**ALGORITMA DAN STRUKTUR DATA**

Laporan Praktikum Linked List



**Oleh :**

Fahrial Ananta (5223600030)

**PROGRAM STUDI D4 TEKNOLOGI GAME**

**DEPARTEMEN TEKNOLOGI MULTIMEDIA KREATIF**

**POLITEKNIK ELEKTRONIKA NEGERI SURABAYA**

#include <iostream>

using namespace std;

struct Node {

int \_Data;

Node\* \_Next;

};

class SingleLinkedList

{

Node\* \_Tail = nullptr;

Node\* \_Head = nullptr;

public:

Node\* CreateNode(int data) {

Node\* newNode = new Node;

newNode->\_Data = data;

newNode->\_Next = nullptr;

return newNode;

}

void InsertNode(int data) {

Node\* NewNode = CreateNode(data);

if (!\_Head && !\_Tail)

{

\_Tail = NewNode;

\_Head = NewNode;

}

else

{

NewNode->\_Next = \_Head;

\_Head = NewNode;

}

}

void DeleteNode(int data)

{

Node\* CurrectNode = \_Head;

bool IsNotFound = true;

while (IsNotFound)

{

if (CurrectNode->\_Data == data)

{

delete CurrectNode;

\_Head->\_Next = nullptr;

IsNotFound = false;

return;

}

else if (CurrectNode->\_Next)

CurrectNode = CurrectNode->\_Next;

else

{

IsNotFound = false;

cout << "Data Not Found" << endl;

}

}

}

string SearchNode(int data)

{

Node\* CurrectNode = \_Head;

bool IsNotFound = true;

while (IsNotFound)

{

if (CurrectNode->\_Data == data)

return "Founded";

else if (CurrectNode->\_Next)

CurrectNode = CurrectNode->\_Next;

else

return "NotFound";

}

}

void ShowNode() {

Node\* temp = \_Head;

while (temp) {

cout << temp->\_Data << " ";

temp = temp->\_Next;

}

cout << endl;

}

void ShowTail()

{

cout << \_Tail->\_Data;

}

void ShowHead()

{

cout << \_Head->\_Data;

}

};

bool Repeat()

{

string input;

cout << endl << "want to repeat the program (1 = yes / whatever = no) : ";

cin >> input;

switch (input[0])

{

case '1':

cin.ignore(256, '\n');

return true;

break;

default:

return false;

break;

}

}

int main() {

SingleLinkedList \_SingleLinkedList;

string \_Input;

int \_NodeInput;

\_SingleLinkedList.InsertNode(1);

\_SingleLinkedList.InsertNode(30);

\_SingleLinkedList.InsertNode(53);

\_SingleLinkedList.InsertNode(12);

\_SingleLinkedList.InsertNode(88);

\_SingleLinkedList.InsertNode(56);

\_SingleLinkedList.InsertNode(66);

do

{

cout << endl << "Select Operation" << endl;

cout << "1. Implementasikan operasi dasar Single linked list : Menyisipkan sebagai simpul ujung(awal) dari linked list" << endl;

cout << "2. Implementasikan operasi dasar Single linked list : Membaca atau menampilkan" << endl;

cout << "3. Implementasikan operasi dasar Single linked list : Mencari sebuah simpul tertentu.Tambahkan kondisi jika yang dicari adalah data yang paling depan" << endl;

cout << "4. Implementasikan operasi dasar Single linked list : Menyisipkan sebagai simpul terakhir" << endl;

cout << "Input : ";

cin >> \_Input;

switch (\_Input[0])

{

case '1':

\_SingleLinkedList.ShowTail();

break;

case '2':

\_SingleLinkedList.ShowNode();

break;

case '3':

\_SingleLinkedList.ShowNode();

cout << "Enter the data you want to search for : ";

cin >> \_NodeInput;

cout << \_SingleLinkedList.SearchNode(\_NodeInput);

break;

case '4':

\_SingleLinkedList.ShowHead();

break;

default:

cout << "Not Found";

break;

}

} while (Repeat());

}