

```

#include <iostream>
using namespace std;

struct p{
    int a;
    p * pNext;
    p * pPrev;
};

int del (int to_delete, p * & pAnker);

int main()
{
    p *pAnchor, *pNew, *pLast;
    int i = 1;
    int tmp;

    cout << "Geben Sie 6 Zahlen ein (getrennt durch Leerzeichen): ";
    pAnchor = new p;
    cin >> tmp;
    pAnchor -> a = tmp;
    pAnchor -> pNext = pAnchor -> pPrev = NULL;
    pLast = pAnchor;
    while (i < 6)
    {
        pNew = new p;
        cin >> tmp;
        pNew -> a = tmp;
        pLast -> pNext = pNew;
        pNew -> pPrev = pLast;
        pLast = pNew;
        pLast -> pNext = NULL;
        i=i+1;
    }

    cout << "Geben Sie die zu loeschende Zahl ein: ";
    cin >> tmp;
    cout << del(tmp,pAnchor) << " Elemente geloescht!" << endl;

    // alle a-Instanzen von hinten beginnend ausgeben
    // ACHTUNG: pLast koennte durch Loeschen des letzten Elementes
    // ungueltig geworden sein, also manuell pLast aktualisieren
    pLast = pAnchor;
    if(pLast != NULL)
        while(pLast->pNext != NULL)
            pLast = pLast->pNext;
    cout << "Ausgabe von hinten nach Loeschvorgang: ";
    p* pHelp = pLast;
    tmp = 0;
    while(pHelp != NULL)
    {
        cout << pHelp->a << " ";
        pHelp = pHelp->pPrev;
        tmp++;
    }
    if(!tmp)
        cout << "Keine Elemente mehr in der Liste!";
    cout << endl;

    return 0;
}

```

```

int del (int to_delete, p * & pAnker)
{
    p *pSearch = pAnker;
    int count_del = 0;

    // Vorkommen von to_delete suchen und loeschen
    while(pSearch != NULL)
    {
        if(pSearch->a == to_delete)
        {
            p *pDelete = pSearch;
            // erstes Element zu loeschen
            if(pDelete->pPrev == NULL){
                pAnker = pDelete->pNext;
                // wenn erstes Element nicht auch letztes
                if(pAnker != NULL)
                    pAnker->pPrev = NULL;
            }
            // letztes Element zu loeschen
            else if(pDelete->pNext == NULL){
                pDelete->pPrev->pNext = NULL;
            }
            // Element in der Mitte zu loeschen
            else{
                pDelete->pPrev->pNext = pDelete->pNext;
                pDelete->pNext->pPrev = pDelete->pPrev;
            }
            pSearch = pSearch->pNext;
            delete pDelete;
            count_del++;
        }
        else
            pSearch = pSearch->pNext;
    }

    return count_del;
}

```