Input:

[number_of_book] *//following #number_of_book rows*

[book_author] [book_subject]

…

[number_of_user] //following #number_of_user rows

[user_type] [user_name] [predefined_borrow_book_number]*/*There will be a predefined number if user_type is borrower.*/*

…

//command loop

[Function]

Function and Output:

//Don't enter "input:" or print "output:" String

1. [user_name1] addBook

input: [book_author] [book_subject]

output: Borrower can not add book*//if [user_name1] is a borrower*

2. [user_name1] removeBook [book_id]

output: Borrower can not remove book*//if [user_name1] is a borrower*

3. [user_name1] checkout [user_name2]

input: [book1_id] [book2_id]…

output: Borrower can not check out the books *//if [user_name1] is a borrower*

output: Can not check out since the number of books exceed the limitation of user can check-out /**if amount of books more than[predefined_borrow_book_number]of [user_name2]**/

output: Can not check out since the book is checked out /*if the book is checked out*/

4. [user_name1] return [book_id]

output: Borrower can not return book// *if [user_name1] is a borrower*

output: Can not return since the book isn't checked out /*if the book is checked out

5. [user_name1] listAuthor [book_author]

output: ID: [book_id] Author: [book_author] Subject: [book_subject]

…

6. [user_name1] listSubject [book_subject]

output: ID: [book_id] Author: [book_author] Subject: [book_subject]

…

7. [user_name1] findChecked [user_name2]

output: Borrower can not find books checked out by other users /**if [user_name1] is a borrower and [user_name1] is not [user_name2]**/

output: ID: [book_id] Author: [book_author] Subject: [book_subject]//*else*

…

8. [user_name1] findBorrower [book_id]

output: User: [user_name]

output: Borrower can not find borrower //if [user_name1] is a borrower

Comment:

[user_type] must be one of following:

Staff

Borrower

[book_id] is integer and increases from 0 with sequential order form input.

[predefined_borrow_book_number] limit amount of books user can borrow at one time, not total.

If some books can't be borrowed since the book is checked out, the others still can be borrowed as long as they satisfy the constraints.

format:

output: ID: [book_id] Author: [book_author] Subject: [book_subject]

…

should be shown with increasing [book_id]

You are asked to write a main function in Class Main

We'll test your program through "java Main inputFile"

e.g java Main sampleInput

And show output to standard output.

Sample input and output are in the folder.

Please zip your source code and upload it.

The folder structure should be:

unzip [your_name].zip

=> [dir] [your_name]

=> [your_name]/*.java

```
// sampleInput file

3

AuthorA SubjectA

AuthorB SubjectB

AuthorC SubjectC

2

Staff UserA

Borrower UserB 2

UserA addBook

AuthorD SubjectD

UserA removeBook 3

UserA checkout UserB

0 1

UserA return 0

UserA listAuthor AuthorA

UserA listSubject SubjectA

UserA findChecked UserB

UserA findBorrower 1
```

```
// sampleOutput file

ID: 0 Author: AuthorA Subject: SubjectA

ID: 0 Author: AuthorA Subject: SubjectA

ID: 1 Author: AuthorB Subject: SubjectB

User: UserB
```