Muhamad Fachri Haikal
1301202398
IF-44-01
AJF MODUL 8
Header:

SLL_Circular.cpp | main.cpp | SLL_Circular.h

```h
#ifndef SLL_CIRCULAR_H_INCLUDED
#define SLL_CIRCULAR_H_INCLUDED

#include <string>
#include <iostream>
using namespace std;

#define info(P) (P)->info
#define next(P) (P)->next
#define first(L) ((L).first)

struct data{
    string nama;
    int prioritas;
    int sisa_durasi;
};

typedef data infotype;
typedef struct elmList *adr;

struct elmList{
    infotype info;
    adr next;
};

struct List{
    adr first;
};

void createList_1301202398(List &L);
adr createElemen_1301202398(infotype dataBaru);
void insertFirst_1301202398(List &L, adr p);
void insertLast_1301202398(List &L, adr p);
void insertAfter_1301202398(List &L, adr p);
void insertDescending_1301202398(List &L, infotype dataBaru);
void deleteFirst_1301202398(List &L);
void deleteLast_1301202398(List &L);
void deleteAfter_1301202398(List &L, adr prec);
void deleteElm_1301202398(List &L, adr p);
void printList_1301202398(List L);
int panjangList_1301202398(List L);
void insertAplikasi_1301202398(List &L, infotype dataBaru);
int eksekusi_1301202398(adr p, int durasi);
void eksekusiMulti_1301202398(List &L, adr &current, int N, int duration)

#endif // SLL_CIRCULAR_H_INCLUDED

```

Muhamad Fachri Haikal
1301202398
IF-44-01
AJF MODUL 8

Body:

*SLL_Circular.cpp | SLL_Circular.h | *main.cpp

```cpp
#include "SLL_Circular.h"

void createList_1301202398(List &L){
    first(L) = NULL;
}

adr createElemen_1301202398(infotype dataBaru){
    adr p = new elmList;

    info(p) = dataBaru;
    next(p) = NULL;
    return p;
}

void insertFirst_1301202398(List &L, adr p){
    adr q;

    if (first(L) == NULL){
        first(L) = P;
        next(p) = first(L);
    }else{
        next(p) = first(L);
        q = first(L);
        while (next(q) != first(L)){
            q = next(q);
        }
        next(q) = p;
        first(L) = p;
    }
}

void insertLast_1301202398(List &L, adr p){
    adr q;
    if (first(L) == NULL){
        first(L) = P;
        next(p) = first(L);
    }else{
        next(p) = first(L);
        q = first(L);
        while (next(q) != First(L)){
            q = next(q);
        }
        next(q) = p;

    }
}
```

*SLL_Circular.cpp | SLL_Circular.h | *main.cpp

```cpp
}
void insertAfter_1301202398(List &L, adr prec, adr p){
    next(p) = next(prec);
    next(prec) = p;
}

void insertDescending_1301202398(List &L, infotype dataBaru){
    adr p, q, prec;

    p = createElemen_1301202398(dataBaru);
    q = first(L);
    if (first(L) == NULL && last(L) == NULL){
        insertFirst_1301202398(L, p);
    }else{
        while (next(q) != first(L) && info(p).prioritas > info(q).prioritas){
            prec = q;
            q = next(q);
        }if (q == first(L)){
            insertFirst_1301202398(L, p);
        }else if (next(q) == first(L)){
            insertLast_1301202398(L, p);
        }else{
            insertAfter_1301202398(L, prec, p);
        }
    }

void deleteFirst_1301202398(List &L){
    adr p, q;

    p = first(L);
    if (next(first(L)) == first(L)){
        first(L) = NULL;
        next(p) = NULL;
    }else{
        first(L) = next(first(L));
        next(p) = NULL;
        q = first(L);
        while (next(q) != p){
            q = next(q);
        }
        next(q) = first(L);
    }
}
```

*SLL_Circular.cpp | SLL_Circular.h | *main.cpp

```cpp
void printList_1301202398(List L){
    adr p;
    int i;

    i = 1
    if (first(L) == NULL){
        cout << "List Kosong!" << endl;
    }else{
        p = first(L);
        while (next(p) != first(L)){
            cout << "[" << i << "] ";
            cout << info(p).nama;
            cout << info(p).prioritas;
            cout << info(p).sisa_durasi;
            cout << endl;
            i++;
            p = next(p);
        }
        cout << "[" << i << "] ";
        cout << info(p).nama;
        cout << info(p).prioritas;
        cout << info(p).sisa_durasi;
        cout << endl;
    }
}

int panjangList_1301202398(List L){
    int jum;
    adr p;

    jum = 0
    p = first(L);
    while (next(p) != first(L)){
        jum++;
        p = next(p);
    }
    jum++
    return jum;
}
```

*SLL_Circular.cpp | SLL_Circular.h | *main.cpp

```cpp
}

void deleteLast_1301202398(List &L){
    adr p, q;
    q = first(L);
    p = first(L);
    while (next(p) != first(L)){
        q = p;
        p = next(p);
    }
    p = next(q);
    next(q) = first(L);
    next(p) = NULL;
}

void deleteAfter_1301202398(List &L){
    adr p;

    p = next(prec);
    next(prec) = next(p);
    next(p) = NULL;
}

void deleteElm_1301202398(List &L, adr p){
    adr q, prec;
    bool ketemu;

    q = first(L);
    ketemu = false;
    while (next(q) != first(L) && ketemu == false){
        if (info(q) == info(p)){
            ketemu = true;
        }else{
            prec = q;
            q = next(q);
        }
    }
    if (ketemu == true){
        if (q == first(L)){
            deleteFirst_1301202398(L);
        }else if (next(q) == first(L)){
            deleteLast_1301202398(L);
        }else{
            deleteAfter_1301202398(L, prec);
        }
    }else{
        cout << "Data tidak ditemukan" << endl;
    }
}
```

Muhamad Fachri Haikal
1301202398
IF-44-01
AJF MODUL 8

SLL_Circular.cpp | main.cpp | SLL_Circular.h

```
}

int panjangList_1301202398(List L){
    int jum;
    adr p;

    jum = 0;
    p = first(L);
    while (next(p) != first(L)){
        jum++;
        p = next(p);
    }
    jum++;
    return jum;
}

void insertAplikasi_1301202398(List &L, infotype dataBaru){
    adr p;
    int panjang;

    panjang = panjangList_1301202398(L);
    if (panjang == 0){
        insertDescending_1301202398(L, dataBaru);
    }else if (panjang <= 5){
        insertDescending_1301202398(L, dataBaru);
    }else{
         p = first(L);
         while (next(p) != first(L)){
            p = next(p);
         }
         if (dataBaru.prioritas > info(p).prioritas){
            deleteElm_1301202398(L, p);
            insertAplikasi_1301202398(L, dataBaru);
        }
    }
}

int eksekusi_1301202398(adr p, int durasi){
    int sisa;

    sisa = info(p).sisa_durasi - durasi;
    return sisa;
}

void eksekusiMulti_1301202398(List &L, adr &current, int N, int duration){
    int i;

    i = 1;
    while (i<=N){
        eksekusi_1301202398(current, duration);
```

```
int eksekusi_1301202398(adr p, int durasi){
    int sisa;

    sisa = info(p).sisa_durasi - durasi;
    return sisa;
}

void eksekusiMulti_1301202398(List &L, adr &current, int N, int duration){
    int i;

    i = 1;
    while (i<=N){
        eksekusi_1301202398(current, duration);
        if (info(current).sisa_durasi <= 0){
            deleteElm_1301202398(L, current);
        }
        current = next(current);
        i++;
    }
}
```

Main:

*SLL_Circular.cpp | main.cpp | SLL_Circular.h

```
#include "SLL_Circular.h"

int main(){
    List L;
    adr p;
    infotype dataBaru;

    const max_application = 5;

    createList_1301202398(L)
    for (int i=1; i<=5; i++){
        cout << "Masukkan data ke-" << i << ":"  << endl;
        cout << "Nama: ";
        cin >> dataBaru.nama;
        cout << "Prioritas: ";
        cin >> dataBaru.prioritas;
        cout << "Sisa Durasi: ";
        cin >> dataBaru.sisa_durasi;
        cout << endl;
        insertAplikasi_1301202398(L, dataBaru);
    }

    printList_1301202398(L);

    p = first(L);
    eksekusiMulti_1301202398(p, 3, 10);

    printList_1301202398(L);

    eksekusiMulti_1301202398(p, 3, 10);

    printList_1301202398(L);
}
```