv last = first;

        while (last -> nextDiv != NULL)

        {

            last = last -> nextDiv;

        }

        last -> nextDiv = newDiv;

    }

}

void deleteFirstDivisi(List &first, pDiv &pDelete)

{

    if (first == NULL)

    {

        pDelete = NULL;

        cout << "List Divisi Kosong\n";

    }

    else if (first->nextDiv == NULL)

    {

        pDelete = first;

        first = NULL;

        cout << "Divisi " << pDelete->divPeg << " Berhasil Dihapus\n";

    }

    else

    {

        pDelete = first;

        first = first -> nextDiv;

        pDelete -> nextDiv = NULL;

        cout << "Divisi " << pDelete->divPeg << " Berhasil Dihapus\n";

    }

}

void linearSearchDivisi(List first, char key[], int &status, pDiv &pDivisi)

{

    status = 0;

    pDivisi = first;

    while (pDivisi != NULL && status == 0)

    {

        if (strcmp(pDivisi->divPeg, key) == 0)

        {

            status = 1;

        }

        else

        {

            pDivisi = pDivisi->nextDiv;

        }

    }

}

void linearSearchNIP(pDiv pDivisi, char key[], int &status, pNIP &pNIP)

{

    status = 0;

    pNIP = pDivisi->firstNIP;

    while (pNIP != NULL && status == 0)

    {

        if (strcmp(pNIP->NIPKar, key) == 0)

        {

            status = 1;

        }

        else

        {

            pNIP = pNIP->nextNIP;

        }

    }

}

void insertFirstNIP(pDiv pDivisi, pNIP newNIP)

{

    if (pDivisi->firstNIP == NULL)

    {

        pDivisi->firstNIP = newNIP;

    }

    else

    {

        newNIP->nextNIP = pDivisi->firstNIP;

        pDivisi->firstNIP = newNIP;

    }

}

void deleteFirstNIP(pDiv pDivisi, pNIP &pDelete)

{

    if (pDivisi->firstNIP == NULL)

    {

        pDelete = NULL;

        cout << "List NIP Kosong\n";

    }

    else if (pDivisi->firstNIP->nextNIP == NULL)

    {

        pDelete = pDivisi->firstNIP;

        pDivisi->firstNIP = NULL;

        cout << "NIP " << pDelete->NIPKar << " Berhasil Dihapus\n";

    }

    else

    {

        pDelete = pDivisi->firstNIP;

        pDivisi->firstNIP = pDivisi->firstNIP->nextNIP;

        pDelete->nextNIP = NULL;

        cout << "NIP " << pDelete->NIPKar << " Berhasil Dihapus\n";

    }

}

void insertFirstKar(pNIP pNIP, pKaryawan newKaryawan)

{

    if (pNIP->firstNIP == NULL)

    {

        pNIP->firstNIP = newKaryawan;

    }

    else

    {

        newKaryawan->nextKar = pNIP->firstNIP;

        pNIP->firstNIP = newKaryawan;

    }

}

void deleteFirstKar(pNIP pNIP, pKaryawan &pDelete)

{

    if (pNIP->firstNIP == NULL)

    {

        pDelete = NULL;

        cout << "List Karyawan Kosong\n";

    }

    else if (pNIP->firstNIP->nextKar == NULL)

    {

        pDelete = pNIP->firstNIP;

        pNIP->firstNIP = NULL;

        cout << "Karyawan " << pDelete->namaKar << " Berhasil Dihapus\n";

    }

    else

    {

        pDelete = pNIP->firstNIP;

        pNIP->firstNIP = pNIP->firstNIP->nextKar;

        pDelete->nextKar = NULL;

        cout << "Karyawan " << pDelete->namaKar << " Berhasil Dihapus\n";

    }

}

void traversalDivisi(List first)

{

    pDiv pDivisi = first;

    int no = 1;

    if (first == NULL)

    {

        cout << "List Divisi kosong" << endl;

    }

    else

    {

        cout << "===============================================================================================\n";

        cout << setw(5) << "No" << setw(30) << "Nama Divisi" << setw(30) << "NIP" << setw(30) << "Karyawan" << endl;

        do

        {

            pNIP pNIP = pDivisi->firstNIP;

            cout << setw(5) << no;

            cout << setw(30) << pDivisi->divPeg;

            if (pNIP == NULL)

            {

                cout << setw(30) << "NIP kosong";

            }

            else

            {

                cout << setw(30) << pNIP->NIPKar;

                pKaryawan pKaryawan = pNIP->firstNIP;

                if (pKaryawan != NULL)

                {

                    cout << setw(30) << pKaryawan->namaKar ;

                    pKaryawan = pKaryawan->nextKar;

                    while (pKaryawan != NULL)

                    {

                        cout << endl;

                        cout << setw(95) << pKaryawan->namaKar;

                        pKaryawan = pKaryawan->nextKar;

                    }

                }

                else

                {

                    cout << setw(30) << "Karyawan kosong";

                }

                cout << endl;

                pNIP = pNIP->nextNIP;

                while (pNIP != NULL)

                {

                    pKaryawan = pNIP->firstNIP;

                    cout << setw(65) << pNIP->NIPKar;

                    if (pKaryawan == NULL)

                    {

                        cout << setw(30) << "Karyawan kosong" << endl;

                    }

                    else

                    {

                        cout << setw(30) << pKaryawan->namaKar;

                        pKaryawan = pKaryawan->nextKar;

                        while (pKaryawan != NULL)

                        {

                            cout << endl;

                            cout << setw(95) << pKaryawan->namaKar;

                            pKaryawan = pKaryawan->nextKar;

                        }

                    }

                    pNIP = pNIP->nextNIP;

                }

            }

            cout << endl;

            no++;

            pDivisi = pDivisi->nextDiv;

        }while (pDivisi != NULL);

        cout << "===============================================================================================\n";

    }

}

int main()

{

    List listDivisi;

    pDiv pNewDivisi, pDeleteDivisi, pDivisi;

    pNIP pNewNIP, pDeleteNIP, pNIP;

    pKaryawan pNewKaryawan, pDeleteKaryawan;

    bool program = true;

    char opsi, key[50];

    int status;

    createList(listDivisi);

    while (program)

    {

        system("cls");

        opsi = menu();

        cin.ignore();

        switch (opsi)

        {

            case '1' :

                createDivisi(pNewDivisi);

                insertLastDivisi(listDivisi, pNewDivisi);

                    createNIP(pNewNIP);

                    insertFirstNIP(pNewDivisi, pNewNIP);

                        createKaryawan(pNewKaryawan);

                        insertFirstKar(pNewNIP, pNewKaryawan);

                cout << "Data berhasil ditambahkan" << endl;

                system("pause");

                break;

            case '2':

                createDivisi(pNewDivisi);

                insertLastDivisi(listDivisi, pNewDivisi);

                cout << "Divisi berhasil ditambahkan" << endl;

                system("pause");

                break;

            case '3':

                //cari Divisi

                cout << "Masukkan nama Divisi : "; cin.getline(key, 50);

                linearSearchDivisi(listDivisi, key, status, pDivisi);

                if (status == 1)

                {

                    //tambahkan NIP

                    createNIP(pNewNIP);

                    insertFirstNIP(pDivisi, pNewNIP);

                    cout << "NIP berhasil ditambahkan" << endl;

                }

                else

                {

                    cout << "Divisi tidak ditemukan" << endl;

                }

                system("pause");

                break;

            case '4':

                //cari Divisi

                cout << "Masukkan nama Divisi : "; cin.getline(key, 50);

                linearSearchDivisi(listDivisi, key, status, pDivisi);

                if (status == 1)

                {

                    //cari NIP

                    cout << "Masukkan NIP : "; cin.getline(key, 20);

                    linearSearchNIP(pDivisi, key, status, pNIP);

                    if (status == 1)

                    {

                        //tambah Karyawan

                        createKaryawan(pNewKaryawan);

                        insertFirstKar(pNIP, pNewKaryawan);

                        cout << "Karyawan berhasil ditambahkan" << endl;

                    }

                    else

                    {

                        cout << "NIP tidak ditemukan" << endl;

                    }

                }

                else

                {

                    cout << "Divisi tidak ditemukan" << endl;

                }

                system("pause");

                break;

            case '5':

                //hapus Divisi

                deleteFirstDivisi(listDivisi, pDeleteDivisi);

                system("pause");

                break;

            case '6':

                //hapus NIP

                cout << "Masukkan nama Divisi : "; cin.getline(key, 50);

                linearSearchDivisi(listDivisi, key, status, pDivisi);

                if (status == 1)

                {

                    deleteFirstNIP(pDivisi, pDeleteNIP);

                }

                else

                {

                    cout << "Divisi tidak ditemukan" << endl;

                }

                system("pause");

                break;

            case '7':

                //hapus Karyawan

                cout << "Masukkan nama Divisi : "; cin.getline(key, 50);

                linearSearchDivisi(listDivisi, key, status, pDivisi);

                if (status == 1)

                {

                    cout << "Masukkan NIP : "; cin.getline(key, 20);

                    linearSearchNIP(pDivisi, key, status, pDeleteNIP);

                    if (status == 1)

                    {

                        deleteFirstKar(pDeleteNIP, pDeleteKaryawan);

                    }

                    else

                    {

                        cout << "NIP tidak ditemukan" << endl;

                    }

                }

                else

                {

                    cout << "Divisi tidak ditemukan" << endl;

                }

                system("pause");

                break;

            case '8':

                //traversal

                system("cls");

                traversalDivisi(listDivisi);

                system("pause");

                break;

            case '0':

                program = false;

                break;

            default :

                cout << "opsi tidak ada" << endl;

                system("pause");

                break;

        }

        cout << "Program Selesai Terima Kasih!\n";

    }

}

