```cpp
// BuchBibliothek.cpp : AUFGABE 2
//

#include "stdafx.h"
#include <iostream>
#include <string>
using namespace std;

enum Version {V1,V2,V3}; // Versionen: V1 = nur 5 Bücher im Array, V2 = Konstruktor ohne Zwangsabfrage der Parameter, V3 = dynamische Listenversion
Version ThisVersion = V3;

// ################ Klassendefinition: Buch ########################
class Buch {
private:
    string Titel;
    string Autor;
    int Seitenzahl;
    static int AnzahlBuecher;
public:
    Buch(); // Parameterloser Konstruktor
    Buch(string t, string a, int s) : Titel(t), Autor(a), Seitenzahl(s) {AnzahlBuecher++;}
    string getTitel() { return Titel; }
    string getAutor() { return Autor; }
    int getSeiten() { return Seitenzahl; }
    static int getAnzahlBuecher() {return AnzahlBuecher; }
};
int Buch::AnzahlBuecher = 0; // Initialisierung

// Konstruktor, je nach Version:
Buch::Buch() {
    if (ThisVersion!=V2) {
        cout << "Bitte geben Sie Titel, Autor und Seitenzahl zum neuen Buch ein:\n";
        cin.sync(); // denkt sonst, das letzte cin wäre bereits eine Eingabe gewesen...
        cout << "Zunaechst der Titel : "; getline(cin,Titel); // statt: cin >> Titel;
        cout << "Nun der Autor       : "; getline(cin,Autor); // statt: cin >> Autor;
        cout << "Und die Seitenzahl  : "; cin >> Seitenzahl;
        AnzahlBuecher++;
    }
}
// ######################## OBACHT: ################################################
// Funktion vorab deklarieren, da sie auch in der Klasse "Bibliothek" benötigt wird:
void listeBuecher(Buch* Buchliste[]);

// ################ Klassendefinition: Bibliothek ########################
class Bibliothek {
    struct Buecherliste {
        Buch* Leihbuch;  // Pointer auf BuchObjekt
        int InventarNr;  // Inventarnummer ab 10000
        bool verfuegbar; // true: vorhanden, false: ausgeliehen
    };
private:
    const string Name;
    int AnzahlBuecher; // NICHT static, weil jede Bibliothek ja selbst zählt...
    Buecherliste Liste[1000]; // wie in Aufgabe 1 gelernt: als Array aus Zeigern auf vorhandene Bücher
public:
    Bibliothek(string n) : Name(n) {AnzahlBuecher=0;}
    void Buch_beschaffen(Buch* Buchliste[]);
    void Buch_ausleihen();
    void Buch_zurueckgeben();
    void listeLeihe(bool Vorgang); // alternativ eine Funktion für beides (ausleihen und zurückgeben)
};
// ################## die Implementierung der Bibliotheks-Methoden: ##################
void Bibliothek::Buch_beschaffen(Buch* Buchliste[]) {
    // Liste der erfassten Bücher, Auswahl eines davon und aufnehmen als Zeiger mit Setzen der Inventarnummer und verfuegbar
    int Auswahl;
    listeBuecher(Buchliste); // die allgemeine Bücherliste anzeigen
    // jetzt noch Buchnummer aufnehmen
    cin >> Auswahl;
    Liste[AnzahlBuecher].Leihbuch = Buchliste[Auswahl]; // +- 1???
    Liste[AnzahlBuecher].InventarNr = 10000 + AnzahlBuecher;
```

```
71      Liste[AnzahlBuecher].verfuegbar = true;
72      AnzahlBuecher++;
73  }
74  void Bibliothek::Buch_ausleihen() {
75      // Liste der verfuegbaren Bücher, Auswahl eines davon und markieren als ausgeliehen
76  }
77  void Bibliothek::Buch_zurueckgeben() {
78      // Liste der ausgeliehenen Bücher, Auswahl eines davon und markieren als verfuegbar
79  }
80  void Bibliothek::listeLeihe(bool Vorgang) {
81      // alternativ eine Funktion für beides (ausleihen und zurückgeben)
82
83      int Anzahl=0,Auswahl,i=0;
84      cout << "BUECHERLISTE der " << Name << endl;
85      // eine while-Schleife verwenden und feststellen, ob ein Buch ausleihbar/
        zurueckgebbar ist!
86      while (i<AnzahlBuecher) {
87          if (Liste[i].verfuegbar == Vorgang) {
88              cout << i << " : " << Liste[i].Leihbuch->getAutor() << ": " << Liste[i].
        Leihbuch->getTitel();
89              cout << " (" << Liste[i].Leihbuch->getSeiten() << " Seiten, INr:" << Liste
        [i].InventarNr << ")\n";
90              Anzahl++;
91          }
92          i++;
93      };
94      if (Anzahl==0) // es gibt gar keine Bücher...
95          cout << "Keine Buecher vorhanden!\n";
96      else {
97          cout << "Welches Buch moechten Sie " << ((Vorgang)?"ausleihen":"zurueckgeben")
        << "? ";
98          cin >> Auswahl;
99          Liste[Auswahl].verfuegbar = !Vorgang; // Verfuegbarkeit invertieren...
100     }
101 }
102
103 // ################## die Funktionen für die Bücherverwaltung ########################
104
105 // listeBuecher: zeigt eine Liste aller erfassten Bücher am Bildschirm an (mit Nummer
        zum späteren Auswählen)
106
107 void listeBuecher(Buch* Buchliste[]) { // wir übergeben "das Buchlisten-Array", also
        den Zeiger auf das erste Buchobjekt!
108     cout << "BUECHERLISTE:\n";
109     if (Buch::getAnzahlBuecher()==0) // es gibt gar keine Bücher...
110         cout << "Noch keine Buecher vorhanden!\n";
111     else {
112         for (int i=0; i<Buch::getAnzahlBuecher(); i++) {
113             cout << i << " : " << Buchliste[i]->getAutor() << ": " << Buchliste[i]->
        getTitel();
114             cout << " (" << Buchliste[i]->getSeiten() << " Seiten)\n";
115         }
116     }
117 }
118
119 // neuesBuch: erzeugt ein neues Buch-Objekt und speichert den Zeiger auf das erzeugte
        Buch in unser Bücherlisten-Array
120
121 void neuesBuch(Buch* Buchliste[]) { // wir übergeben "das Buchlisten-Array", also den
        Zeiger auf das erste Buchobjekt!
122     Buch* B = new Buch; // ein neues Buch erzeugen ( und den Konstruktor nach den Daten
         fragen lassen)
123     Buchliste[Buch::getAnzahlBuecher()-1] = B; // den Zeiger auf das neue Buch in das
        Array eintragen (-1, weil die Anzahl ja bereits im K. erhöht wurde)
124 }
125 // ################## Ende der Funktionen für die Bücherverwaltung ###################
        ###
126
127
128 // ########################## Hier beginnt das Hauptprogramm: ########################
        ###
129 int main()
130 {
131     // Buch Buchliste[2000]; // würde uns 2000 mal nach den Buchdaten fragen -> das ist
         doof!
```

```
132     // Buch Buchliste[5]     // Version V1: fragt 5 mal nach den Buchdaten, das geht, trifft aber nicht den Sinn (neue Bücher anlegen, Array erweitern...)
133     Buch* Buchliste[2000];   // also ein Array von 2000 Pointern auf die Objekte "Buch"
134     Buchliste[0] =  NULL;    // mit dem ersten Pointer (es gibt noch kein Buch) auf NULL
135
136     Bibliothek Buecherei("Stadtbibliothek Kleinkleckersdorf"); // eine Bibliothek anlegen
137
138     char Eingabe, Auswahl;
139     do {
140         system("cls"); // Bildschirm löschen
141         cout << "BIBLIOTHEKSPROGRAMM\n";
142         cout << "1: Bibliothek verwalten\n";
143         cout << "2: Buecherliste verwalten\n";
144         cout << "3: Programm beenden\n";
145         cin >> Auswahl;
146         switch (Auswahl) {
147         case '1':
148             do {
149                 cout << "BIBLIOTHEKSVERWALTUNG\n";
150                 cout << "1: Neues Buch beschaffen\n";
151                 cout << "2: Buch ausleihen\n";
152                 cout << "3: Buch zurueckgeben\n";
153                 cout << "4: Bibliotheksverwaltung beenden\n";
154                 cin >> Eingabe;
155                 switch (Eingabe) {
156                 case '1':
157                     cout << "Welches Buch moechten Sie beschaffen:\n";
158                     Buecherei.Buch_beschaffen(Buchliste);
159                     break;
160                 case '2':
161                     cout << "Welches Buch moechten Sie ausleihen:\n";
162                     Buecherei.listeLeihe(true);
163                     break;
164                 case '3':
165                     cout << "Welches Buch moechten Sie zurueckgeben:\n";
166                     Buecherei.listeLeihe(false);
167                     break;
168                 case '4': break; // nur damit die Eingabe '4' keine Falscheingabe ist
169                 default: cout << "Falsche Eingabe (1...4)\n";
170                 }
171             } while (Eingabe !='4');
172             break;
173         case '2':
174             do {
175                 cout << "BUCHVERWALTUNG:\n";
176                 cout << "1: Neues Buch erfassen\n";
177                 cout << "2: Alle " << Buch::getAnzahlBuecher() << " Buecher auflisten\n";
178                 cout << "3: Buchverwaltung beenden\n";
179                 cin >> Eingabe;
180                 switch (Eingabe) {
181                 case '1': neuesBuch(Buchliste); break;
182                 case '2': listeBuecher(Buchliste); break;
183                 case '3': break; // nur damit die Eingabe '3' keine Falscheingabe ist
184                 default: cout << "Falsche Eingabe (1...3)\n";
185                 }
186             } while (Eingabe != '3');
187             break;
188         case '3': break; // nur damit die Auswahl '3' keine Falscheingabe ist
189         default: cout << "Falsche Eingabe (1...3)\n";
190         } // Ende main-switch
191     } while (Auswahl != '3');
192
193     return 0;
194 }
```