/\* File : linked.h \*/

/\* Deskripsi : ADT linked list dengan representasi fisik pointer \*/

/\* Representasi address dengan pointer \*/

/\* InfoType adalah integer \*/

/\* Dibuat oleh : Ade Chandra Nugraha \*/

#ifndef linked\_H

#define linked\_H

#include <stdio.h>

#include <malloc.h>

#include "boolean.h"

typedef int infotype;

typedef struct tElmtList \*address;

typedef struct tElmtList {

infotype info;

address next;

} ElmtList;

boolean isEmpty(address p);

// Tujuan : memeriksa apakah linked masih kosong

// Linked list kosong , jika Head == NULL

void Create\_Node (address \*p);

// Tujuan : memesan memory dengan alokasi dinamis

// Jika berhasil mengirimkan address, dan jika gagal mengirimkan NULL

// Parameter : P (parameter Output)

void Isi\_Node (address \*p , infotype nilai);

// Tujuan : Mengisi node yang sudah dipesan dengan Nilai yang dikirimkan

// Periksa keberadaan P, Nilai diisi jika P tidak NULL

// \*p.next diisi NULL

// Parameter : P (parameter Input-Output); Nilai (parameter Input)

void Tampil\_List (address p);

// Tujuan : Menampilkan seluruh isi Linked List

// Parameter : p (parameter input)

void Ins\_Awal (address \*p, address PNew);

// Tujuan : Insert 1 node baru (PNew) di awal Linked List

// Parameter : P (input-output)

// PNew (input)

void Ins\_Akhir (address \*p, address PNew);

/\* Tujuan : menambahkan elemen list (PNew) di akhir LinkedList \*/

/\* IS : p mungkin Kosong \*/

/\* FS : menyambungkan elemen baru (PNew) di akhir Linked List \*/

address Search (address p, infotype nilai);

/\* Mencari apakah ada elemen list dengan Info(P) = nilai \*/

/\* Jika ada, mengirimkan address elemen tsb. \*/

/\* Jika tidak ada, mengirimkan Nil \*/

/\* Menggunakan variabel bertype boolean \*/

void InsertAfter (address \* pBef, address PNew);

/\* Tujuan : Menyambungkan 1 Node baru(PNew) stlah node tertentu(PBef) \*/

/\* IS : pBef sudah dialokasi \*/

/\* FS : Insert PNew sebagai elemen sesudah PBef \*/

/\* Asumsi : pBef adalah posisi insert \*/

void Del\_Awal (address \* p, infotype \* X);

/\* IS : P TIDAK Kosong \*/

/\* FS : Elemen pertama List dihapus, nilai info disimpan ke X \*/

/\* dan alamat elemen pertama di dealokasi \*/

void Del\_Akhir (address \* p, infotype \* X);

/\* IS : P TIDAK Kosong \*/

/\* FS : Elemen terakhir list dihapus : nilai info disimpan pada X \*/

/\* dan alamat elemen terakhir di dealokasi \*/

void Del\_After (address \* pBef, infotype \* X);

/\* IS : pBef TIDAK Kosong (hasil search posisi node sebelum yang didel)\*/

/\* FS : menghapus Node setelah pBef \*/

/\* nilai info node yang dihapus disimpan pada X \*/

/\* dan alamat elemen setelah pBef di dealokasi \*/

void DeAlokasi (address \* p);

/\* IS : P terdefinisi \*/

/\* FS : P dikembalikan ke sistem \*/

/\* Melakukan dealokasi / pengembalian address P ke system \*/

int NbElmt (address p);

/\* Mengirimkan banyaknya elemen list, mengirimkan 0 jika list kosong \*/

// Recursif Mode

infotype Min (address p);

/\* Mengirimkan nilai Info(P) yang minimum \*/

infotype Rerata (address p);

/\* Mengirimkan nilai rata-rata Info(P) \*/

address BalikList (address p);

/\* IS : p sembarang \*/

/\* FS : Elemen List dibalik : elemen terakhir menjadi elemen pertama, dst \*/

#endif