string0_better.cpp - Beispiel für besser entworfene Klasse mit C-string

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

class zk { //Beispiel fuer korrekte Klasse mit dynamischen Zeichenketten
        char *s;
public:
       zk(char *z = 0) : s( z ? strcpy( new char[ strlen(z)+1 ], z ) : 0 ){
          cout<<"Konstruktor zk, s = "<<(this->s ? this->s : "0")<<endl;
          cout<<"        Adresse s = "<<(int *)s<<endl;
       }

// Kopierkonstruktor
     zk(zk &zkd):s(&zkd && zkd.s ? strcpy(new char[strlen(zkd.s)+1],zkd.s):0){
        cout<<"Kopierkonstruktor zk, zkd.s = "<<zkd.s<<endl;
     }

// Zuweisungsoperator
    zk &operator = ( const zk &zkd ){
        if(&zkd != this){
             delete [] s; s=0;
             s=&zkd && zkd.s ? strcpy( new char[ strlen( zkd.s )+1 ], zkd.s ) : 0;
       }
       return *this;
    }

    ~zk(){ cout << "Destruktor zk,  s = " << s << endl; delete [] s; s=0; }
```

string0_better.cpp - Beispiel für besser entworfene Klasse mit C-string

```
    char *get_s(){ return s ? strcpy( new char[strlen(s)+1 ], s ) : 0; }

    void set_s( char *z=0 ){ delete [] s;
                             this->s = z ? strcpy( new char[ strlen(z)+1 ], z ):0; }

    zk &operator+( const zk &zkd ){
        if(s && zkd.s){
            char *t = new char[ strlen(s) + strlen(zkd.s)+1 ];
            strcpy(t,s); strcpy(t+strlen(s), zkd.s);
            delete [] s; s = t;
       }
       if(!s && zkd.s){
            s = strcpy( new char[ strlen(zkd.s)+1 ], zkd.s );
       }
       return *this;
    }

// hier wird this->s durch zkd->s ersetzt, ist ueberfluessig
    void set_zk( zk &zkd ){ delete [] s;
                      s=zkd.s ? strcpy( new char[ strlen(zkd.s)+1 ], zkd.s ) : 0; }
};

void main(){
    char *zt = "HTW Dresden";
    char *z = strcpy (new char[ strlen(zt)+1 ], zt );
    cout<<"        Adresse z = " << (int *)z << endl;
```

string0_better.cpp - Beispiel für besser entworfene Klasse mit C-string

```
zk *s1 = new zk( z);           //Konstruktor
zk *s2 = new zk(*s1);          //Kopierkonstruktor
delete [] z; z=0;

char *s02 = s2->get_s();                    //Rueckgabe Kopie s2->s
cout<<"s2->s = "<<(s02?s02:"0")<<endl;
delete [] s02; s02 = 0;

*s2 = *s1;                                  //Zuweisungsoperator
s02 = s2->get_s();
cout << "s2->s = " << ( s02 ? s02 : "0" ) << endl;
delete [] s02; s02 = 0;

s1->set_s( "TU Dresden" );
char * s01 = s1->get_s();
cout << "s1->s = " << (s01 ? s01 : "0" ) << endl;
delete [] s01; s01 = 0;

s2->set_zk(*s1);
s02 = s2->get_s();
cout << "s2->s = " << ( s02 ? s02 : "0" ) << endl;
delete [] s02; s02 = 0;

*s2+*s1;
s02 = s2->get_s();
```

```
    cout << "s2->s = " << ( s02 ? s02 : "0" ) << endl;
    delete [] s02; s02 = 0;
    delete s1; s1 = 0;
    delete s2; s2 = 0;
    cin.get();
}

/*
        Adresse z = 00322AF0
Konstruktor zk, s = HTW Dresden
        Adresse s = 00322D80
Kopierkonstruktor zk, zkd.s = HTW Dresden
s2->s = HTW Dresden
s2->s = HTW Dresden
s1->s = TU Dresden
s2->s = TU Dresden
Destruktor zk,  s = TU Dresden
Destruktor zk,  s = TU Dresden
*/
```