**Soal**

1. Buatlah sebuah stack dengan menggunakan double linked list.
2. Buatlah sebuah queue dengan menggunakan single linked list menggunakan penanda pointer head dan tail.
3. Buatlah sebuah deque dengan menggunakan single linked list.
4. Buatlah sebuah deque dengan menggunakan double linked list.

**Jawaban**

* Program: stack dengan menggunakan double linked list.

#include <iostream>

#include <conio.h>

using namespace std;

struct Node

{

int data;

struct Node\* prev;

struct Node\* next;

};

Node\* start = NULL;

Node\* top = NULL;

// Check if stack is empty

bool isEmpty()

{

if (start == NULL)

return true;

return false;

}

// pushes element onto stack

void push(int d)

{

struct Node\* n;

n = new Node();

n->data = d;

if (isEmpty())

{

n->prev = NULL;

n->next = NULL;

// As it is first node if stack

// is empty

start = n;

top = n;

}

else

{

top->next = n;

n->next = NULL;

n->prev = top;

top = n;

}

}

// Pops top element from stack

void pop()

{

struct Node\* n;

n = top;

if (isEmpty())

printf("Stack is empty");

else if (top == start)

{

top = NULL;

start = NULL;

free(n);

}

else

{

top->prev->next = NULL;

top = n->prev;

free(n);

}

}

// Prints top element of the stack

void topelement()

{

if (isEmpty())

printf("Stack is empty");

else

printf(

"The element at top of the stack is : %d \n",

top->data);

}

// Determines the size of the stackvoid stacksize()

{

int c = 0;

if (isEmpty())

printf("Stack is empty");

else

{

struct Node\* ptr = start;

while (ptr != NULL)

{

c++;

ptr = ptr->next;

}

}

printf("Size of the stack is : %d \n ", c);

}

// Determines the size of the stack

void printstack()

{

if (isEmpty())

printf("Stack is empty");

else

{

struct Node\* ptr = start;

printf("ISI STACK : ");

while (ptr != NULL)

{

printf("%d ", ptr->data);

ptr = ptr->next;

}

printf("\n");

}

}

// Driver code

int main()

{

int pilih, data;

do

{

system("cls");

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

cout << "= MENU =" << endl;

cout << "= 1.PUSH = " << endl;

cout << "= 2.POP =" << endl;

cout << "= 3.VIEW =" << endl;

cout << "= 4.EXIT =" << endl;

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

cout << "PILIH : ";

cin >> pilih;

switch (pilih)

{

case 1:

cout << "MASUKKAN DATA : "; cin >> data;

push(data);

cout << "KLIK UNTUK MELANJUTKAN ";

break;

case 2:

pop();

cout << "KLIK UNTUK MELANJUTKAN "; break;

case 3:

printstack();

cout << "KLIK UNTUK MELANJUTKAN ";

break;

default:

cout << "PILIHAN TIDAK ADA " << endl;

cout << "KLIK UNTUK MELANJUTKAN ";

break;

}

getch();

} while (pilih != 4);

return 0;

}

* Program: deque dengan menggunakan single linked list.

#include <iostream>

#include <stdlib.h>

using namespace std;

struct node

{

char data;

struct node\* next;

struct node\* prev;

};

typedef struct node node;

node\* head, \* tail;

int choice;

char item; int count = 0;

int keluar = 0;

void initial()

{

head = tail = NULL;

}

int isEmpty()

{

if (tail == NULL)

return 1;

else

return 0;

}

void enqueue(char item)

{

node\* baru = new node;

baru->data = item;

baru->next = baru;

baru->prev = baru;

if (isEmpty() == 1)

{

head = tail = baru;

head->next = head;

head->prev = head;

tail->next = tail;

tail->prev = tail;

}

else {

baru->next = head;

head->prev = baru;

head = baru;

head->prev = tail;

head->next = head;

}

cout << "\n# Queue : No urut/index : " << count << ", Value :"

<< item;

count++;

}

void dequeue()

{

if (isEmpty() == 0)

{

if (head->next != tail)

{

node\* hapus = tail;

tail = tail->prev;

tail->next = head;

head->prev = tail;

delete hapus;

cout << "\n##Dequeue result:" << item;

cout << "\n##jumlah item dalam queue : " << count;

--count;

}

else

{

head = tail = NULL;

}

}

else

{

cout << "\n## Queue kosong";

}

}

void printAll()

{

cout << "\n## Queue Size : " << count;

node\* temp = head;

int i = 0;

if (isEmpty() == 0)

{

do

{

cout << "\n## No Urut/index : " << i << ", Value :" << temp -

> data;

temp = temp->next;

i++;

} while (temp != head);

}

else

{

cout << "List Kosong.";

}

}

void menu() {

cout << "\nMasukkan operasi yang akan dilakukan (1:enqueue,

2:dequeue, 3 : print) : ";

cin >> choice;

switch (choice)

{

case 1:

{

cout << "\nMasukkan huruf yang akan dimasukkan dalam

queue : ";

cin >> item;

enqueue(item);

break;

}

case 2:

dequeue();

break;

case 3:

printAll();

break;

default:

cout << "\n1:enqueue, 2:dequeue, 3:print\n";

keluar = 1;

break;

}

}

int main()

{

initial(); do

{

menu();

} while (keluar == 0);

}

* Program: deque dengan menggunakan double linked list.

#include <iostream>

#include <windows.h>

using namespace std;

// Queue for Double Linklist

struct dlist

{

dlist\* prev;

int data;

dlist\* next;

};

dlist\* first, \* current, \* previos, \* tamp;

int dlinklist\_counter = 0;

void dlinklist\_insert();

void dlinklist\_call();

void dlinklist\_dequeu();

void dlinklist\_show();

void dlinklist\_front();

int main()

{

system("cls"); dlinklist\_call();

return 0;

}

// function of DOUBLE LINK LIST

void dlinklist\_call()

{

dlinklist\_start:

system("cls");

cout << "\t\t\t\t Welcome in Double linklist Queue";

int dinput;

cout << "\n 1- Enqueu \n 2- Dequeu \n 3- show list \n 4- Front\n 5-

Exit\n";

cin >> dinput;

switch (dinput)

{

case 1:

dlinklist\_insert();

cout << " Number entered \n";

system("pause");

goto dlinklist\_start;

case 2:

dlinklist\_dequeu();

cout << "Number deleted \n ";

system("pause");

goto dlinklist\_start;

case 3:

dlinklist\_show();

goto dlinklist\_start;

case 4:dlinklist\_front();

goto dlinklist\_start;

case 5:

break;

default:

cout << " You enter invalid number ";

system("pause");

goto dlinklist\_start;

}

}

void dlinklist\_insert()

{

current = new dlist;

if (dlinklist\_counter == 0)

{

previos = current;

first = current;

current->prev = NULL;

cout << " Enter Data ";

cin >> current->data;

}

else

{

previos->next = current;

current->prev = previos;

previos = current;

cout << " Enter Data ";

cin >> current->data;

}

current->next = NULL; dlinklist\_counter++;

}

void dlinklist\_dequeu()

{

if (dlinklist\_counter == 0)

{

cout << " Queue is empty";

system("pause");

}

else

{

first = first->next;

dlinklist\_counter--;

}

}

void dlinklist\_show()

{

if (dlinklist\_counter == 0)

{

cout << " Queue is empty";

}

else

{

tamp = first;

while (tamp->next != NULL)

{

cout << " " << tamp->data;

tamp = tamp->next;

}

cout << " " << tamp->data;

}

system("pause");

}

void dlinklist\_front()

{

if (dlinklist\_counter == 0)

{

cout << " Queue is empty";

}

else

{

cout << " " << first->data;

}

system("pause");

}