## ❸ ソートキーを指定して検索結果を並べ替える

検索結果を特定のフィールドをキーにして並べ替えることができます。Sort.by( ) メソッドの引数に、並べ替えのキーに指定するフィールド名を指定するだけです。

```
find("genre='NOVEL' and price<=2000", Sort.by(price).list();
```

デフォルトでは昇順に並び替えますが、**Direction.Descending** を引数に追加すると、降順にすることができます。

```
find("genre='NOVEL' and price<=2000", Sort.by(price, Direction.Descending)).list();
```

さらに、ソートの 2 次、3 次のキーを and( ) を連結して指定できます。次は、著者名順の中で、さらに価格順に並び替えます。

```
find("genre='NOVEL' and price<=2000", Sort.by(author).and(price)).list();
```

## ❹ ページングにより検索結果の一部だけを取得する

検索結果の件数が多すぎると、全部を一度に表示することはできません。検索の速度も遅くなり、メモリーも大量に消費します。

そこで、1 度に表示できる件数、つまり、1 ページ分の件数を決めておいて、「3 ページ目のデータ」のように、指定したページのデータだけを取得するのが普通です。これを**ページング**といいます。

1 ページ分を 10 件として、その 1 ページ目だけを取得するには次のようにします。ページ番号は 0 から始まることに注意してください。

```
find("genre='NOVEL'", Sort.by("author")).page(Page.of(0, 10)).list();
```

二重引用符と括弧を挿入します

また、**pageCount( )** メソッドを使うと、次のように、全体のページ数も取得できます。

```
int pages = find("genre='NOVEL'").page(Page.ofSize(10)).pageCount(); // 全ページ数
```

二重引用符を挿入します

全体のページ数と現在のページ位置を管理すれば、「次のページへ」とか、「○ページ目へ」のように、表示するページ位置を移動するプログラムも容易に作成できます。