
## دانشکده ریاضی و علوم کامپیوتر

## برنامه‌سازی پیشرفته و کارگاه

# تمرین Exception Handling

استاد درس

**دکتر مهدی قطعی**

استاد دوم

**بهنام یوسفی مهر**

نگارش

**یونس کاظمی مقدم**

**اسفند ۱۴۰۳**

# فهرست

# مقدمه

این تمرین قراره کمک کنه که **Exception Handling** رو توی جاوا بهتر یاد بگیرین. می‌خوایم یه **سیستم مدیریت کتابخونه** بسازیم که توی اون بتونین کتاب اضافه کنین، کتاب قرض بگیرین و کتاب رو پس بدین. ولی خب، همیشه همه چی خوب پیش نمی‌ره؛ ممکنه بخواین یه کتابی رو بگیرین که تو کتابخونه نیست، یا کتابخونه خالی باشه و بخواین لیست کتاب‌ها رو داشته باشین! اینجاست که باید با **Custom Exceptions** و **try-catch** تو جاوا کار کنین تا بتونین برنامه‌ای بنویسین که کرش نکنه و پیام‌های درستی به کاربر بده.

راستی حتما قبل از شروع به نوشتن این تمرین، قسمت **«چند نکته»** در پایان این داکیومنت رو هم بخونین.

# شرح کلی تمرین و معرفی کلاس‌ها

در این قسمت، کلاس‌های مختلف پروژه و نقش هر کدوم رو توضیح می‌دیم.

## کلاس های Exception

در این تمرین سه کلاس Exception اختصاصی داریم: BookNotFoundException و EmptyLibraryException و InvalidBookException.

توجه کنین که هر یک از این کلاس‌ها باید در یک فایل جداگانه نوشته بشن.

همچنین هر یک از این کلاس‌ها از کلاس Exception ارث‌بری میکنن.

در تمامی این کلاس‌ها باید کانستراکتوری که ورودی آن تنها یک String هست نوشته شه (ترجیحا نام این String رو message بذارین). در بدنه این کانستراکتور، کانستراکتور والد با آرگومان message، فراخوانی می‌شه.

## کلاس Book

این کلاس مسئول نگهداری اطلاعات مربوط به یه کتابه.

### فیلدها

**private String title:** عنوان کتاب

**private int pageCount:** تعداد صفحات کتاب

### متدها

**public Book(String title, int pageCount):** کانستراکتور کلاس هست که یک کتاب رو با نام title و تعداد صفحات pageCount ایجاد میکنه.

دقت کنین که اگه نام کتاب خالی بود[1] باید یک IllegalArgumentException با پیام *Title cannot be empty* رو throw کنین.

[1] یک title رو میگیم خالی هروقت برابر null باشه یا یک رشته تهی باشه.

همچنین اگه تعداد صفحات کتاب صفر یا منفی بود باید یک IllegalArgumentException با پیام *Page count must be positive* رو throw کنین.

در صورتی که نام و تعداد صفحات کتاب مشکلی نداشتن، فیلدهای مربوطه، با این مقادیر مقداردهی می‌شن.

**تمام فیلدهای private باید getter با نام مناسب داشته باشن[2].**

**public String toString():** متد toString باید Override بشه. به این شکل که اگه کتابی با عنوان Sample Book دارای 100 صفحه هست و متد toString روی اون صدا زده بشه، باید خروجی متد مطابق زیر باشه:

Sample Book (100 pages)

## کلاس Library

این کلاس به عنوان مدیریت‌کننده‌ی کتاب‌ها عمل می‌کنه و تمام کتاب‌های موجود تو کتابخونه رو نگه می‌داره.

### فیلدها

**private ArrayList<Book> books:** کتاب‌های موجود در کتابخونه

### متدها

**public Library():** کانستراکتور کلاس هست که توی اون آبجکت books ساخته میشه (new میشه).

**public void addBook(Book book) throws InvalidBookException:** این متد سعی میکنه کتاب book رو به لیست books اضافه کنه. اگه آبجکت book برابر null باشه باید یک InvalidBookException با پیام *Book should not be null.* رو throw کنین.

در نهایت، کتاب book رو به لیست books اضافه کنین.

[2] منظور از نام مناسب این هست که اگه مثلا اسم یه فیلد sample هست، باید اسم متد getter اون getSample باشه.

**private Book findBook(String title):** این متد به دنبال کتابی با عنوان title در لیست books می‌گرده. اگه چنین کتابی وجود داشت، اون کتاب رو برمی‌گردونه و در غیراینصورت null رو بر می‌گردونه.

**public void borrowBook(String title) throws BookNotFoundException, EmptyLibraryException:** این متد سعی میکنه کتابی با نام title رو امانت بده.

در ابتدا اگه لیست books خالی بود باید یک EmptyLibraryException با پیام *Library is empty, no books to borrow* رو throw کنین.

سپس با صدا زدن متد findBook به دنبال کتابی با عنوان title در لیست books می‌گردیم. اگه چنین کتابی وجود نداشت باید یک BookNotFoundException با پیام *Book with title " + title + " not found.* رو throw کنین.

در نهایت اگه مشکلی نبود، پیامی مبنی بر موفقیت‌آمیز بودن امانت گرفتن کتاب به همراه مشخصات کتاب چاپ می‌شه.

**public void returnBook(String title) throws BookNotFoundException:** این متد سعی می‌کنه کتابی که دارای عنوان title هست رو برگردونه.

در ابتدا با صدا زدن متد findBook بررسی میشه آیا کتابی با این عنوان در لیست books وجود داره یا نه. اگه وجود نداشت باید یک BookNotFoundException با پیام *Cannot return. Book with title " + title + " not found.* رو throw کنین.

اگر چنین کتابی وجود داشت، پیامی مبنی بر موفقیت‌آمیز بودن برگرداندن کتاب به همراه مشخصات کتاب چاپ می‌شه.

**public void listBooks() throws EmptyLibraryException:** این متد سعی می‌کنه اطلاعات تمام کتاب‌های موجود در کتابخونه رو چاپ کنه.

در ابتدا اگه لیست books خالی بود باید یک EmptyLibraryException با پیام *Library is empty.* رو throw کنین.

اگه books خالی نبود، ابتدا پیام *Books in the library:* چاپ می‌شه و در خطوط بعدی کتاب‌های موجود در کتابخونه چاپ میشن.

**تمام فیلدهای private باید getter با نام مناسب داشته باشن.**

## کلاس Main

در کلاس Main به متد main (که جداگانه در اختیارتون قرار داده شده) باید بلوک‌های try-catch رو به شکلی اضافه کنین که با اجرای برنامه، تمام exception های throw شده، catch بشن و اصطلاحا برنامه کرش نکنه.

دقت کنین که از کلاس Main نباید کدی رو حذف کنین یا ترتیب خطوط رو تغییر بدین (البته میتونین بین خطوط کد هایی رو اضافه کنین).

همچنین متد main نباید هیچ اکسپشنی رو throw کنه.

# چند نکته

1. همونطور که در طول تمرین هم گفته شد، در همه کلاس‌ها باید تمام فیلدهای private دارای getter با نام مناسب باشن.

2. در طول تمرین هر جا گفته شده باید مشخصات یک کتاب چاپ بشه، **حتما متد toString رو روی اون آبجکت کتاب صدا بزنین** (در این تمرین از getter ها برای گرفتن عنوان و تعداد صفحات کتاب استفاده نکنین). مثلا به یکی از دو صورت زیر:

```
System.out.println("This is a sample text. " + book1.toString());
// System.out.println("This is a sample text. " + book1);
```

3. بلوک‌های try-catch باید به شکلی نوشته بشن که مانع اجرای دستورات مستقل از اون نشن. مثلا اگه داریم 10 تا کتاب رو به کتابخونه اضافه می‌کنیم و در اضافه کردن کتاب اول اکسپشنی throw شد، اینطور نباشه که بیخیال 9 تا کتاب دیگه بشیم (صرفا اون کتاب اول اضافه نمیشه. 9 خط دیگه باید اجرا بشن). به‌طور خلاصه، هر دستوری که احتمال میدین منجر به throw کردن اکسپشن بشه رو در بلوک try-catch جداگانه‌ای قرار بدین.

4. بدنه بلوک catch باید به‌صورت زیر باشه (منظور این هست که فقط پیام چاپ بشه):

```
catch (SampleException e){
    System.out.println(e.getMessage());
}
```

5. اکسپشن بلوک catch رو به صورت زیر ننویسین 😊 (اکسپشن مورد نظر رو به‌صورت دقیق بنویسین. نوشتن خود کلاس Exception که والد تمام اکسپشن‌هاست قابل قبول نیست).

```
catch (Exception e){
    // Some stuff
}
```

6. در زیر یک نمای کلی از کلاس‌ها آمده است:


Library
Library()
books ArrayList<Book>
returnBook(String) void
borrowBook(String) void
listBooks() void
addBook(Book) void
findBook(String) Book?
getBooks() ArrayList<Book>
Book
Book(String, int)
pageCount int
title String
getPageCount() int
getTitle() String
toString() String
EmptyLibraryException
EmptyLibraryException(String)
InvalidBookException
InvalidBookException(String)
BookNotFoundException
BookNotFoundException(String)
Main
Main()
main(String[]) void