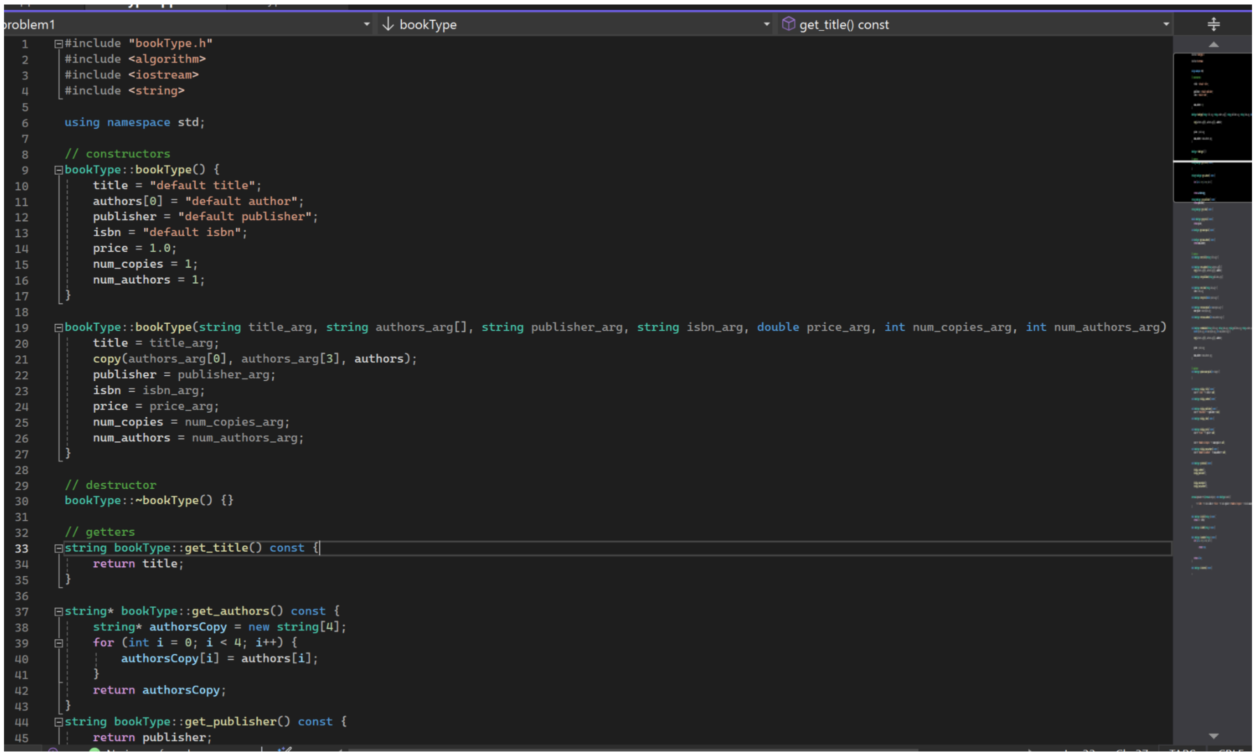
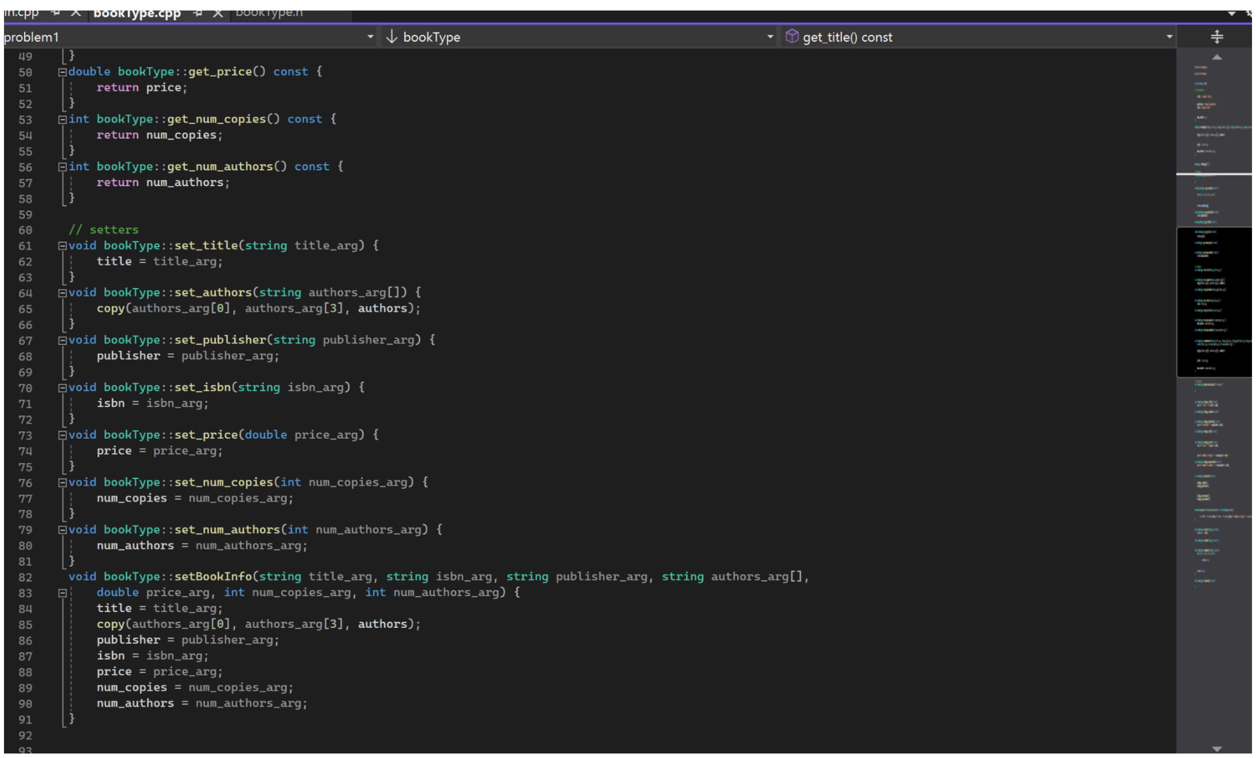
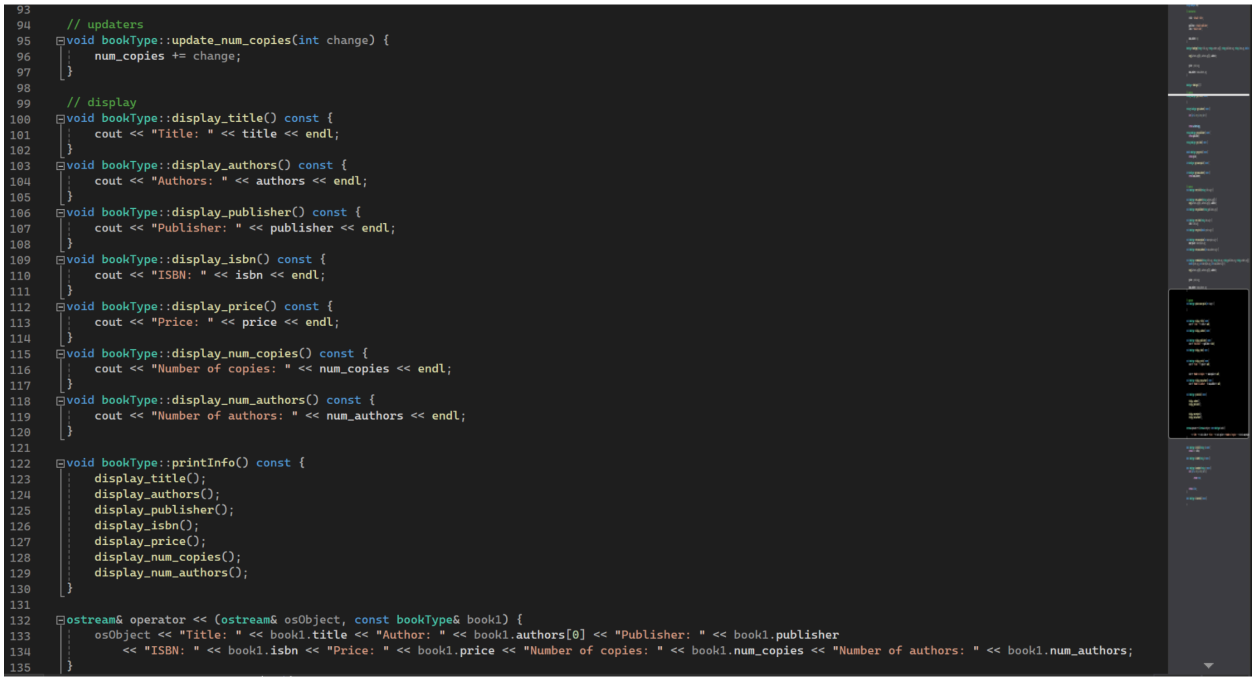
Problem 1
Code:



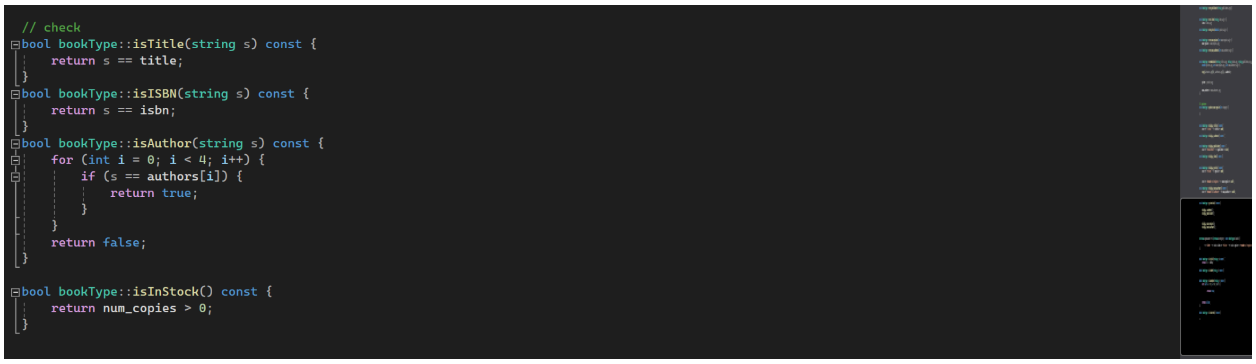
```cpp
#include "bookType.h"
#include <algorithm>
#include <iostream>
#include <string>

using namespace std;

// constructors
bookType::bookType() {
    title = "default title";
    authors[0] = "default author";
    publisher = "default publisher";
    isbn = "default isbn";
    price = 1.0;
    num_copies = 1;
    num_authors = 1;
}

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)
    title = title_arg;
    copy(authors_arg[0], authors_arg[3], authors);
    publisher = publisher_arg;
    isbn = isbn_arg;
    price = price_arg;
    num_copies = num_copies_arg;
    num_authors = num_authors_arg;
}

// destructor
bookType::~bookType() {}

// getters
string bookType::get_title() const {
    return title;
}

string* bookType::get_authors() const {
    string* authorsCopy = new string[4];
    for (int i = 0; i < 4; i++) {
        authorsCopy[i] = authors[i];
    }
    return authorsCopy;
}
string bookType::get_publisher() const {
    return publisher;
```



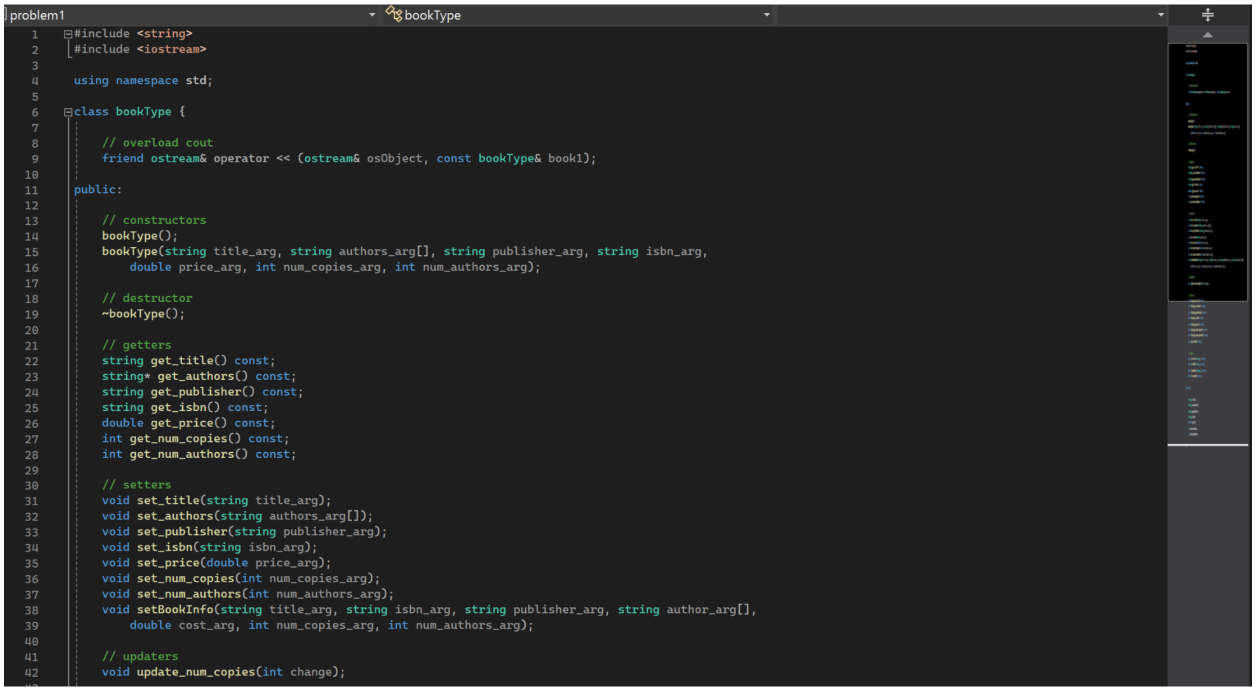
```cpp
}
double bookType::get_price() const {
    return price;
}
int bookType::get_num_copies() const {
    return num_copies;
}
int bookType::get_num_authors() const {
    return num_authors;
}

// setters
void bookType::set_title(string title_arg) {
    title = title_arg;
}
void bookType::set_authors(string authors_arg[]) {
    copy(authors_arg[0], authors_arg[3], authors);
}
void bookType::set_publisher(string publisher_arg) {
    publisher = publisher_arg;
}
void bookType::set_isbn(string isbn_arg) {
    isbn = isbn_arg;
}
void bookType::set_price(double price_arg) {
    price = price_arg;
}
void bookType::set_num_copies(int num_copies_arg) {
    num_copies = num_copies_arg;
}
void bookType::set_num_authors(int num_authors_arg) {
    num_authors = num_authors_arg;
}
void bookType::setBookInfo(string title_arg, string isbn_arg, string publisher_arg, string authors_arg[],
    double price_arg, int num_copies_arg, int num_authors_arg) {
    title = title_arg;
    copy(authors_arg[0], authors_arg[3], authors);
    publisher = publisher_arg;
    isbn = isbn_arg;
    price = price_arg;
    num_copies = num_copies_arg;
    num_authors = num_authors_arg;
}
```



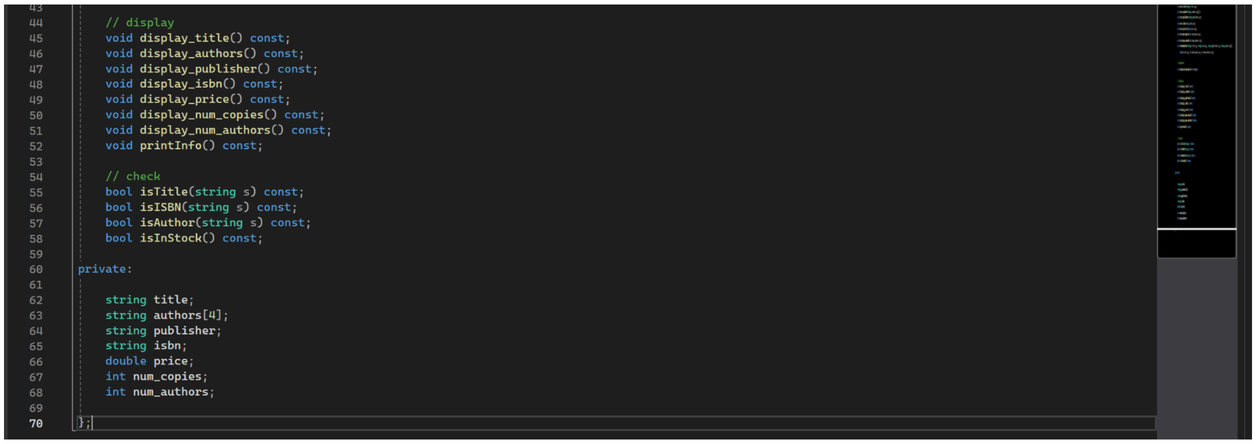
```cpp

// updaters
void bookType::update_num_copies(int change) {
    num_copies += change;
}

// display
void bookType::display_title() const {
    cout << "Title: " << title << endl;
}
void bookType::display_authors() const {
    cout << "Authors: " << authors << endl;
}
void bookType::display_publisher() const {
    cout << "Publisher: " << publisher << endl;
}
void bookType::display_isbn() const {
    cout << "ISBN: " << isbn << endl;
}
void bookType::display_price() const {
    cout << "Price: " << price << endl;
}
void bookType::display_num_copies() const {
    cout << "Number of copies: " << num_copies << endl;
}
void bookType::display_num_authors() const {
    cout << "Number of authors: " << num_authors << endl;
}

void bookType::printInfo() const {
    display_title();
    display_authors();
    display_publisher();
    display_isbn();
    display_price();
    display_num_copies();
    display_num_authors();
}

ostream& operator << (ostream& osObject, const bookType& book1) {
    osObject << "Title: " << book1.title << "Author: " << book1.authors[0] << "Publisher: " << book1.publisher
        << "ISBN: " << book1.isbn << "Price: " << book1.price << "Number of copies: " << book1.num_copies << "Number of authors: " << book1.num_authors;
}
```



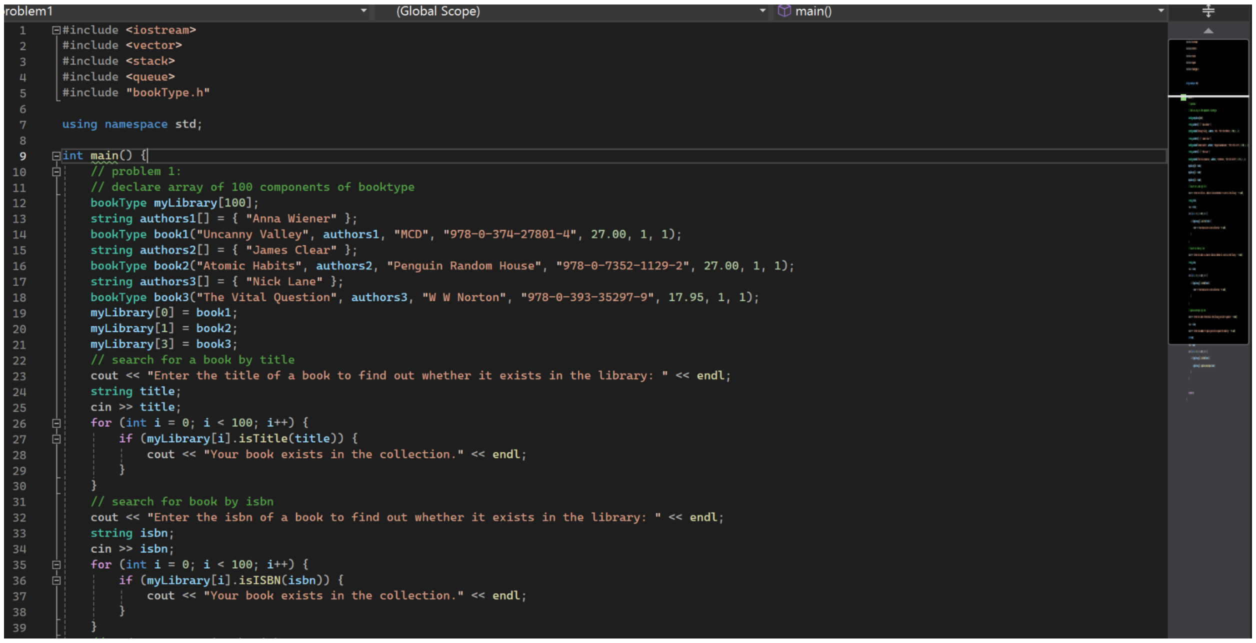
```cpp
// check
bool bookType::isTitle(string s) const {
    return s == title;
}
bool bookType::isISBN(string s) const {
    return s == isbn;
}
bool bookType::isAuthor(string s) const {
    for (int i = 0; i < 4; i++) {
        if (s == authors[i]) {
            return true;
        }
    }
    return false;
}

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



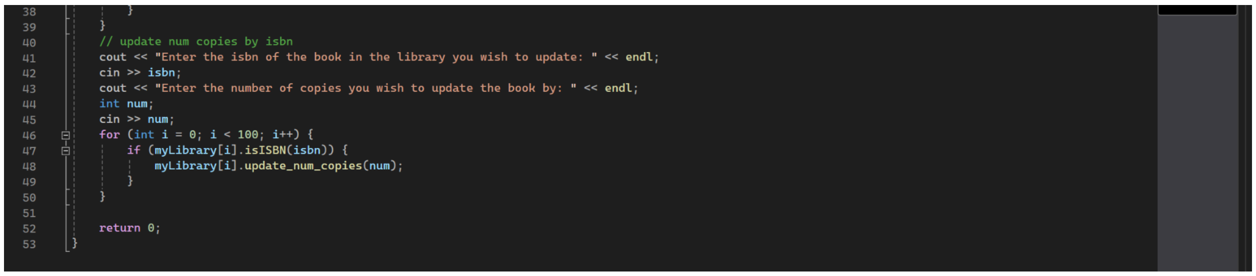
```cpp
#include <string>
#include <iostream>

using namespace std;

class bookType {

    // overload cout
    friend ostream& operator << (ostream& osObject, const bookType& book1);

public:

    // constructors
    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);

    // destructor
    ~bookType();

    // getters
    string get_title() const;
    string* get_authors() const;
    string get_publisher() const;
    string get_isbn() const;
    double get_price() const;
    int get_num_copies() const;
    int get_num_authors() const;

    // setters
    void set_title(string title_arg);
    void set_authors(string authors_arg[]);
    void set_publisher(string publisher_arg);
    void set_isbn(string isbn_arg);
    void set_price(double price_arg);
    void set_num_copies(int num_copies_arg);
    void set_num_authors(int num_authors_arg);
    void setBookInfo(string title_arg, string isbn_arg, string publisher_arg, string author_arg[],
        double cost_arg, int num_copies_arg, int num_authors_arg);

    // updaters
    void update_num_copies(int change);
```



```cpp

    // display
    void display_title() const;
    void display_authors() const;
    void display_publisher() const;
    void display_isbn() const;
    void display_price() const;
    void display_num_copies() const;
    void display_num_authors() const;
    void printInfo() const;

    // check
    bool isTitle(string s) const;
    bool isISBN(string s) const;
    bool isAuthor(string s) const;
    bool isInStock() const;

private:

    string title;
    string authors[4];
    string publisher;
    string isbn;
    double price;
    int num_copies;
    int num_authors;

};
```



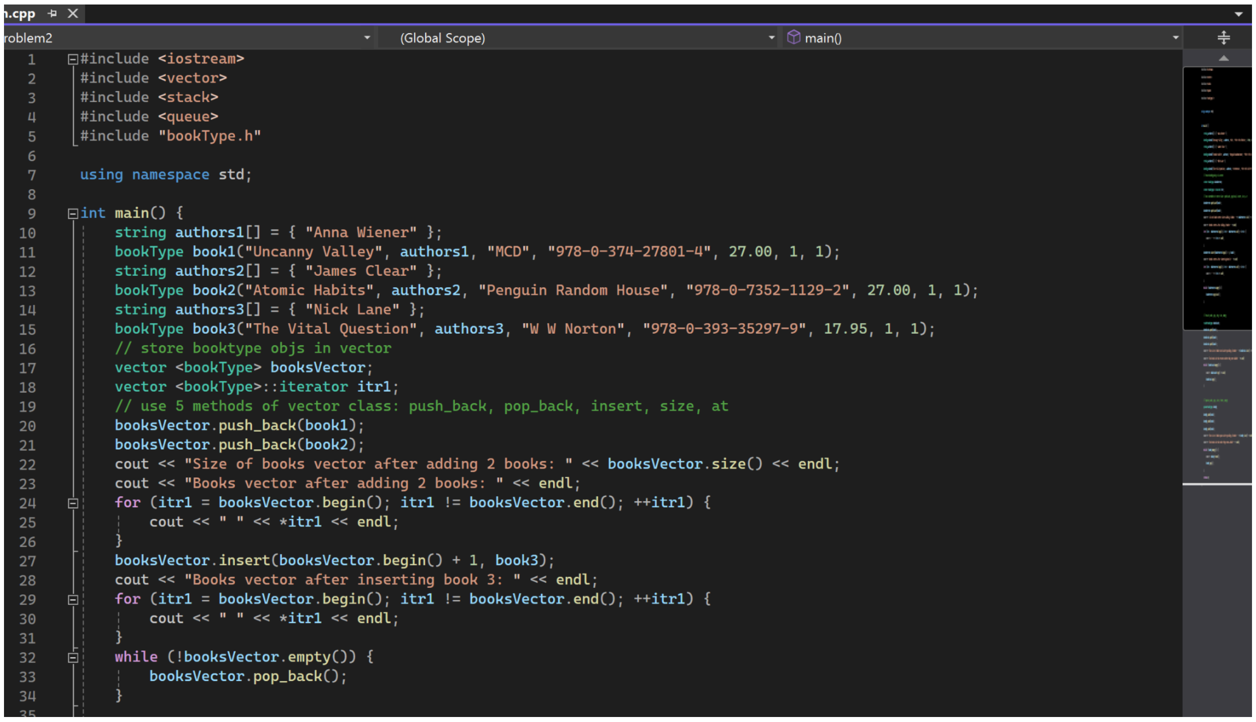
```cpp
#include <iostream>
#include <vector>
#include <stack>
#include <queue>
#include "bookType.h"

using namespace std;

int main() {
    // problem 1:
    // declare array of 100 components of booktype
    bookType myLibrary[100];
    string authors1[] = { "Anna Wiener" };
    bookType book1("Uncanny Valley", authors1, "MCD", "978-0-374-27801-4", 27.00, 1, 1);
    string authors2[] = { "James Clear" };
    bookType book2("Atomic Habits", authors2, "Penguin Random House", "978-0-7352-1129-2", 27.00, 1, 1);
    string authors3[] = { "Nick Lane" };
    bookType book3("The Vital Question", authors3, "W W Norton", "978-0-393-35297-9", 17.95, 1, 1);
    myLibrary[0] = book1;
    myLibrary[1] = book2;
    myLibrary[3] = book3;
    // search for a book by title
    cout << "Enter the title of a book to find out whether it exists in the library: " << endl;
    string title;
    cin >> title;
    for (int i = 0; i < 100; i++) {
        if (myLibrary[i].isTitle(title)) {
            cout << "Your book exists in the collection." << endl;
        }
    }
    // search for book by isbn
    cout << "Enter the isbn of a book to find out whether it exists in the library: " << endl;
    string isbn;
    cin >> isbn;
    for (int i = 0; i < 100; i++) {
        if (myLibrary[i].isISBN(isbn)) {
            cout << "Your book exists in the collection." << endl;
        }
    }
```



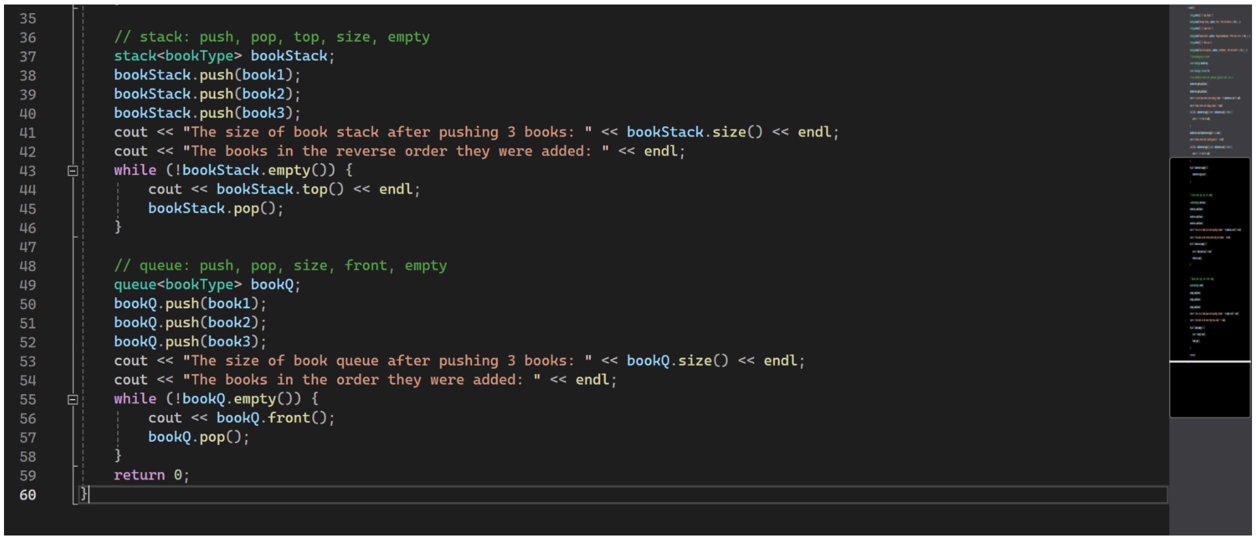
```cpp
        }
    }
    // update num copies by isbn
    cout << "Enter the isbn of the book in the library you wish to update: " << endl;
    cin >> isbn;
    cout << "Enter the number of copies you wish to update the book by: " << endl;
    int num;
    cin >> num;
    for (int i = 0; i < 100; i++) {
        if (myLibrary[i].isISBN(isbn)) {
            myLibrary[i].update_num_copies(num);
        }
    }

    return 0;
}
```

Output:
No output, couldn’t run program despite no errors.

Problem 2:
Code:
Same bookType.cpp, bookType.h



```cpp
#include <iostream>
#include <vector>
#include <stack>
#include <queue>
#include "bookType.h"

using namespace std;

int main() {
    string authors1[] = { "Anna Wiener" };
    bookType book1("Uncanny Valley", authors1, "MCD", "978-0-374-27801-4", 27.00, 1, 1);
    string authors2[] = { "James Clear" };
    bookType book2("Atomic Habits", authors2, "Penguin Random House", "978-0-7352-1129-2", 27.00, 1, 1);
    string authors3[] = { "Nick Lane" };
    bookType book3("The Vital Question", authors3, "W W Norton", "978-0-393-35297-9", 17.95, 1, 1);
    // store booktype objs in vector
    vector <bookType> booksVector;
    vector <bookType>::iterator itr1;
    // use 5 methods of vector class: push_back, pop_back, insert, size, at
    booksVector.push_back(book1);
    booksVector.push_back(book2);
    cout << "Size of books vector after adding 2 books: " << booksVector.size() << endl;
    cout << "Books vector after adding 2 books: " << endl;
    for (itr1 = booksVector.begin(); itr1 != booksVector.end(); ++itr1) {
        cout << " " << *itr1 << endl;
    }
    booksVector.insert(booksVector.begin() + 1, book3);
    cout << "Books vector after inserting book 3: " << endl;
    for (itr1 = booksVector.begin(); itr1 != booksVector.end(); ++itr1) {
        cout << " " << *itr1 << endl;
    }
    while (!booksVector.empty()) {
        booksVector.pop_back();
    }
```



```cpp

    // stack: push, pop, top, size, empty
    stack<bookType> bookStack;
    bookStack.push(book1);
    bookStack.push(book2);
    bookStack.push(book3);
    cout << "The size of book stack after pushing 3 books: " << bookStack.size() << endl;
    cout << "The books in the reverse order they were added: " << endl;
    while (!bookStack.empty()) {
        cout << bookStack.top() << endl;
        bookStack.pop();
    }

    // queue: push, pop, size, front, empty
    queue<bookType> bookQ;
    bookQ.push(book1);
    bookQ.push(book2);
    bookQ.push(book3);
    cout << "The size of book queue after pushing 3 books: " << bookQ.size() << endl;
    cout << "The books in the order they were added: " << endl;
    while (!bookQ.empty()) {
        cout << bookQ.front();
        bookQ.pop();
    }
    return 0;
}
```