

Простая декомпозиция. В этом случае наследование.

Класс Item

```
#ifndef ITEM_H /* Если имя ещё не определено */
#define ITEM_H /* Определить имя */

#include <string>

using namespace std;

class Item {
private:
    // Инвентарный номер.
    int invNumber;
    // Взято на руки - 0; имеется в наличии - 1.
    bool taken;
public:
    // Конструктор по умолчанию.
    Item()
        : invNumber(0), taken(1){};

    // Конструктор инициализации.
    Item(int invNumber, bool taken)
        : invNumber(invNumber), taken(taken) {};

    // Показать информацию о единице хранения.
    virtual void show();

    // Есть ли единица хранения в наличии?
    bool is_available();

    // Возвращает инвентарный номер.
    int get_inv_number();

    // Операция "взять".
    void take();

    // Операция "вернуть".
    void back();
};
```

Класс Book

```
#ifndef BOOK_H /* Если имя ещё не определено */
#define BOOK_H /* Определить имя */

#include "item.h"
#include <string>

using namespace std;

class Book: public Item {
private:
    // Имя автора.
    string author;
    // Название книги.
    string title;
    // Название издательства.
    string publisher;
    // Год издания.
    unsigned year;
public:
    // Конструктор по умолчанию.
    Book()
        : author(""), title(""), publisher(""), year(0) {};

    // Конструктор инициализации.
    Book(int invNumber, bool taken, string author, string title, string publisher, unsigned year)
        : Item(invNumber, taken), author(author), title(title), publisher(publisher), year(year) {};

    // Возвращает имя автора.
    string get_author();

    // Возвращает название книги.
    string get_title();

    // Возвращает название издательства.
    string get_publisher();
};
```

