目次へ

ファイルへの書き出し

前回までに、外部のテキストファイルを読み込んで何かの処理をする方法を見てきました。

外部ファイルのもう一つの重要な使い方は、データを書き出すことです。

直前の回（オマケの回）で、残業時間の統計をファイルに書き出しておいて、あとでそれを使ってExcelに結果をコピーしてグラフ化する、という例を見せました。これを作った方法をここで再現しておきましょう。タブ区切りのデータを作ったのも目新しい点です。これをどうやったかについても簡単に書きます。

書き出しモードで、ファイルを開く

ファイルの書き出し準備をするための最初の書き方は、こうです。

```
outfile = open("report.txt", "w")
```

今までにいくつか書いたopenに、"w"という指定を付け足します。'write'っていう意味でしょうね。これで、"report.txt"というファイルに何かを書き出す準備ができました。「ファイルを開いているよという目印」がoutfileという変数に入りましたから、今後はこの変数を使いながら書き出し処理を表現します。（outfileという変数の名前は適当ですので、実際はなんでもいいです）

で、実際に何かを書き出すときは、こうです。

```
print >>outfile, "Hello, world"
```

今までにたくさん見たprint文ですが、＞＞outfile, という部分が足されています。これが、「printする内容は、画面に表示する代わりに outfile で示すファイルに書き足しなさい」という指示になります。

こういった手順を参考に、今までやった print をちょっと手直しすればOK。コマンドプロンプト（今まで見てきた、黒地に白の画面）に結果を出すのも、ファイルに書き出すのも、実際のところはほとんど同じようなもんです。

で、この直前の「オマケの回」でやってみせたエクセルへのデータ連携に使ったデータは実際にどんな感じのスクリプトで出力したか、お見せします。こんな感じ。

3_example.py

```
report = {}
for line in open("leavetime3.txt"):
  a = line[:-1].split(",")
  ym = a[0][0:6]
  k = ym + "¥t" + a[1]       …(1)
  lv = a[2].split(":")
  z_minute = (int(lv[0]) * 60 + int(lv[1])) - (17 * 60 + 20)
  if z_minute < 0:
    z_minute += 24 * 60
  report.setdefault(k, 0)
  report[k] += z_minute

outfile = open("report.txt", "w")       …(2)

ks = report.keys()
for k in ks:
  print >>outfile,  k + "¥t" + str(report[k])      …(3)
```

「残業時間編ファイナル」の答えのようにみえて、ちょっと違います。あの練習問題は「残業最長記録」を探せっていうものでしたからね。今回は、全部の統計情報を単に出力してしまうだけです。

順に説明します…っていうか、まあ、前半はおおかた分かっていただけるはずなので、省略です。reportという名前の辞書にそれぞれ統計情報が溜まっていくんだと分かればそれでいいです。

でも、(1) の部分はちょっと説明がいるでしょうか。辞書のキーを作るのに、今までは "201006,Yamamoto" といった感じにコンマ記号を使いましたが、今回は　"\t" を使います。タブ記号のことです。Excelにコピペするときに、タブ記号で区切られたデータは勝手にセルに分かれて入りますから、あらかじめキーを作るときにもこういった細工を施しておくのです。

(2)の部分は、「今から report.txt っていう名前のファイルに何か書き出すつもりなんだ。目印はoutfileと名づけるよ」という宣言ですね。"w" がついているところが、普通に読むときとの違いです。この部分が実行されると、すでに report.txt っていう名前のファイルがあるときも、いったん空っぽのファイルになって、また最初から書き出すことになります。（大事なファイルをこうやって失ってしまう、という危険もあるわけなので、よく注意しましょうね。今回みたいなものなら、何回失ったって怖かないけど）

通常、この書き方だと、スクリプトを作ったところと同じフォルダの中に report.txt っていうファイルが出現します。見つけやすいし、まずはこれでいいよね。

(3)が、いままでのprintと少し違うところですね。表示したいものは、 k + "\t" + str(report[k]) の部分です。こいつをさっき開いた出力用ファイルに向けたいので、»outfile, というのを先につけたというわけ。

str( )は、紹介するのは初めてでしたね。int( )の逆だと思ってくれればよいです。report辞書の中からなにかのキーで値を出すと、このままだと数字が入っているはずです。ここでは文字を「+」でつなげて出力しようとしているんですが、数字が入ってくると足し算と間違えられてしまうんです。もちろん文字列と数字の足し算はできません。str( )ってのは、中に数字が入ったときに、こいつを文字列に直すという機能をもった関数です。

(3)についてもうちょっと。ここで k ってのは、report辞書のキーがひとつづつ入ってくる場所です。キーは、"201006\tYamamoto" とかいう風に入れていましたね。なので、ここのprintは、結果として "201006\tYamamoto\t1250" といった感じの出力になるのでした。

まあ、実行結果でも確かめておいてください。

ここで示したスクリプトをコピーするなりして 3_example.py というファイルに格納し、実行します。スクリプトファイルの横にできた report.txt を適当なエディタで開いてみればわかります。（「メモ帳」を使って開くと、タブ記号の表示がわかりにくいかも知れません。「サクラエディタ」とか、またはもっと他のやつとか、タブ記号が目に見えるやつを使ったほうがよいです） あ、「…(1)」とか「…(2)」とかは消してから実行してくださいね。これは説明のために書き込んじゃっただけのもので、スクリプトとしては無効ですから。

keys()でキー一覧を得るとき、それの並び順はそろっていません。結構まちまちな順番で結果が出力されるでしょう。Excelなんかをつかうなら、ソートもそこでやればいいんじゃない？ と思うので、手を抜けると思ったらこんな風に適当にやってもいいですよね。

えーと、この回は、練習問題はなし、かな。