

Problem 1

Code:

```
problem1  bookType  get_title() const
1  #include "bookType.h"
2  #include <algorithm>
3  #include <iostream>
4  #include <string>
5
6  using namespace std;
7
8  // constructors
9  bookType::bookType() {
10     title = "default title";
11     authors[0] = "default author";
12     publisher = "default publisher";
13     isbn = "default isbn";
14     price = 1.0;
15     num_copies = 1;
16     num_authors = 1;
17 }
18
19 bookType::bookType(string title_arg, string authors_arg[], string publisher_arg, string isbn_arg, double price_arg, int num_copies_arg, int num_authors_arg)
20 {
21     title = title_arg;
22     copy(authors_arg[0], authors_arg[3], authors);
23     publisher = publisher_arg;
24     isbn = isbn_arg;
25     price = price_arg;
26     num_copies = num_copies_arg;
27     num_authors = num_authors_arg;
28 }
29
30 // destructor
31 bookType::~bookType() {}
32
33 // getters
34 string bookType::get_title() const {
35     return title;
36 }
37
38 string* bookType::get_authors() const {
39     string* authorsCopy = new string[4];
40     for (int i = 0; i < 4; i++) {
41         authorsCopy[i] = authors[i];
42     }
43     return authorsCopy;
44 }
45
46 string bookType::get_publisher() const {
47     return publisher;
48 }
```

```
mkcpp  bookType.cpp  bookType.h
problem1  bookType  get_title() const
49 }
50 double bookType::get_price() const {
51     return price;
52 }
53 int bookType::get_num_copies() const {
54     return num_copies;
55 }
56 int bookType::get_num_authors() const {
57     return num_authors;
58 }
59
60 // setters
61 void bookType::set_title(string title_arg) {
62     title = title_arg;
63 }
64 void bookType::set_authors(string authors_arg[]) {
65     copy(authors_arg[0], authors_arg[3], authors);
66 }
67 void bookType::set_publisher(string publisher_arg) {
68     publisher = publisher_arg;
69 }
70 void bookType::set_isbn(string isbn_arg) {
71     isbn = isbn_arg;
72 }
73 void bookType::set_price(double price_arg) {
74     price = price_arg;
75 }
76 void bookType::set_num_copies(int num_copies_arg) {
77     num_copies = num_copies_arg;
78 }
79 void bookType::set_num_authors(int num_authors_arg) {
80     num_authors = num_authors_arg;
81 }
82 void bookType::setBookInfo(string title_arg, string isbn_arg, string publisher_arg, string authors_arg[],
83 double price_arg, int num_copies_arg, int num_authors_arg) {
84     title = title_arg;
85     copy(authors_arg[0], authors_arg[3], authors);
86     publisher = publisher_arg;
87     isbn = isbn_arg;
88     price = price_arg;
89     num_copies = num_copies_arg;
90     num_authors = num_authors_arg;
91 }
92
93 }
```

```

93
94 // updaters
95 void bookType::update_num_copies(int change) {
96     num_copies += change;
97 }
98
99 // display
100 void bookType::display_title() const {
101     cout << "Title: " << title << endl;
102 }
103 void bookType::display_authors() const {
104     cout << "Authors: " << authors << endl;
105 }
106 void bookType::display_publisher() const {
107     cout << "Publisher: " << publisher << endl;
108 }
109 void bookType::display_isbn() const {
110     cout << "ISBN: " << isbn << endl;
111 }
112 void bookType::display_price() const {
113     cout << "Price: " << price << endl;
114 }
115 void bookType::display_num_copies() const {
116     cout << "Number of copies: " << num_copies << endl;
117 }
118 void bookType::display_num_authors() const {
119     cout << "Number of authors: " << num_authors << endl;
120 }
121
122 void bookType::printInfo() const {
123     display_title();
124     display_authors();
125     display_publisher();
126     display_isbn();
127     display_price();
128     display_num_copies();
129     display_num_authors();
130 }
131
132 ostream& operator << (ostream& osObject, const bookType& book1) {
133     osObject << "Title: " << book1.title << "Author: " << book1.authors[0] << "Publisher: " << book1.publisher
134     << "ISBN: " << book1.isbn << "Price: " << book1.price << "Number of copies: " << book1.num_copies << "Number of authors: " << book1.num_authors;
135 }

```

```

// check
bool bookType::isTitle(string s) const {
    return s == title;
}

bool bookType::isISBN(string s) const {
    return s == isbn;
}

bool bookType::isAuthor(string s) const {
    for (int i = 0; i < 4; i++) {
        if (s == authors[i]) {
            return true;
        }
    }
    return false;
}

bool bookType::isInStock() const {
    return num_copies > 0;
}

```

```
problem1 - bookType
1  #include <string>
2  #include <iostream>
3
4  using namespace std;
5
6  class bookType {
7
8      // overload cout
9      friend ostream& operator << (ostream& osObject, const bookType& book1);
10
11  public:
12
13      // constructors
14      bookType();
15      bookType(string title_arg, string authors_arg[], string publisher_arg, string isbn_arg,
16              double price_arg, int num_copies_arg, int num_authors_arg);
17
18      // destructor
19      ~bookType();
20
21      // getters
22      string get_title() const;
23      string* get_authors() const;
24      string get_publisher() const;
25      string get_isbn() const;
26      double get_price() const;
27      int get_num_copies() const;
28      int get_num_authors() const;
29
30      // setters
31      void set_title(string title_arg);
32      void set_authors(string authors_arg[]);
33      void set_publisher(string publisher_arg);
34      void set_isbn(string isbn_arg);
35      void set_price(double price_arg);
36      void set_num_copies(int num_copies_arg);
37      void set_num_authors(int num_authors_arg);
38      void setBookInfo(string title_arg, string isbn_arg, string publisher_arg, string author_arg[],
39                      double cost_arg, int num_copies_arg, int num_authors_arg);
40
41      // updaters
42      void update_num_copies(int change);
43
44      // display
45      void display_title() const;
46      void display_authors() const;
47      void display_publisher() const;
48      void display_isbn() const;
49      void display_price() const;
50      void display_num_copies() const;
51      void display_num_authors() const;
52      void printInfo() const;
53
54      // check
55      bool isTitle(string s) const;
56      bool isISBN(string s) const;
57      bool isAuthor(string s) const;
58      bool isInStock() const;
59
60  private:
61
62      string title;
63      string authors[4];
64      string publisher;
65      string isbn;
66      double price;
67      int num_copies;
68      int num_authors;
69
70  };
```

```
problem1 (Global Scope) main()
1 #include <iostream>
2 #include <vector>
3 #include <stack>
4 #include <queue>
5 #include "bookType.h"
6
7 using namespace std;
8
9 int main() {
10     // problem 1:
11     // declare array of 100 components of booktype
12     bookType myLibrary[100];
13     string authors1[] = { "Anna Wiener" };
14     bookType book1("Uncanny Valley", authors1, "MCD", "978-0-374-27801-4", 27.00, 1, 1);
15     string authors2[] = { "James Clear" };
16     bookType book2("Atomic Habits", authors2, "Penguin Random House", "978-0-7352-1129-2", 27.00, 1, 1);
17     string authors3[] = { "Nick Lane" };
18     bookType book3("The Vital Question", authors3, "W W Norton", "978-0-393-35297-9", 17.95, 1, 1);
19     myLibrary[0] = book1;
20     myLibrary[1] = book2;
21     myLibrary[3] = book3;
22     // search for a book by title
23     cout << "Enter the title of a book to find out whether it exists in the library: " << endl;
24     string title;
25     cin >> title;
26     for (int i = 0; i < 100; i++) {
27         if (myLibrary[i].isTitle(title)) {
28             cout << "Your book exists in the collection." << endl;
29         }
30     }
31     // search for book by isbn
32     cout << "Enter the isbn of a book to find out whether it exists in the library: " << endl;
33     string isbn;
34     cin >> isbn;
35     for (int i = 0; i < 100; i++) {
36         if (myLibrary[i].isISBN(isbn)) {
37             cout << "Your book exists in the collection." << endl;
38         }
39     }
40
41     // update num copies by isbn
42     cout << "Enter the isbn of the book in the library you wish to update: " << endl;
43     cin >> isbn;
44     cout << "Enter the number of copies you wish to update the book by: " << endl;
45     int num;
46     cin >> num;
47     for (int i = 0; i < 100; i++) {
48         if (myLibrary[i].isISBN(isbn)) {
49             myLibrary[i].update_num_copies(num);
50         }
51     }
52     return 0;
53 }
```

Output:

No output, couldn't run program despite no errors.

Problem 2:

Code:

Same bookType.cpp, bookType.h

```
h.cpp x
problem2 (Global Scope) main()
1 #include <iostream>
2 #include <vector>
3 #include <stack>
4 #include <queue>
5 #include "bookType.h"
6
7 using namespace std;
8
9 int main() {
10     string authors1[] = { "Anna Wiener" };
11     bookType book1("Uncanny Valley", authors1, "MCD", "978-0-374-27801-4", 27.00, 1, 1);
12     string authors2[] = { "James Clear" };
13     bookType book2("Atomic Habits", authors2, "Penguin Random House", "978-0-7352-1129-2", 27.00, 1, 1);
14     string authors3[] = { "Nick Lane" };
15     bookType book3("The Vital Question", authors3, "W W Norton", "978-0-393-35297-9", 17.95, 1, 1);
16     // store bookType objs in vector
17     vector<bookType> booksVector;
18     vector<bookType>::iterator itr1;
19     // use 5 methods of vector class: push_back, pop_back, insert, size, at
20     booksVector.push_back(book1);
21     booksVector.push_back(book2);
22     cout << "Size of books vector after adding 2 books: " << booksVector.size() << endl;
23     cout << "Books vector after adding 2 books: " << endl;
24     for (itr1 = booksVector.begin(); itr1 != booksVector.end(); ++itr1) {
25         cout << " " << *itr1 << endl;
26     }
27     booksVector.insert(booksVector.begin() + 1, book3);
28     cout << "Books vector after inserting book 3: " << endl;
29     for (itr1 = booksVector.begin(); itr1 != booksVector.end(); ++itr1) {
30         cout << " " << *itr1 << endl;
31     }
32     while (!booksVector.empty()) {
33         booksVector.pop_back();
34     }
35 }
```

```
35
36 // stack: push, pop, top, size, empty
37 stack<bookType> bookStack;
38 bookStack.push(book1);
39 bookStack.push(book2);
40 bookStack.push(book3);
41 cout << "The size of book stack after pushing 3 books: " << bookStack.size() << endl;
42 cout << "The books in the reverse order they were added: " << endl;
43 while (!bookStack.empty()) {
44     cout << bookStack.top() << endl;
45     bookStack.pop();
46 }
47
48 // queue: push, pop, size, front, empty
49 queue<bookType> bookQ;
50 bookQ.push(book1);
51 bookQ.push(book2);
52 bookQ.push(book3);
53 cout << "The size of book queue after pushing 3 books: " << bookQ.size() << endl;
54 cout << "The books in the order they were added: " << endl;
55 while (!bookQ.empty()) {
56     cout << bookQ.front();
57     bookQ.pop();
58 }
59 return 0;
60 }
```