目次へ

HTML生成ファイナル・指南編

例によって、いただいた回答に色々コメントをさしあげたりします。説明の都合のために掲載の順序が必ずしも提出の順序と同じではありませんのでご了承くださいませ。

最初の方

できた。

```
# coding: cp932
o=open("mybookslist.html", "w")
print>>o, """<html>
<head>
<metadata http-equiv="Content-Type" content="text/html; charaset=shift_jis">
<title>蔵書リスト</title>
</head>
<body>
<h1>蔵書リスト</h1>
<table border="1">"""

book_dic={}
infile=open("lendingbooks.txt")
for line in infile:
  a=line[:-1].split("¥t")
  book_dic[a[0]]=[a[1], a[2]]
infile.close()


infile=open("mybooks.txt")
for line in infile:
  a=line[:-1].split("¥t")
  b_no=a[0]
  if book_dic.has_key(b_no):
     if a[1]==book_dic[b_no][0]:
        if a[2]==book_dic[b_no][1]:
           print >>o, "<tr bgcolor=yellow>"
           print >>o, "<td>%s</td><td>%s</td><td>%s</td>" % (b_no, book_dic[b_no][0],
 book_dic[b_no][1])
           print >>o, "</tr>"
        else:
```

```
            print >>o, "<tr bgcolor=yellow>"
            print >>o, "<td>%s</td><td>%s</td><td><font color=red>%s</font></td>" % (b_no,
 book_dic[b_no][0], book_dic[b_no][1])
            print >>o, "</tr>"

     else:
        if a[2]==book_dic[b_no][1]:
            print >>o, "<tr bgcolor=yellow>"
            print >>o, "<td>%s</td><td><font color=red>%s</font></td><td>%s</td>" % (b_no,
 book_dic[b_no][0], book_dic[b_no][1])
            print >>o, "</tr>"
        else:
            print >>o, "<tr bgcolor=yellow>"
            print >>o, "<td>%s</td><td><font color=red>%s</font></td><td><font color=red>
%s</font></td>" % (b_no, book_dic[b_no][0], book_dic[b_no][1])
            print >>o, "</tr>"

  else:
    print >>o, "<tr>"
    print >>o, "<td>%s</td><td>%s</td><td>%s</td>" % (b_no, a[1], a[2])
    print >>o, "</tr>"

infile.close()
print>>o, """</table>
</body>
</html>"""

o.close()
```

お疲れ様です。筆者の環境でも動作することを確認しました。順にスクリプトを拝見します。

まず、mybooklist.txtを書き込み用に開きますよということを宣言しています。で、HTMLのヘッダ等、データによって変化しない部分を固定の文字列としてまず出力しています。<table>タグまで固定ですから、これでよいですね。"""記号を使った、複数行にわたる文字列を扱うやりかたもこれでよいです。

で、貸出中の本のデータを book_dic 辞書に読み込んでいます。lendingbook.txt を一行ずつ読み込み、行末の一文字を取り除いてからタブ記号でバラしてリストaに格納。aの中は[本コード, 著者, タイトル] となっているはずですから、辞書に格納するときに本コードはキーに使い、著者とタイトルはこれを改めてリストにしてから、辞書の値として一度に格納しています。

で、貸出中のデータを全部読みこんだら、次に蔵書データ mybooks.txt の処理に入ります。

蔵書データを一行読み込むごとに、まずb_noという変数に本コードを入れています。今後何度も使うものなので、こうして変数に格納しておいて全体の記述を短く済ませることができるからいいですね。

で、いよいよ、ifが何重にも錯綜した難しい部分に入ります。

一番外側のifは、もしbook_dic辞書に該当本コードのキーがあったら、ということで、つまり「貸出中」だったら、という意味ですね。貸出中だったらその直後にさらに難しいifに突入して、貸出中でなかったら、字下げが同じのelse、つまり一番下のelseに大ジャンプして、貸出中でないときの表示（いわば貸出可能状態かな）を行います。

貸出中だったときの表示は、全部で4通りあるとこのスクリプトでは想定されています。すなわち、

- 著者名も書名も、R氏のデータとボスのデータで食い違いはない
- 著者名はR氏のデータと同じだが、書名は違っているようだ
- 著者名がR氏のデータと違うが、書名は同じ
- 著者名も書名も、ボスのデータとR氏のデータでは違っている

です。スクリプト内のifとelseの組み合わせで問題ないでしょう。

4つのパターンですべて行の背景色をyellowに指定してくれたようですね。今回のようなときは、ifのゴチャゴチャが始まる前に一か所だけ書いてしまえば少し楽になったはずですよ。下の例みたいに。

```
  if book_dic.has_key(b_no):
     print >>o, "<tr bgcolor=yellow>"     <---ココラヘン
     if a[1]==book_dic[b_no][0]:
     ...
```

同様に、</tr>を書きだす記述も4ヶ所（いや、貸出可能のときも含めると5ヶ所か）ありますが、if分の外にうまく置くと、一回だけの記述で済みますよ。下のようになるかな。

```
    ...
  else:
    print >>o, "<tr>"
    print >>o, "<td>%s</td><td>%s</td><td>%s</td>" % (b_no, a[1], a[2])
  print >>o, "</tr>"     <----ココラヘン
```

ともあれ、全体としてはお見事です。

つぎの方

できたよ。

```
# coding: cp932

lend_data = {}

# R氏への貸し出し中データを読み込む
inputfile1 = open("lendingbooks.txt")
for line in inputfile1:
    tmp = line[:-1].split("¥t")
    lend_data[ tmp[0] ] = [ tmp[1], tmp[2] ]
    # キー：蔵書番号 ⇒ 値：リスト[著者名, タイトル]
inputfile1.close()

# HTML生成

o = open("mybooks.html", "w")
print >>o, """<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html lang="ja">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=shift_jis">
<meta http-equiv="Content-Style-Type" content="text/css">
<title>わたしの蔵書リスト</title>
<style type="text/css">
<!--
.checkout { background-color: yellow; }
.kuitigai { color: red; }
table, th, td { border-width: 1px;
                border-color: silver; }
```

```
table { border-style: ridge;
        border-collapse: separate; }
th, td    { border-style: inset; }
-->
</style>
</head>
<body>
<h1>わたしの蔵書リスト</h1>
<table summary="わたしの蔵書について、蔵書番号、著者名、タイトルを1冊ずつ表示">
<col><col><col>
<thead>
<tr><th>蔵書番号</th><th>著者名</th><th>タイトル</th></tr>
</thead>
<tbody>
"""

inputfile2 = open("mybooks.txt")
for line in inputfile2:
    tmp = line[:-1].split("¥t")
    # tmp[0]蔵書番号、tmp[1]著者名、tmp[2]タイトル

    if lend_data.has_key(tmp[0]):
        if lend_data[tmp[0]][0] == tmp[1]: # 著者名比較
            author = tmp[1]
            diff_auth = ""
        else:
            author = lend_data[tmp[0]][0]
            diff_auth = ' class="kuitigai"'

        if lend_data[tmp[0]][1] == tmp[2]: # タイトル比較
            title = tmp[2]
            diff_ttl = ""
        else:
            title = lend_data[tmp[0]][1]
            diff_ttl = ' class="kuitigai"'

        # 出力
        print >>o, '<tr class="checkout">'
        print >>o, "<td>%s</td><td%s>%s</td><td%s>%s</td>" % (tmp[0], diff_auth, author,
 diff_ttl, title)
        print >>o, "</tr>"
    else:
        # 出力
        print >>o, "<tr>"
        print >>o, "<td>%s</td><td>%s</td><td>%s</td>" % (tmp[0], tmp[1], tmp[2])
        print >>o, "</tr>"

inputfile2.close()

print >>o, """</tbody></table>
<p>
黄色く塗りつぶされているものは現在貸し出し中。<br>
著者名、タイトルが赤字のものは要データチェック。
</p>
</body>
</html>"""

o.close()
```

スタイルシートを持ちだしてきましたか。やるなあ。

知らない人にスタイルシート(CSS)のことを簡単に説明しておくと、HTML全体の見え方を一律に定義しておける仕組みです。詳しくは省きますが、たとえばHTML中に .checkout { background: yellow; } なんて記述が見えますね。これで、今後、「checkoutスタイルで表示」という指示をすると、そこはみな黄色い背景色になるわけです。あとで黄色を水色に変更したくなったときに、HTML内に散在する色指定を全部直すより、スタイルシートを一カ所だけ直す

ほうがずっと楽でしょ。まあ、そんな感じのものです。できる人は使ってもいいけど、できないからといってガッカリすることはありませんよ。筆者だってスタイルシートは得意ではありません。

で、スタイルシートはともかく、スクリプト全体の流れを拝見します。

スクリプトの一番最初でR氏への貸出中データを読み込むのはほぼ同じ。lend_dataという名前の辞書に格納しました。

そしてその直後に、HTMLの不変な前半部分を一気に"""を使って出力。テーブルの見出し列まで作ってくれています。（ああ、そういえば、ボスの提示した仕様書には見出し列まで書いてあったなあ）<col> タグとか <thead> <tbody> タグの使い方は実は筆者はよく知りませんが、ふーん、こんな感じに使うのか。

で、いよいよ蔵書データを読み込みながらの処理ですが、まず

```
    # tmp[0]蔵書番号、tmp[1]著者名、tmp[2]タイトル
```

とコメントを書いてくれているのが親切ですね。何番目に何のデータが入ってたっけ、と迷わないで済みますから。

で、次にR氏に貸出中か、タイトル表記はあってるか、著者名はあってるか、ということを調べるためにifを重ねて書いていくわけですが、ここで直接HTMLを出力していないところが注目に値します。author, diff_auth, title, dif_ttl という変数を用意して、そこに必要な情報を入れておくだけとなっています。で、面倒なif文が終わったあとで、その変数の内容を使ってHTMLを組み立てると。これはスクリプトを短くする（＝ミスが減る）効果があって、いやー、スマートですね。

で、HTMLの最後には、黄色とか赤字がどういう意味かをちゃんと注記してくれます。ボスはこんなことをわざわざ指示しませんでしたが、実際は見る人にとって必須の情報なんですよね。

完璧と思います。

今後もご指導ご鞭撻よろしくお願いします！（筆者が）

まだどうぞ