sll.h

```
#include <iostream>

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

using namespace std;

typedef int infotype;
typedef struct element *adr;

struct element {
	infotype info;
	adr next;
};

struct List {
	adr first;
};

void create_list(List &N);
adr new_element(infotype x);
void insert_first(List &N, adr p);
void insert_after(List &N, adr p, int x);
void insert_last(List &N, adr p);
void show(List N);
adr delete_last(List &N);
```

sll.cpp

```
#include "sll.h"

void create_list(List &N) {
	first(N) = nil;
}

adr new_element(infotype x) {
	adr p = new element;
	info(p) = x;
	next(p) = nil;
	return p;
}

void insert_first(List &N, adr p) {
	if (first(N) == nil) {
	    first(N) = p;
	} else {
	    next(p) = first(N);
```

```
            first(N) = p;
        }
}

void insert_after(List &N, adr p, infotype x) {
        if (first(N) != nil) {
            adr q = first(N);
            while (q != nil && info(q) != x) {
                q = next(q);
            }
            if (q != nil) {
                adr tmp = next(q);
                next(q) = p;
                next(next(q)) = tmp;
            }
        }

}

void insert_last(List &N, adr p) {
        if (first(N) == nil) {
            first(N) = p;
        } else {
            adr q = first(N);
            while (next(q) != nil) {
                q = next(q);
            }
            next(q) = p;
        }
        // cout << "DEBUG: " << info(q) << endl;
}

void show(List N) {
        adr p;
        if (first(N) != nil) {
            p = first(N);
            while (p != nil) {
                cout << info(p) << " ";
                p = next(p);
            }
            cout << endl;
        } else {
            cout << "List Kosong" << endl;
        }
}

adr delete_last(List &N) {
        adr p, q;
        if (first(N) == nil) {
            p = nil;
            cout << "List Kosong" << endl;
        } else if (next(first(N)) == nil) {
            p = first(N);
            first(N) = nil;
```

```
        } else {
            q = first(N);
            p = first(N);
            while (next(p) != nil) {
                q = p;
                p = next(p);
            }
            next(q) = nil;
        }
        return p;
}
```

main.cpp

```
#include <iostream>
#include "sll.h"

int main() {
        List L;

        // cout << "first(L) sebelum createLast : " << first(L) << endl;

        create_list(L);

        // cout << "first(L) setelah createLast : " << first(L) << endl;

        // adr p;
        // p = new_element(100);

        // cout << "Info(p) : " << info(p) << endl;
        // cout << "Next(p) : " << next(p) << endl;

        // cout << "First(L) sebelum insertFirst: " << first(L) << endl;
        // insert_first(L, p);
        // cout << "First(L) setelah insertFirst: " << first(L) << endl;
        // cout << "info First(L): " << info(first(L)) << endl;

        // p = new_element(15);
        // insert_first(L, p);
        // cout << "info First(L): " << info(first(L)) << endl;
        // cout << endl;

        // insert_first(L, new_element(23));
        // insert_first(L, new_element(40));


        // show(L);

        // delete_last(L);
        // show(L);

        // delete_last(L);
```

```
        // show(L);
        // insert_first(L, new_element(14));

        insert_last(L, new_element(1));
        insert_last(L, new_element(2));
        insert_last(L, new_element(3));
        insert_last(L, new_element(4));
        insert_last(L, new_element(5));

        show(L);

        insert_after(L, new_element(8), 3);
        insert_after(L, new_element(7), 3);

        show(L);

        return 0;
}
```