目次へ

作ってる最中のスクリプトについて、具合を確認する

作ったスクリプトが一発で意図どおりに動くということは、実際のところ、めったにありません。短いものならまだしも、だんだんと書くスクリプトが長くなってくると、「動いたらラッキー」くらいの気持ちになってくるんじゃないですか。それは別にあなただけのことではなく、たとえば筆者も、実のところはそんなもんです。

ということで、スクリプトがちゃんと動いているかを、その途中経過をみながら確かめるという方法について考えます。まあ、今から結論を言ってしまうと、「printでいろいろ晒して見るといいよ」ということなんですけど。

ただの文法エラーなら、途中経過が云々とかいうこをそんなに悩むこともないです。実行開始したら直ちに、SyntaxError とか行番号とかが出てきて、このあたりでエラーがあるよ、とpythonが教えてくれますから、そこだけ確認すればほとんどはOKです。

めんどいのは、ちゃんとスクリプトは実行が終わったのに、意図した結果になってない、というときですね。または、文法エラー以外のエラーが出てきて実行中のあるタイミングで終わってしまうときも。（そういえばエラーメッセージのちゃんとした読み方についてはあんまり説明してないですね。英語だけど、ちょっと読めば見当つくんじゃない？　というくらいの姿勢でいるんですけど…）

例をひとつ挙げながら要領を見てみると話が早いでしょうか。先にやった県の鳥リストの練習問題に取り組んでいる最中にどんな途中確認がありうるか、そんなのを見てみましょう。間違いの少ない簡単なスクリプトがちょっとづつ育っていくという様子も確認してもらえるといいです。エラーを直すときのヒントも含まれているはずですよ。

最初に、県の鳥のデータファイルを読むとき。まずはデータがちゃんと一行ずつ読み込めているかなあ、というところが自信ないかも知れません。だったらまず下のようなスクリプトまでを書いて、実行結果を確かめることにしましょう。

```
#coding: cp932
for line in open("birds.txt"):
  print line
```

実行結果はこんなもん。

```
中国　　広島　　アビ
近畿　　兵庫　　コウノトリ
東北　　岩手　　キジ
近畿　　京都　　オオミズナギドリ
関東　　群馬　　ヤマドリ
...
```

単に一行づつ読み込んで出力するだけですから、そうそうは間違いません。

でもここで一応注意すべきは、行の間が妙にスカスカなことです。ここまで読んだ人はお分かりでしょうが、読んだ行がlineに入るとき、改行文字も一緒に入ります。で、print文も改行文字を出力してくれるから、結果として改行が一行についてふたつづつ挟まってしまうんですね。

これが、読みこんだ行は最後の一文字を切り詰めなさいという理由です。まあ、復習ですね。

で、これをもとに、もうちょっと難しいことをするスクリプトに育てました。

```
#coding: cp932
for line in open("birds.txt"):
  a = line[:-1].split("¥t")
  print a
```

line変数の最後の一文字は切り詰めて、それをタブ文字でsplitしてますね。ここまでうまくいっているかを、aリストをprintして確かめようとしています。

で、実行。

```
['¥x92¥x86¥x8d¥x91', '¥x8dL¥x93¥x87', '¥x83A¥x83r']
['¥x8b¥xdf¥x8bE', '¥x95¥xba¥x8c¥xc9', '¥x83R¥x83E¥x83m¥x83g¥x83¥x8a']
['¥x93¥x8c¥x96k', '¥x8a¥xe2¥x8e¥xe8', '¥x83L¥x83W']
['¥x8b¥xdf¥x8bE', '¥x8b¥x9e¥x93s', '¥x83I¥x83I¥x83~¥x83Y¥x83i¥x83M¥x83h¥x83¥x8a']
['¥x8a¥xd6¥x93¥x8c', '¥x8cQ¥x94n', '¥x83¥x84¥x83}¥x83h¥x83¥x8a']
...
```

ぎょっ？

なんか、リストの中身が変な感じに壊れているようですが...

コレは実は壊れていません。日本語文字列を含んだリストを、printで直接表示しようとするとこんな風になってしまうという特徴がpythonにはありまして、ちょっと詳しい説明はそのうちしようと思います。目印としては、「\x」という文字がたくさん出てくる場合は、別に問題ないときが多い、とでも今は理解してください。

リストを一度に表示しようとするとこんな風ですが、そのなかの要素をひとつだけ選んでprintすると大丈夫なんです。だから下のように一時的にスクリプトを直して、それなりに意図通りのデータが入っているんだと一応確認しておきましょうか。

```
#coding: cp932
for line in open("birds.txt"):
  a = line[:-1].split("¥t")
  print a[0]
```

これの実行結果はこんなふう。

```
中国
近畿
東北
近畿
関東
...
```

とりあえず最初の一要素だけ表示してみて、ああ予定どおりの文字が入っているな、と確認しました。a[1]は県名で、a[2]は鳥の名前が入っているんでしょうね。ここらへんの確認は、とりあえずいいでしょう。

さて、今回はこの各行をあとで並び替えたいんですから、全部いきなりprintしちゃうんじゃなくて、別にリストをひとつ作っておいて、そいつに順に付け足していくべきなんです。じゃあ、birdsという空っぽのリストを準備しておいて、こいつにできあがったaリストをappendしていきましょう。

birdsリストが予定どおりにできあがったか、最後にprintして確認…できるもんなのか、まずは試してみましょうか。

```
#coding: cp932
birds = []
for line in open("birds.txt"):
  a = line[:-1].split("¥t")
  birds.append(a)

print birds
```

```
[['¥x92¥x86¥x8d¥x91', '¥x8dL¥x93¥x87', '¥x83A¥x83r¥r'],
 ['¥x8b¥xdf¥x8bE', '¥x95¥xba¥x8c¥xc9', '¥x83R¥x83E¥x83m
¥x83g¥x83¥x8a¥r'], ['¥x93¥x8c¥x96k', '¥x8a¥xe2¥x8e¥xe8'
, '¥x83L¥x83W¥r'], ['¥x8b¥xdf¥x8bE', '¥x8b¥x9e¥x93s',
¥x83I¥x83I¥x83~¥x83Y¥x83i¥x83M¥x83h¥x83¥x8a¥r'], ['¥x8a
¥xd6¥x93¥x8c', '¥x8cQ¥x94n', '¥x83¥x84¥x83}¥x83h¥x83¥x8
a¥r'], ['¥x96k¥x97¥xa4', '¥x95x¥x8eR', '¥x83¥x89¥x83C¥x
83`¥x83¥x87¥x83E¥r'], ['¥x93¥x8c¥x96k', '¥x9O¥xc2¥x9OX'
, '¥x83n¥x83N¥x83`¥x83¥x87¥x83E¥r'], ['¥x92¥x86¥x8d¥x91
', '¥x8eR¥x8c¥xfb', '¥x83i¥x83x¥x83d¥x83¥x8b¥r'], ['¥x9
2¥x86¥x8d¥x91', '¥x93¥x87¥x8d¥xaa', '¥x83I¥x83I¥x83n¥x8
3N¥x83`¥x83¥x87¥x83E¥r'], ['¥x8b¥xdf¥x8bE', '¥x8eO¥x8fd
', '¥x83V¥x83¥x8d¥x83`¥x83h¥x83¥x8a¥r'], ['¥x96k¥x97¥xa
4', '¥x9OV¥x8a¥x83', '¥x83g¥x83L¥r'], ['¥x8el¥x8d¥x91',
 '¥x93¥xbf¥x93¥x87', '¥x83V¥x83¥x89¥x83T¥x83M¥r'], ['¥x
93¥x8c¥x8aC', '¥x92¥xb7¥x96¥xec', '¥x83¥x89¥x83C¥x83`¥x
83¥x87¥x83E¥r'], ['¥x8b¥xdf¥x8bE', '¥x8e¥xaO¥x89¥xea',
'¥x83J¥x83C¥x83c¥x83u¥x83¥x8a¥r'], ['¥x93¥x8c¥x8aC', '¥
x8eR¥x97¥x9c', '¥x83E¥x83O¥x83C¥x83X¥r'], ['¥x96k¥x97¥x
a4', '¥x95¥x9f¥x88¥xe4', '¥x83c¥x83O¥x83~¥r'], ['¥x8b¥x
e3¥x8fB', '¥x95¥x9f¥x89¥xaa', '¥x83E¥x83O¥x83C¥x83X¥r']
, ['¥x93¥x8c¥x96k', '¥x8eR¥x8c`', '¥x83I¥x83V¥x83h¥x83¥
x8a¥r'], ['¥x8b¥xe3¥x8fB', '¥x89¥xab¥x93¥xea', '¥x83m¥x
83O¥x83`¥x83Q¥x83¥x89¥r'], ['¥x8b¥xe3¥x8fB', '¥x8e¥xad¥
x8e¥x99¥x93¥x87', '¥x83¥x8b¥x83¥x8a¥x83J¥x83P¥x83X¥r'],
```

うーん、薄々分かっちゃあいました。birdsは全データがひとつのリストにくっついているわけなので、なんせデカいに決まってます。

あ、でも、一番最初に [[で始まってますから、リストのリストがbirds変数に入ったのかな、と、気づける人は気づいてください。

次は、できあがったbirdsリストを改めてひとつづつprintさせてみることにします。そのとき、またリスト表示は壊れますから、頭の一要素だけをprintしましょう。

```
#coding: cp932
birds = []
for line in open("birds.txt"):
  a = line[:-1].split("¥t")
  birds.append(a)

for i in birds:
  print i[0],
print
```

実行結果は下のよう。あまりさっきと変わらない。でもデータをいったん溜め込んでから改めて吐き出したんだから、意味は違っています。

```
中国 近畿 東北 近畿 関東 北陸 東北 中国 中国 近畿 北陸 四国 東海 近畿 東海 北陸
九州 東北 九州 九州 関東 北陸 中国 九州 関東 九州 北海道 近畿 関東 中国 九州 東
海 関東 近畿 東北 関東 四国 九州 東北 四国 関東 近畿 東北 東海 四国 東海 九州
```

あ、縦にprintして出力が流れていくのはちょっと不便なので、今回はprintに「,」を加えました。これで改行を省きながら確認ができます。（どっかでこの方法は説明したっけ？　してなかったら、まあ、今覚えておくといいかもね。）

あ、ついでにこのタイミングで、表示が縦にズラズラ流れてしまうのをちょっとづつ確かめる方法について説明しておきます。テキストのはじめのほうで紹介した実行方法を取っているなら、黒地に白のコマンドプロンプトから、スクリプト名を打ち込んでEnter、という風に動作を試しているでしょう。下みたいに。

```
> some_script.py
（実行結果…）
```

このときちょっと書き加えて、

```
> some_script.py | more
```

と書き加えから[Enter]とすると、一画面分の表示だけしてから、

```
（実行結果が途中まで出力）
-- More --
```

と、こんな表示がされます。ここでスペースキーを押すと一画面進み、最後まで表示し終わるとまたプロンプトの入力ができるようになります。(Ctrl+Cで途中終了)

こいつを覚えておくと、たくさん表示されるものを確実に全部確認できますよ。

まあ、それはともかく、作ってる最中のスクリプトに戻ります。

鳥のリストは地域ごとに並べ替えをしたいという指定ですから、birdsリストをsortで並べ替えることにしました。うまくいってるか確認したいから、またさっきと同じようなprintを行います。

```
#coding: cp932
birds = []
for line in open("birds.txt"):
  a = line[:-1].split("¥t")
  birds.append(a)

birds.sort()

for i in birds:
  print i[0],
print
```

実行結果は下。

```
関東 関東 関東 関東 関東 関東 関東 近畿 近畿 近畿 近畿 近畿 近畿 近畿 九州 九州
九州 九州 九州 九州 九州 九州 四国 四国 四国 四国 中国 中国 中国 中国 中国 東海
東海 東海 東海 東海 東北 東北 東北 東北 東北 東北 北海道 北陸 北陸 北陸 北陸
```

おお、並んだ並んだ。

さて、次は、この並び替え後のデータを順に見ながら、地域が変化したタイミングで「見出し」を表示するという仕様でしたから、これを実現させます。

メモしておく、という意味をスクリプト上にも表しておきたいから、導入する変数の名前はmemoとしましょうか。

```
#coding: cp932
birds = []
for line in open("birds.txt"):
  a = line[:-1].split("¥t")
  birds.append(a)

birds.sort()

memo = ''
for i in birds:
  if memo != i[0]:
    print "break!"
    print "new area is %s" % i[0]
    memo = i[0]
  print i[0]
```

最初にmemoは何にもない文字列にしておいて、最初の一行目が必ず見出し行として反応されるようにします。

で、if memo != ... というところが「見出しを表示すべきかな」という判断の部分です。このとき、見出し行をここで出力するはずだ、という目印をまずはprintしてみて、この実現方法が正しいかを確認します。ここで「break!」と表示するんですが、データの並びに一区切りついた状態を、プログラム屋さんは「ブレイク」と呼ぶことがあるから、まあ試しに書いてみました。

見出し行を表示したら、memoを更新しておくことも必要です。

で、実行結果...

```
break!
new area is 関東
```

```
関東
関東
関東
関東
関東
関東
関東
break!
new area is 近畿
近畿
近畿
近畿
近畿
近畿
近畿
近畿
break!
new area is 九州
九州
...
```

ちゃんと予定どおりのタイミングで見出しが表示できそうだと分かります。そろそろ地域名だけ表示するんじゃなくて、県名と鳥の名前を出すように書き換えましょう。変更部分は、ほんのちょっとです。

```
#coding: cp932
birds = []
for line in open("birds.txt"):
  a = line[:-1].split("¥t")
  birds.append(a)

birds.sort()

memo = ''
for i in birds:
  if memo != i[0]:
    print "break!"
    print "new area is %s" % i[0]
    memo = i[0]
  print i[1], i[2]
```

実行結果。

```
break!
new area is 関東
茨城 ヒバリ
群馬 ヤマドリ
埼玉 シラコバト
神奈川 カモメ
千葉 ホオジロ
東京 ユリカモメ
栃木 オオルリ
break!
new area is 近畿
京都 オオミズナギドリ
三重 シロチドリ
滋賀 カイツブリ
大阪 モズ
奈良 コマドリ
兵庫 コウノトリ
和歌山 メジロ
break!
new area is 九州
沖縄 ノグチゲラ
...
```

ここまで来れば、あとは、いわば「裸の」printで済ましていたところを、HTMLの要素として出てくるように整形していくだけですね。必要な「ロジック」、つまり繰り返しや判断のタイミングや要領は完成したと判断できます。あとは飾りの問題です。

とまあ、こんな風に一歩ずつ足場を確かめながら作るといいですよ。問題が簡単なうちに見つかって直せますからね。

なんせ、コツは、printしまくって、目視です。なんか期待と違う実行結果だなあと思ったときも、こんな感じで実行中に変数の中身とかをprintして、頭の中ではこうなるはずなんだという結果を照らしあわせていくんです。

地道ですね。他にもデバッガとか何とかという、実行途中をトレースするためのうまいツールが存在するんですが、まあ、いちばん手っ取り早く試せるのは、やっぱりこの方法ですよ。