

```
1: // Klasse Student:
2: // Output-Operator (friend); Mehrere Studien
3: #include "student.h"
4: #include <iostream>
5:
6: using namespace std;
7:
8: int main()
9: {
10:     cout << "Hello world!" << endl;
11:
12:     Student gg("Tyson", "8956256215", 421); // Parmeterkonstruktor
13:
14:     cout << gg.GetName() << endl;
15:     {
16:         const Student pp(gg); // Copy-K
onstruktor
17:         cout << pp.GetName() << endl;
18:
19:         cout << pp._name << endl;
20:     } // impliziter Destruktoraufruf fuer pp
21:
22:
23:
24:     cout << gg.Getmatr_nr();
25:     cout << "\n#####\n";
26:
27:     cout << "test 1: " << gg << endl;
28:
29:     cout << "set new values for name and skz: ";
30:     cin >> gg;
31:     cout << "test 2: " << gg << endl;
32:
33:     gg.Add_SKZ(112);
34:     cout << "test 3: " << gg << endl;
35:     gg.Del_SKZ(666);
36:     gg.Add_SKZ(112); // should not add 112 to the vector again
37:
38:     cout << "test 4: " << gg << endl;
39:
40:     gg.Del_SKZ(421);
41:     gg.Del_SKZ(112);
42:     cout << "test 5: " << gg << endl;
43:
44:
45:     return 0;
46: }
```

```
1: #include "student.h"
2: #include <algorithm>           // copy
3: #include <set>
4: #include <string>
5: #include <vector>
6: using namespace std;
7:
8: Student::Student()
9: : _name(), _matr_nr(), _skz()
10: {
11:     //ctor
12: }
13:
14: Student::Student(const string& name, const string& matrikel, int stud
ium)
15: : _name(name), _matr_nr(matrikel), _skz{studium}
16: {
17:     //ctor
18:
19: }
20:
21: Student::Student(const string & name, const string& matnr, const vect
or<int>& skz)
22: : _name(name), _matr_nr(matnr), _skz()
23: {
24:     //ctor
25:     for (auto v:skz)
26:     {
27:         _skz.insert(v);
28:     }
29: }
30:
31:
32: void Student::Add_SKZ(int skz)
33: {
34:     _skz.insert(skz);
35: }
36:
37: void Student::Del_SKZ(int skz)
38: {
39:     _skz.erase(skz);
40: }
41:
42: const vector<int> Student::Get_SKZ() const
43: {
44:     vector<int> vv(num_Studies()); // allocate memory
45:     copy(_skz.cbegin(), _skz.cend(), vv.begin());
46:     return vv;
47: }
48:
49:
50:
51: ostream& operator<<(ostream& s, const Student& rhs)
52: {
53:     s << rhs._name << endl;
54:     //s << rhs.skz << endl;
55:     for (auto v: rhs._skz)
```

```
56:     {
57:         s << v << " ";
58:     }
59:     s << endl;
60:     return s;
61: }
62:
63: istream& operator>>(istream& s, Student& rhs)
64: {
65:     s >> rhs._name;
66:     int tmp;
67:     s >> tmp;
68:     //    rhs._skz.push_back(tmp);
69:     rhs.Add_SKZ(tmp);
70:     return s;
71: }
```

```
1: #pragma once
2: #include <iostream>    // ostream
3: #include <string>
4: #include <set>
5: #include <vector>
6: //using namespace std;
7:
8: /** Klasse Student, welche nun mehrere SKZ speichert.
9: */
10: class Student
11: {
12: public:
13:     /** Default constructor */
14:     Student();
15:
16:     /// Constructor for a student with a single study ID
17:     ///
18:     /// @param[in] name      name of student
19:     /// @param[in] matrikel student ID
20:     /// @param[in] studium  study ID
21:     ///
22:     Student(const std::string& name, const std::string& matrikel, con
st int studium);
23:
24:     /// Constructor for a student with a single study ID
25:     ///
26:     /// @param[in] name      name of student
27:     /// @param[in] matnr     student ID
28:     /// @param[in] skz      vector of study IDs
29:     ///
30:     Student(const std::string & name, const std::string& matnr, const
std::vector<int>& skz);
31:
32:     // automatic generation of copy constructor and assignment operat
or by the compiler
33:     // in C++11: the Assignment operator (and any other method) can b
e explicetely deleted via:
34:     // Student& operator=(const Student& other) = delete;
35:     Student(const Student &orig) = default;
36:     Student(Student &&orig) = default;
37:     // const member '_matr_nr' ==> kein Zuweisungsoperator moeglich!
38:     Student& operator=(const Student &rhs) = delete;
39:     Student& operator=(Student &&rhs) = delete;
40:
41:
42:     /** destructor */
43:     //virtual
44:     ~Student() { }
45:
46:     /** Access name
47:      * \return The current value of name
48:      */
49:     std::string Getname() const
50:     {
51:         return _name;
52:     }
53:
```

```
54:  /** Set name
55:   * \param val New value to set
56:   */
57:  void Setname(const std::string &val) { _name = val; }
58:
59:  /** Access student ID
60:   * \return The current value of matr_nr
61:   */
62:
63:  std::string Getmatr_nr() const
64:  {
65:      return _matr_nr;
66:  }
67:
68:  /** Fuegt eine Studienkennzahl (SKZ) hinzu.
69:   * \param[in] skz_in  zusaetzliche SKZ
70:   * \warning  Es wird nicht ueberprueft, ob diese SKZ bereits gesp
eichert ist.
71:   *          Dies wuerde ein find oder ein unique erfordern.
72:   */
73:  void Add_SKZ(const int skz_in);
74:
75:  /** Entfernt eine Studienkennzahl (SKZ).
76:   * \param[in] skz_in  zu entfernende SKZ
77:   * \warning  Es wird nicht ueberprueft, ob diese SKZ mehrfach ges
peichert ist.
78:   */
79:  void Del_SKZ(const int skz_in);
80:
81:  /// Asks for the number of enrolled studies
82:  ///
83:  /// @return number of enrolled studies
84:  ///
85:  int num_Studies() const
86:  {
87:      return static_cast<int>(_skz.size());
88:  }
89:
90:  /// Gets the vector of all study IDs
91:  ///
92:  /// @return vector of all study IDs
93:  ///
94:  /// const std::vector<int>& Get_SKZ() const {return _skz;}
95:  const std::vector<int> Get_SKZ() const;
96:
97:  /** Ausgabeoperator fuer die Klasse.
98:   * \param[in] s  ein beliebiger Ausgabestrom
99:   * \param[in] rhs  die auszugebende Instanz
100:   */
101:  friend std::ostream& operator<<(std::ostream& s, const Student& r
hs);
102:
103:  /** Eingabeoperator fuer die Klasse.
104:   * \param[in] s  ein beliebiger Eingabestrom
105:   * \param[in] rhs  die einzugebende Instanz
106:   * \warning Hier wird nur eine SKZ eingegeben.
107:   */
```

```
108:     friend std::istream& operator>>(std::istream& s, Student& rhs);
109: protected:
110:
111: private:
112:     std::string _name;    ///  
Familiennamen der Studentin
113:     // const member ==> kein Zuweisungsoperator moeglich!
114:     const std::string _matr_nr; ///  
Matrikelnummer
115:     std::set<int> _skz; ///  
neu: beliebig viele SKZs koennen ge
speichert werden
116: };
```